Š^ç^!æ ā * Á! [&•• Áã & ç^! ^ Á ã ãæ & Á~ • ç^! ā * Áæ å Á^ çç
{ ā ā * Á! Á ç || ā ^ } óæ æ • ā Á Á &ã ^ } ó æ æ ^ { ^ } Á! [&•• ^•
R &@ } ÁÖ^ Á ^ ^! å ËU^]] ^ Áæ å ^ } ÁÓ! [~ & ^ ÉRæ Áæ çæ } ^ } Éæ å Áæ ç Áæ • ^ } •

DEPARTMENT OF DECISION SCIENCES AND INFORMATION MANAGEMENT (KBI)

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

Jochen De Weerd^{*}, Seppe K.L.M. vanden Broucke^{*}, Jan Vanthienen^{*}, and Bart Baesens^{*†}

^{*}Department of Decision Sciences and Information Management, KU Leuven

Naamsestraat 69, B-3000 Leuven

Email: Jochen.DeWeerd^t@econ.kuleuven.be

[†]School of Management, University of Southampton

Highfield Southampton, SO17 1BJ, United Kingdom

Abstract—Recent years have witnessed the ability to gather an enormous amount of data in a large number of domains. Also in the field of business process management, there exists an urgent need to beneficially use these data to retrieve actionable knowledge about the actual way of working in the context of a certain business process. The research field concerned is process mining, which can be defined as a whole family of analysis techniques for extracting knowledge from information system event logs. In this paper, we present a solution strategy to leverage traditional process discovery techniques in the flexible environment of incident management processes. In such environments, it is typically observed that single model discovery techniques are incapable of dealing with the large number of different types of execution traces. Accordingly, we propose a combination of trace clustering and text mining to enhance process discovery techniques with the purpose of retrieving more useful insights from process data.

I. INTRODUCTION

Process mining has been demonstrated to possess the capabilities to profoundly assess business processes [1]. Within this domain, process discovery is considered the most important learning task. Process discovery entails the extraction of process models from information system event logs, which are basically sets of process executions as captured by some kind of logging infrastructure. Due to the fact that event logs rarely contain negative events to record that a particular activity could not have taken place, process discovery remains a largely unsupervised learning task [2].

In highly flexible environments, business process data in the form of an event log typically exhibit a large variety of control-flow behavior. This is due to the low number of restrictions imposed by the information system with respect to the possible executions steps for a certain case at a certain moment in time.

Nonetheless, process mining can be valuable in such environments, since these intelligent techniques allow to provide insight into the actual behavior of the system. However, in flexible environments, traditional single model process discovery techniques [2]–[6] are confronted with an important challenge. Due to the high variety of process behavior, it often becomes problematic to discover process models that are both accurate and comprehensible.

With this paper, we propose a practicable methodology to leverage process discovery techniques with a combination of trace clustering and text mining. Trace clustering is used to separate execution traces into different groups for which a more accurate and comprehensible process model can be discovered. Next, a combination of text mining and decision tree learning is employed to get insight into the atypical behavior in the system. To this purpose, the paper is structured as follows. Section II outlines the problem statement and describes the practical context in which this methodology is proven useful. Further, in Section III a novel trace clustering technique is proposed and it is demonstrated how this technique can be used to discover more accurate and comprehensible process models. In addition, Section IV describes how deviations from highly frequent behavior can be characterized with the help of text mining and as such contribute to tangible knowledge discovery. In Section V, a number of key discussion points are commented on. Finally, Section VI presents some related work before the conclusions are formulated in Section VII.

II. PROBLEM STATEMENT

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

A. Incident Management Process: Data Description

Over a six month period, data about 2.726 cases was gathered. For each of these cases, two different types of data are available: case data and event log data. Case data, as depicted in Table I, consists of an ID, a textual description and a type. In the system, four different types are distinguished:

Bugfix/Repair, How-to/Advice, Productive Intervention and New Need. Note however that for 40% of the cases, the type is not specified. Next to type information, the data also consists of a textual description. The only restriction with regard to this field is that the text should be limited to at most 256 characters.

