A Simulation-Based Approach to Process Conformance

Pedro M. Martins

IST – Technical University of Lisbon
Avenida Prof. Dr. Cavaco Silva
2744-016 Porto Salvo, Portugal
pedro.m.martins@tagus.ist.utl.pt

Abstract. Organizations are seeking out new ways to monitor the run-time behavior of their business processes. With process mining techniques it becomes possible to extract process models from event logs and compare them to the expected behavior. However, each process mining technique has its own merits and different techniques produce different kinds of models. The use of an evaluation tool became fundamental. Based on process conformance, this work presents a new evaluation framework, using process simulation and a set of metrics to compare the event logs produced by different models. This framework provides an assessment of the capabilities of the available process mining techniques. Based on the same evaluation method it is also proposed an framework to evaluate process conformance between an event log and a model. With that, a generic framework evaluation for process mining and process conformance is proposed, increasing the analysis methods for both techniques.

Key words: Process Mining, Conformance Checking, Evaluation Framework, Evaluation Metrics

1 Introduction

Due either to legislation or competitive forces, in recent years there has been an increasing interest from organizations in monitoring the run-time behavior of their business processes. At the same time, business processes are being implemented, automated and supported by process-aware information systems which keep track of process execution by recording relevant data, user information and workflow events as business activities are performed. From these data it is possible to apply process mining techniques [1] to discover the run-time behavior of those processes.

The concept of *conformance checking* [2] has been proposed as way to compare the expected behavior of business processes with the behavior that has been actually observed at run-time. Usually, there will be a process model describing the expected behavior, and there will be an event log containing the events recorded during the execution of several instances from that process. Conformance checking aims at assessing the degree to which the behavior in the event log conforms with the predefined process model. This can be estimated quantitatively by means of metrics such as *fitness* and *appropriateness* [3]. Another approach to process conformance is presented in [4] where the authors make use of a logic formalism to evaluate the alignment between what they call *legal space* and the *business space*.

The relationship between process mining and conformance checking is twofold. On one hand, process mining can help discover the run-time behavior that will be checked for conformance against a predefined process model. On the other hand, conformance checking can be used as a tool to evaluate the quality of process models extracted from events logs using different process mining techniques. In this paper we address both perspectives: we develop an evaluation framework for process mining algorithms and we illustrate the use of this framework to evaluate process conformance.

There are several process mining techniques available within the ProM framework [5] and these techniques produce different kinds of models. In general, it may be difficult to compare the results of

process mining with a predefined process model if the behavior of the mined model is indeed different from the predefined model, and if in addition the predefined model has been described using a different modeling notation. To address these problems, we turn to a conformance framework that is based on simulation, where the goal is to compare two process models based on the event logs that these models are able to generate, rather than on their actual description. The approach has a probabilistic nature and through the use of a metric it is able to quantify the degree to which two models produce similar behavior.

The paper is organized as follows. Section 2 discusses the approaches to conformance checking and how simulation can be useful for this purpose. In Section 3 the two frameworks that apply this approach to the evaluation of process mining algorithms and process conformance. In Sections 4 and 5 will be presented an experiment and two case studys, respectively, to both frameworks. Finally, section 6 concludes the paper.

2 Approaches to conformance checking

It has already been mentioned that one approach to check the conformance between a process model and an event log is to use metrics such as *fitness* and *appropriateness* [2]. Among other features, these metrics evaluate how well a given process model can reproduce the workflow traces recorded in an event log. If the model is provided as a Petri net, for example, then a token will be added to the input place and in principle there will be a sequence of firing transitions matching a given trace. In this approach, the comparison is made directly between the log and the model.

On the other hand, it has also been shown that it is possible to compare process models directly, namely by making use process mining [6]. Starting from an existing event log, it is possible to mine it in order to extract a first model. Having this mined model (in *Coulored Petri Net* format [7]), is possible to simulate it and generate a new event log that can be mined again, producing a second model. Then it is possible to compare the two mined models in order to determine if both share the same *business logic* – more on this topic ahead in section 2.1.

Finally, as a third approach it is possible to perform conformance checking based on a direct comparison between event logs. There will be a first event log from the daily execution of processes in an information system, which can be used to obtain a mined model. A second event log can be obtained by simulating this mined model, so that it becomes possible to analyze how much behavior the two event logs have in common. The advantage of this approach is that this comparison can be reduced to a sequence-matching problem. With the help of a probabilistic metric it becomes possible to perform a *log-on-log* evaluation.