TABLE I
EXCERPT OF CASE DATA

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

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

TABLE II
EXCERPT OF EVENT LOG DATA

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

B. Process Discovery in Flexible Environments

As stated earlier, process discovery is the most challenging learning task in the process mining domain. The crucial problem is caused by process data stemming from highly

flexible environments. In such environments, little restrictions are put in place to limit the possible execution steps. In this way, there exists a wide variety of execution paths in the event log. Also for the running example data set, a wide variety of control-flow behavior is present, as illustrated by the fact that 338 different types of execution paths (further also denoted as grouped traces or *gt*'s) are found. Furthermore, these various types of traces differ significantly in terms of their frequency. This is depicted with the histogram in Figure 1, which has trace frequency intervals on the horizontal axis and the amount of distinct grouped traces belonging to such a trace frequency interval on the vertical axis. The figure clearly shows that the distribution of trace frequencies (i.e. the amount of cases that follow an identical solution trajectory in terms of the activities performed) is strongly skewed. On the one hand, a lot of low-frequency behavior is observed. For instance, there exist 211 cases that have a unique execution trace within the system. On the other hand, many cases comply with standard solution strategies in terms of their control-flow sequences. This is illustrated by the low number of highly frequent traces that occur in the event log. As such, more than 55% of all traces conform to one of the ten most frequent grouped traces.

Fig. 1. Histogram of trace frequency intervals for the event log

The availability of a wide range of execution behavior makes the application of single model process discovery techniques difficult. What is typically observed is that such techniques fail to discover an accurate and comprehensible process model out of the data. Also for the running example, this is the case. Figure 2 and Table III illustrate the problem at hand. The figure shows the Petri net process model as discovered by the HeuristicsMiner algorithm [5]. As can be seen from the table, this model scores badly with respect to accuracy. For instance, only 0,9% of the traces in the log can

be correctly replayed (cfr. the number of successfully executed traces *set*). Similarly, behavioral recall (r_B^p) of this model equals 0,595 while behavioral precision (p_B) is 0,506. As shown in [7], these values can be aggregated with the F-score to construct an overall measure for accuracy. The F1-score for this process model is only 0,547.

Next to accuracy, the comprehensibility of the discovered process model is mediocre as well. First of all, the discovered Petri net is an unsound workflow net. Furthermore, the model shows a high number of invisible transitions. Together with the high number of arcs connecting the nodes in the net, it should be clear that this model is not ideal for gaining insights into the actual behavior of this business process. Accordingly, we propose a complementary approach for analyzing the data that combines trace clustering and text mining, as shown in Figure 3.

TABLE III
EVENT LOG AND DISCOVERED PROCESS MODEL CHARACTERISTICS FOR THE ENTIRE EVENT LOG

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

C. Solution Strategy

The solution strategy for leveraging process discovery in the current setting of a highly flexible incident management process consists of two important elements. First of all, trace clustering is applied with the purpose of dividing the event log traces into sub-event logs. This division will allow process discovery techniques to discover more accurate and comprehensible models from subsets of event log traces. With respect to trace clustering, a novel approach is outlined in section III. This novel trace clustering technique will yield a number of clusters for which an accurate and comprehensible process model can be discovered. However, a number of traces in the event log do not fit well in one of these created clusters because they exhibit atypical control-flow behavior. Therefore, these cases are separated out to form the subject of the text mining phase. In this second phase, a combination of text mining and data mining is proposed with the purpose of finding interesting patterns for the atypical cases. Instead of relying on control-flow data (event log), we will make use of the case data, both the unstructured description field and the type information, since it can be expected that a lot of knowledge resides in these text fields. The whole solution strategy is represented in Figure 3.