2.1 Simulating mined models

In the context of this work we are aiming at a direct comparison between event logs, and hence the focus will be on the generation of traces, or sequences of events. An event log is usually stored or can be converted to a MXML file, a *de facto* standard format for process mining that can be used in the ProM framework[1] [8]. A MXML file contains relevant data such as user info and workflow events produced as business activities are performed.

Once an event log is available, a mining technique can be applied to extract a process model as a Petri Net, a Heuristic Net, or other. From the discovered model, a simulation is carried out to generate a number of traces[2]. When simulating the mined model, the sequences of events that occur – basically, the transitions between activities – will be recorded in a new event log. An important feature about the simulation is that transitions may have a firing probability, which can be calculated from the results of the mining algorithms. After simulation, an analysis is performed using the input event log and

[1] More information about this open source framework is available at `http://www.processmining.org`

[2] The number of traces depends on different factors, one being the number of tasks in the model.

the simulated event log, providing an understanding of how much behavior they have in common. To perform this analysis, a probabilistic metric is used.

2.2 Comparing event logs

The main goal of this work is to show that it is possible to perform conformance checking by comparing two event logs: the original event log, and the simulated event log from a mined model. Different metrics can be devised to compare the event logs. The focus here is on the different sequences of events that each event log contains, so the first step is to separate the different sequences in each log and aggregate them in order to determine their frequency of occurrence. Table 1 provides an example.

Original Event Log	Simulated Event Log
A, C, B, D	A, B, C, B, B, D
A, C, B, D	A, C, B, D
A, C, B, D	A, B, D
A, B, C, B, B, D	A, B, C, D
A, B, C, B, B, D	A, B, C, B, B, D
A, B, C, B, B, D	A, B, C, D
A, B, C, B, B, D	A, C, D

Table 1. Example of the Event logs, with the respective sequences.

Looking at the two event logs in table 1 it is possible to say that in the original event log there are only two sequences, with 43% of A, C, B, D and 57% of A, B, C, B, B, D, while in the simulated event log there are four types of sequences, with 29% of A, B, C, B, B, D and A, B, C, D each, and 14% of A, C, B, D and A, B, D and A, C, D each. Both sequences from the original log are present in the simulated log but in addition the simulated log contains other sequences that are not present in the original log. This is the case if the mined model allows for more behavior than what is present in the original log; if, on the other hand, the mined model is *overfitted* to the input event log, then both logs will be more similar.

Given the frequency of occurrence of each sequence, it is possible to use the following metric to compare both logs:

Definition 1. *Let $\mathbb{Z}$ be the set of all sequences recorded in both logs. Let $p(\mathbf{z})$ where $\mathbf{z} \in \mathbb{Z}$ be the function that gives the frequency of occurrence of sequence $\mathbf{z}$ in the original log, and let $q(\mathbf{z})$ be the function that gives the frequency of occurrence of sequence $\mathbf{z}$ in the simulated log. Then the G-metric is defined as:*

$$G(p \parallel q) \triangleq \sum_{\mathbf{z} \in \mathbb{Z}} \sqrt{p(\mathbf{z}) \cdot q(\mathbf{z})}$$

This metric compares the two event logs and outputs a value between 0 and 1, where 1 can only be achieved if the logs are perfectly equal. For the example of table 1 we would have: $G(p \parallel q) \cong \sqrt{0.43 \times 0.14} + \sqrt{0.57 \times 0.29} \cong 0.65$.

This approach is, of course, not without problems, as it assumes that all possible behaviors in the mined model will appear in the simulated log. Take for example the simulation of a loop. Since activity transitions have an associated probability, it is quite unlikely that simulation will be able to reproduce the same sequences as in the original log with the same frequency of occurrence. Therefore, sometimes the simulated log may contain behavior, slightly different, that represent the same behavioral pattern, recorded in the original log. Because this metric only considered the common behavior, shared by the two logs, it won't considered this situation.

Hence the importance of consider all types of behavior by their similarity. Thus, it will be possible to distinguish behavioral sequences,without any kind of relationship, from those that, despite being difference, represent the same pattern behavior.

To overcome this problem it was defined a second metric to be more flexible, that the G-metric. That way, this new metric will compare both logs, by analyzing the similarity between the respective behaviors. To this metric we named it G^*-metric.

The G^*-metric In order to contemplate the similarity between sequences, the G^*-metric makes use of the Levenshtein distance algorithm which measures the amount of difference between two sequences. This technique has been already proposed in process mining research area in [6], in clustering algorithms context.

Thus, the Levenshtein distance will be applied between every sequences of the original and the simulated logs. Based on these values, a similarity matrix will be built, where each line will correspond to a sequence of the original log and each column to a sequence of the simulated log.