Fig. 3. Overview of the proposed solution strategy

III. TRACE CLUSTERING

Within the field of process mining, Greco et al. [8] were the first to apply clustering techniques to event traces. The idea of grouping traces exhibiting common behavior is an intuitive approach to reduce the complexity of the discovery problem. By separating execution traces into different groups, one can rely on multiple process models to represent the variety in behavior of a certain event log. Also Song et al. [9] and Ferreira et al. [10] contributed to the development of trace clustering techniques.

The state-of-the-art techniques are described in [11], [12]. The authors propose different clustering algorithms that exploit the context of a process as reflected in the event log traces. For instance, by making use of conserved patterns as attributes in a vector space model, existing distance or similarity metrics can be used in combination with traditional clustering techniques such as hierarchical clustering in order to combine traces into trace clusters. In this study, we propose an adaptation of the MRA-based clustering technique proposed in [11], by changing the underlying hierarchical clustering technique to a semi-supervised clustering approach.

A. A Novel Semi-Supervised Trace Clustering Technique

The purpose of trace clustering is to divide the traces in an event log into multiple sub-event logs with the idea that for these event logs better process models can be discerned with existing process discovery techniques. However, the currently available trace clustering techniques do not explicitly incorporate the evaluation criteria in terms of the accuracy and comprehensibility of the process models underlying the formed clusters in their learning strategies. Instead, these techniques are founded on the existing principle in the field of data mining to maximize inter-cluster dissimilarity while maximizing intra-cluster similarity. As a result, these techniques only assume that the process models underlying the formed trace clusters will be better.

Building further on this idea, our novel clustering approach, denoted *SemSup-MRA*, makes the accuracy improvement in terms of the underlying process models explicit. The technique is outlined in Algorithm 1. *SemSup-MRA* involves a greedy optimization strategy targeting the fitness of the underlying process model of each of the created clusters. This is realized with the implementation of an intelligent selective sampling

Fig. 2. Petri net result of the HeuristicsMiner algorithm performed on the entire event log

procedure. More specifically, the whole algorithm consists of four phases. In a first phase, identical traces in the event log are grouped into grouped traces. In a second phase, grouped traces are iteratively selected with the purpose of being added to the current cluster. This selection procedure is carried out as follows. First, from the entire set of grouped traces, the most frequent trace is added to the current cluster. Then, within a frequency based window, the most similar trace with respect to the traces already in the current cluster is selected relying on the MRA-based Euclidean distance between these traces. This trace is added to the current cluster, only if the resulting process model as discovered with the HeuristicsMiner algorithm remains fully accurate in terms of the ICS-fitness [13]. If this is not the case, the selected trace is not added to the current cluster. At this moment, there exist two options. Either the set of traces in the current cluster is large enough; in this case the selection procedure is halted. On the other hand, if the current cluster is not large enough in terms of the number of traces, the currently selected trace is skipped and the selection procedure is repeated for the remaining traces in the event log.

As soon as the current cluster contains at least $\beta\%$ of the traces in the event log and a grouped trace is encountered for which the resulting process model shows a fitness lower than 1, the third phase of the algorithm initiates. For the current cluster, it is verified whether the set of remaining traces (these traces have not yet been subject of selective sampling) contains traces that fit perfectly in the current cluster's process model.

Each trace that qualifies for this criterion, is added to the current cluster and removed from the set of remaining traces. Note that the fitness of single trace is also evaluated with the ICS-fitness, however only considering this single trace. Once this procedure is terminated, a second iteration of phase two of the algorithm will initiate starting from the set of remaining traces. However, as soon as the maximum number of clusters is reached, the selective sampling procedure is halted. In a fourth and final phase, a residual cluster is created grouping all the remaining traces.

Note that we term this clustering technique *SemSup-MRA* since a semi-supervised learning strategy is employed by making use of a selective sampling procedure in combination with a greedy accuracy optimization method for each cluster's underlying process model. Furthermore, the MRA-based ([11]) Euclidean distance matrix is used within this selective sampling procedure in order to comply with the initial idea of trace clustering to group similar behavior together. Finally, the technique is implemented as the *ActiTraC plugin* in the ProM-framework¹ [14].

B. Results

In accordance with the solution strategy proposed in Section II-C, *SemSup-MRA* is applied to the event log of the running example. In this case we opted to create four clusters, with

¹<http://www.promtools.org/prom6/>

Algorithm 1 SemSup-MRA

```

1: Group identical traces  $t$  in log  $L$  into grouped traces  $gt$  in
   grouped log  $GL$ 
2: Each  $gt$  in  $GL$  has a certain frequency  $f$ 
3:  $CS$  denotes the set of clusters, initially empty
4:  $R$  denotes the set of remaining  $gt$ 's, initialized to all the  $gt$ 's in
    $GL$ 
5: repeat
6:    $C$  denotes the set of  $gt$ 's in the current cluster, initially empty
7:    $I$  denotes the set of ignored  $gt$ 's, initially empty
8:   repeat
9:     if  $C = \emptyset$  then
10:      Add the most frequent  $gt$  in  $R$  to  $C$ 
11:     else
12:       Consider  $W$  to be set as the top  $\alpha\%$  of  $gt$ 's in  $R$ 
         according to frequency
13:       Select the  $gt$  in  $W$  that has the lowest average MRA-
         based Euclidean distance with respect to the  $gt$ 's
         already in  $C$ 
14:       Denote this  $gt$  as  $gt_{cur}$ 
15:       Apply HeuristicsMiner to  $(C \cup gt_{cur})$  with process
         model  $PM$  the according result
16:       if  $ICS\text{-}fitness(PM) \geq 1$  then
17:         Add  $gt_{cur}$  to  $C$ 
18:         Remove  $gt_{cur}$  from  $R$ 
19:       else if  $ICS\text{-}fitness(PM) < 1$  then
20:         if current cluster contains at least  $\beta\%$  of the
           remaining traces then
21:           Apply HeuristicsMiner to  $C$  with process
           model  $PM$  the according result
22:           for all  $gt$  in  $R$  do
23:             if  $gt$  fits  $PM$  then
24:               Add  $gt$  to  $C$ 
25:             Remove  $gt$  from  $R$ 
26:           end if
27:         end for
28:         add a new cluster  $cl$  to  $CS$  with  $cl$  the set of
           ungrouped traces corresp. to the  $gt$ 's in  $C$ 
29:       else
30:         Add  $gt_{cur}$  to  $I$ 
31:       end if
32:     end if
33:   end if
34:   until no more remaining  $gt$ 's or new cluster created
35: until maximum number of clusters is reached
36: Add a residual cluster  $cl$  to  $CS$  with  $cl$  the ungrouped traces
   corresponding to the  $gt$ 's in  $R$ 

```

one of these the residual cluster. Furthermore, the frequency window was set to the 10% most frequent remaining traces (parameter α) and the minimum cluster size was set to 50% of the total number of remaining traces (parameter β). Tables IV and V detail a comparison between the application of *SemSup-MRA* and the standard MRA-technique with its optimal settings for this case (F-score similarity). The tables show both event log information with respect to the clusters created as well as accuracy and comprehensibility metrics for the underlying process models.

The results show a clear difference between both approaches. While *SemSup-MRA* presents three trace clusters with highly accurate underlying process models and one residual cluster with low accuracy, the standard MRA-technique

TABLE IV
EVENT LOG AND DISCOVERED PROCESS MODEL CHARACTERISTICS FOR SEMSUP-MRA CLUSTERING

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

TABLE V
EVENT LOG AND DISCOVERED PROCESS MODEL CHARACTERISTICS FOR STANDARD MRA CLUSTERING

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

discerns only one cluster (clus_3) with an acceptable accuracy level. If one averages the F-scores of the process models according to their trace frequencies, it can be found that *SemSup-MRA* realizes a weighted average F-score of 0,88 while the original MRA-technique equals 0,69. As such, from an accuracy perspective, our novel trace clustering technique can be considered better performing.