From this, it is possible to perform the sequence pairing and apply the G^*-metric, as follows:

Definition 2. *Let L_1 e L_2 be the original and simulated event logs, respectively. Let $\mathbb{Z}$ be the set with the pairs of sequences ($\mathbf{z_1}$,$\mathbf{z_2}$), paired according to the Levenshtein distance, between them. Let $p(\mathbf{z_1})$, where $\mathbf{z_1} \in L_1$, be the function that gives the frequency of occurrence of sequence $\mathbf{z}$ in the original log, and let $q(\mathbf{z2})$, where $\mathbf{z_2} \in L_2$, be the function that gives the frequency of occurrence of sequence $\mathbf{z}$ in the simulated log.Then the G^*-metric is defined as:*

$$G^*(p \parallel q) \triangleq \sum_{\mathbf{z_1} \in \mathbb{Z}} \sum_{\mathbf{z_2} \in \mathbb{Z}} \delta(\mathbf{z_1}, \mathbf{z_2}) \cdot \sqrt{p(\mathbf{z_1}) \cdot q(\mathbf{z_2})}$$

where

$$\delta(\mathbf{z_1}, \mathbf{z_2}) = \begin{cases} 1 \textit{ it they are paired} \\ 0 \textit{ if they are not paired} \end{cases} \tag{1}$$

It is important to mentioned that the G^*-metric was defined in order to be easily extended with different distance algorithms. For that purpose, it is just necessary to redefine the $\delta(\mathbf{z_1}, \mathbf{z_2})$ function.

3 Simulation-based conformance framework

As was introduced, in the beginning of this paper, the main goal of this work is to introduce two different frameworks, to perform process conformance and evaluate process mining algorithms, making use of the metrics logs comparison, introduced before (Section 2.2). Thus, in the Section 3.1 it will be presented the four steps framework used to perform the evaluation of process mining algorithms. Finally, in Section 3.2 a framework to perform process conformance, will be presented.

3.1 Process Mining Evaluation Framework

In this section we describe the structure of a simulation-based conformance framework that provides the means to compare process models via their event logs.

The simulation-based conformance framework main goal is to serve as an evaluation framework for process mining algorithms. The event log that contains the run-time behavior already exists, and a process mining technique (to be evaluated) is applied to that event log in order to extract a mined model. A second event log can be generated from this mined model via simulation, and compared to the first event log in order to determine how good the mined model is as a description of the observed behavior. It may be that another process mining technique, that can also be evaluated in the same way, provides better results. The need for a common framework for the evaluation of process mining techniques had already been stressed by [9].

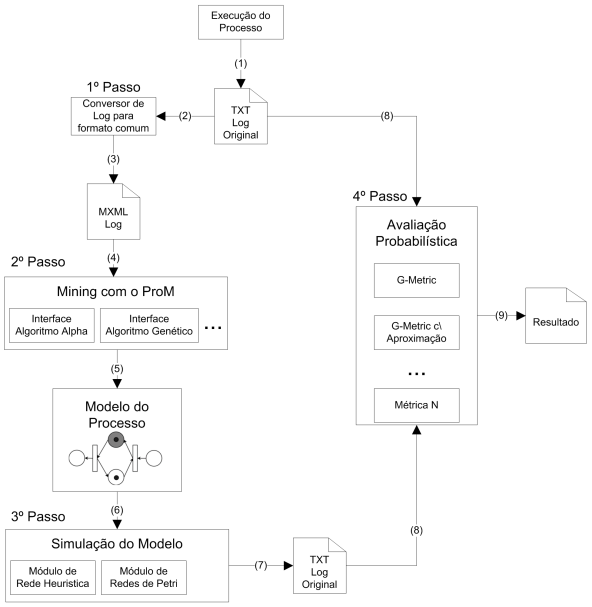

Fig. 1. Main steps in the proposed mining evaluation framework.

Figure 1 presents the structure of the proposed evaluation framework, which comprises four main steps. The framework starts by receiving two text files as input: one contains the framework settings (such as the algorithm to evaluate, the metric to apply, etc.) and another contains a text-based description of the event log under study. Basically, it is a similar format to that of table 1. The **first step** converts this input text file into a regular MXML file that can be used within the ProM framework.

The **second step** mines the MXML event log in ProM, using the selected algorithm. For this purpose, we have developed a set of utility programs to invoke the Java implementation of these plug-ins from the command-line. For each algorithm there is now an interface to invoke the corresponding ProM code, providing a common mechanism to invoke any of the mining algorithms available within ProM. As a result of this second step, the framework obtains a mined model, whose format depends on the mining algorithm being used.