From an event log perspective, the optimization strategy of *SemSup-MRA* drives the algorithm towards the creation of fairly large clusters in absolute terms, hereby clustering only a limited number of grouped traces together. The residual cluster (clus_resid) contains about 16% of the total number of traces, while this cluster contains more than 75% of the grouped traces. The distribution of grouped traces over the different clusters is slightly more balanced for the standard MRA-technique. A final element of comparison involves scalability. Since the computational complexity of *SemSup-MRA* is significantly larger than standard MRA, it is logical that the run times of the algorithms differ. In this case, *SemSup-MRA* takes on average about 33 seconds to yield a result, while standard MRA computes the result in about 2 seconds on a standard stand-alone PC. Note that an exhaustive analysis of the scalability of *SemSup-MRA* is out of the scope this study.

The resulting process models as discovered from the *SemSup-MRA* clustering result with the HeuristicsMiner algorithm are shown in Figure 4. Note that the process model for the residual cluster is not represented for two reasons. First of all, the according process model is inaccurate and incom-

prehensible. Secondly, these residual traces will be the subject of a different knowledge discovery approach as described in the next section.

IV. ANALYSIS OF ATYPICAL BEHAVIOR

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

A. Characterizing Deviating Traces

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

B. Text Mining

For the running example, case data is limited to type information and a fully unstructured textual description. Accordingly, it is this textual description field that is subject of a text mining analysis. The purpose of text mining is the extraction of frequent keywords that subsequently can be used in a classification learning setting which distinguishes between typical and atypical behavior. Text mining from these unstructured text data requires the filtering of stop words and stemming to prove effective. These steps were performed by making use of the RapidMiner² software. Since textual descriptions are available in both English and Dutch, stop word filtering and stemming were performed for both languages. Table VI lists the most frequent words in the data set. Since the majority of textual descriptions are written in Dutch, the most frequent words are in Dutch as well. Note that there is no optimization procedure put in place that matches different words with the same meaning in Dutch and English into one attribute word. The type information together with the

tokenized textual descriptions are now available as attributes for a standard classification learning task.

TABLE VI
FREQUENT WORD LIST

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

C. Classification Results

As stated earlier, text mining is combined with decision tree learning in order to discover useful patterns from case data. To do so, the C4.5-technique [15] was applied. Note that the data set consists of both typical traces that belong to one of the non-residual clusters and atypical behavior. Since the data set is strongly skewed as there exists much more typical traces as compared to atypical, we applied random undersampling in order to balance the data. As such, the input data for the tree learning algorithm consists of 862 traces. Furthermore, we applied 10-fold cross-validation for an assessment of the classification accuracy of C4.5 decision trees.

The result of this data mining analysis is represented in Figure 5. The tree classifies typical (0) and atypical traces (1) by making use of the type information and the tokenized textual description. It is recognized that the accuracy level of 63% is moderate, however, this is mainly due to the fact that the logged textual descriptions are fairly short. The methodology could be more advantageously applied in case there would be more textual information available. However, the logging infrastructure currently could not provide more elaborate case information.

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

²<http://rapid-i.com>

Fig. 4. Petri net result of the HeuristicsMiner algorithm performed on the trace clusters discovered by *SemSup-MRA*

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

V. DISCUSSION

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

A. Trace Clustering

The newly proposed trace clustering technique *SemSup-MRA* shows some interesting results. By optimizing the underlying process models of the clusters, it is demonstrated that currently available trace clustering techniques can be improved. However, the semi-supervised learning strategy entails a definite increase in computational complexity. In future

research, it should be investigated how the scalability of the proposed technique behaves in other real-life environments. Also a comparative evaluation with other available trace clustering techniques is subject of future research. Furthermore, the parameters of the algorithm should be further investigated as well as finding an optimal way for determining the right amount of clusters that should be created.

B. Originator Effect

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

VI. RELATED WORK

As explained earlier, process discovery [2]–[6] and trace clustering [8]–[12] are important foundations of this work. Nevertheless, also conformance checking techniques are key building blocks. For instance, the ICS-fitness metric [13] is the underlying fitness metric of *SemSup-MRA*. For evaluation, we employed the recall and precision metrics from [2] and [7]. Other conformance checking metrics could have been put in place. For instance, the original conformance metrics proposed in [16] are a viable alternative. With respect to precision, the ETC-metric in [17] is very similar to behavioral precision (p_B) as used in this study.

Further, the application of rule learning and decision tree algorithms is not new in the process mining domain. For instance, Rozinat et al. [18] developed a technique that investigates the decision points in a process model based on case data. Furthermore, Maruster et al. [19] proposed the use of Ripper [20] to the process discovery learning problem. Also Linear Temporal Logic (LTL) is often applied in the field of process mining [21]. For instance, Maggi et al. [22] founded their declarative process discovery technique on LTL.

VII. CONCLUSION

In this paper, we have developed a novel solution strategy for leveraging process discovery techniques in highly flexible environments where traditional single-model discovery techniques underperform. Our approach consists of the application of trace clustering and a combination of text mining and data mining in order to extract useful knowledge from the original data. With respect to clustering, we proposed an adaptation of the existing MRA-technique in the sense that we altered the underlying clustering technique from simple hierarchical clustering to a semi-supervised learning algorithm that greedily optimizes the accuracy of the underlying process models. The results in the context of an incident management process show that the proposed methodology is feasible and valuable. Thanks to the novel trace clustering technique, more accurate and comprehensible process models were discovered that represent the typical process behavior in the system. Furthermore, text mining and decision tree learning provided useful results with respect to characterizing the atypical behavior relying on case data instead of control-flow information.

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

ACKNOWLEDGMENT

The authors would like to thank the Flemish Research Council for financial support under Odysseus grant B.0915.09 and KU Leuven for OT/10/010.

REFERENCES

- [1] W. M. P. van der Aalst, *Process Mining - Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.
- [2] S. Goedertier, D. Martens, J. Vanthienen, and B. Baesens, "Robust process discovery with artificial negative events," *Journal of Machine Learning Research*, vol. 10, pp. 1305–1340, 2009.
- [3] W. M. P. van der Aalst, A. J. M. M. Weijters, and L. Maruster, "Workflow mining: Discovering process models from event logs," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 9, pp. 1128–1142, 2004.
- [4] A. K. Alves de Medeiros, A. J. M. M. Weijters, and W. M. P. van der Aalst, "Genetic process mining: an experimental evaluation," *Data Mining and Knowledge Discovery*, vol. 14, no. 2, pp. 245–304, 2007.
- [5] A. J. M. M. Weijters, W. M. P. van der Aalst, and A. K. Alves de Medeiros, "Process mining with the heuristicsminer algorithm," TU Eindhoven, BETA Working Paper Series 166, 2006.
- [6] M. Sole and J. Carmona, "Region-based foldings in process discovery," *IEEE Trans. Knowl. Data Eng.*, vol. 99, no. PrePrints, 2011.
- [7] J. De Weerd, M. De Backer, J. Vanthienen, and B. Baesens, "A robust f-measure for evaluating discovered process models," in *IEEE Symposium Series in Computational Intelligence 2011 (SSCI 2011)*, Paris, France, 2011.
- [8] G. Greco, A. Guzzo, L. Pontieri, and D. Sacca, "Discovering expressive process models by clustering log traces," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 8, pp. 1010–1027, 2006.
- [9] M. Song, C. W. Günther, and W. M. P. van der Aalst, "Trace clustering in process mining," in *Business Process Management Workshops*, ser. Lecture Notes in Business Information Processing, D. Ardagna, M. Mecella, and J. Yang, Eds., vol. 17. Springer, 2008, pp. 109–120.
- [10] D. R. Ferreira, M. Zacarias, M. Malheiros, and P. Ferreira, "Approaching process mining with sequence clustering: Experiments and findings," in *BPM*, ser. Lecture Notes in Computer Science, G. Alonso, P. Dadam, and M. Rosemann, Eds., vol. 4714. Springer, 2007, pp. 360–374.
- [11] R. P. Jagadeesh Chandra Bose and W. M. P. van der Aalst, "Trace clustering based on conserved patterns: Towards achieving better process models," in *Business Process Management Workshops*, ser. Lecture Notes in Business Information Processing, S. Rinderle-Ma, S. W. Sadiq, and F. Leymann, Eds., vol. 43. Springer, 2009, pp. 170–181.
- [12] R. P. Jagadeesh Chandra Bose and W. M. P. van der Aalst, "Context aware trace clustering: Towards improving process mining results," in *SDM*. SIAM, 2009, pp. 401–412.
- [13] A. J. M. M. Weijters and W. M. P. van der Aalst, "Rediscovering workflow models from event-based data using little thumb," *Integrated Computer-Aided Engineering*, vol. 10, no. 2, pp. 151–162, 2003.
- [14] W. M. P. van der Aalst, B. F. van Dongen, C. W. Günther, A. Rozinat, E. Verbeek, and T. Weijters, "ProM: The process mining toolkit," in *BPM (Demos)*, ser. CEUR Workshop Proceedings, A. K. A. de Medeiros and B. Weber, Eds., vol. 489. CEUR-WS.org, 2009.
- [15] J. Quinlan, *C4.5: programs for machine learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.
- [16] A. Rozinat, R. S. Mans, M. Song, and W. M. P. van der Aalst, "Discovering simulation models," *Information Systems*, vol. 34, no. 3, pp. 305–327, 2009.
- [17] J. Munoz-Gama and J. Carmona, "A fresh look at precision in process conformance," in *Business Process Management*, ser. Lecture Notes in Computer Science, R. Hull, J. Mendling, and S. Tai, Eds., vol. 6336. Springer, 2010, pp. 211–226.
- [18] A. Rozinat and W. M. P. van der Aalst, "Decision mining in ProM," in *Business Process Management*, ser. Lecture Notes in Computer Science, S. Dustdar, J. L. Fiadeiro, and A. P. Sheth, Eds., vol. 4102. Springer, 2006, pp. 420–425.
- [19] L. Maruster, A. J. M. M. Weijters, W. M. P. van der Aalst, and A. van den Bosch, "A rule-based approach for process discovery: Dealing with noise and imbalance in process logs," *Data Mining and Knowledge Discovery*, vol. 13, no. 1, pp. 67–87, 2006.
- [20] W. Cohen, "Fast effective rule induction," in *Proceedings of the 12th International Conference on Machine Learning*, A. Prieditis and S. Russell, Eds. Tahoe City, CA: Morgan Kaufmann Publishers, 1995, pp. 115–123.
- [21] W. M. P. van der Aalst, H. T. de Beer, and B. F. van Dongen, "Process mining and verification of properties: An approach based on temporal logic," in *OTM Conferences (1)*, ser. Lecture Notes in Computer Science, R. Meersman, Z. Tari, M.-S. Hacid, J. Mylopoulos, B. Pernici, Ö. Babaoglu, H.-A. Jacobsen, J. P. Loyall, M. Kifer, and S. Spaccapietra, Eds., vol. 3760. Springer, 2005, pp. 130–147.
- [22] F. M. Maggi, A. J. Mooij, and W. M. P. van der Aalst, "User-guided discovery of declarative process models," in *CIDM*. IEEE, 2011, pp. 192–199.