The **third step** is responsible to simulate the mined model, extracted in the second step, and generate a new event log that will be used in the final step. In order to be able to assess a largest number of mining algorithms, it is fundamental to this framework to be able to be model-independent, i.e., it should support any type of process model, as long as it can be simulated. The output of the third step is a log containing the generated sequences.

The framework concludes with a **fourth step** which applies the evaluation metrics to the event logs. In section 2.2 we defined two of such metric, but in practice there may be several metrics to choose from. The higher the metric result, the more similar are the event logs, and hence the more positive is the result of the evaluation.

3.2 Process Conformance Evaluation Framework

In this section we describe the structure of framework that makes use of a set of metrics to compare the event logs, in order to perform process conformance. This framework can be used for conformance checking between a predefined process model and the observed run-time behavior. The event log that contains the run-time behavior already exists, and a second event log can be generated from the predefined model itself. Comparing the two event logs provides an idea of how close or how far the run-time behavior is from the predefined model.

Thus, the defined framework receives, as arguments, a log and a model and returns the degree of behavioral similarity between them. This frameworks is structured over two distinct steps.

Therefore, the **first step** will consist in the standardization of input arguments. While the normalization of the event log is performed in the same way as it was in the process mining evaluation framework, with the model was necessary to define a way to do it.

For such purpose, it was assumed that a model is composed by a finite number of behavioral sequences. It was also assumed that, if is not defined the model's transition probability, in decision points all paths have the same probability of occurrence. Thus, it will be possible to represent the model's behavior using the same format as used in the event logs.

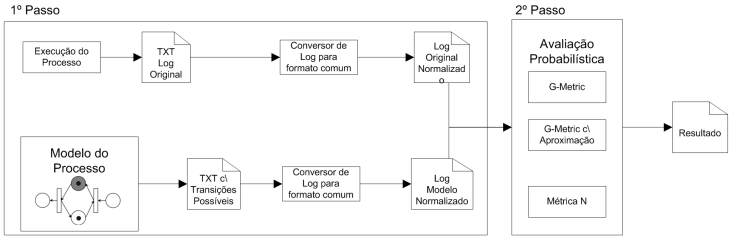

Fig. 2. Main steps in the process conformance evaluation framework.

Then, takes place the **second step** where the evaluation metrics will be applied. This step takes place, exactly, the same way of the fourth step from the process mining framework.

4 Evaluation of process mining algorithms

Currently, the ProM framework includes several process mining techniques with different characteristics. So, for that purpose different deterministic approaches (α[10] and $\alpha++$ [11] algorithms.

Moreover, were also studied three deterministic algorithms, that try to filter the wrong behavior, based on their frequency. Were they: Heuristic Miner [12], DWS [13] and the Genetic Miner [12, 14, 11].

Two out of these five algorithms, namely the α and the α^{++} algorithms, are based on Petri Nets. The remaining three are based on heuristic nets. Therefore, in order to support Petri and Heuristic Nets in the third step of the framework, two simulation modules were developed.

It is also important to draw attention to the fact that, for the purpose of this study, we chose to execute the algorithms with their respective default parameters. The only exception was the GeneticMiner. In this case, performance issues led us to execute the algorithm with a smaller maximum number of generations.

4.1 Workflow patterns

To apply the proposed framework to the evaluation of process mining algorithms, a number of well-known workflow patterns have been used. These patterns have been selected with the aim of providing a representative set of distinct process behaviors. With this we intended on one hand to evaluate the key characteristics of each algorithms and, on the other hand, to compare the results between them in order to confirm some expected results. The studied patterns are:

– Sequence: Simple sequence of activities, where every activity is executed after his previous tasks being complete;
– One-loop: Sequence of activities, where one of them repeats, an indefinite number of times.

– Two-loop: Sequence where a group of two activities repeats, an indefinite number of times.
– Three-loop: Sequence where a group of three activities repeats, an indefinite number of times.
– And Split/Join: Sequence in which two or more activities occur in parallel.
– XOr Split/Join: Sequence in which two or more activities occur in concurrence.
– And XOr: Sequence that merges the sequences *And Split/Join* e *XOr Split/Join*.
– And One-loop: Sequence that merges the sequences *And Split/Join* and *1-loop*.
– And Two-loop: Sequence that merges the sequences *And Split/Join* and *2-loop*.
– And Three-loop: Sequence that merges the sequences *And Split/Join* and *3-loop*.

4.2 Metric results

Table 2 presents the results of applying the proposed conformance framework to evaluate the ability of the process mining algorithms to capture the selected workflow patterns. These values were collected from multiple executions of the conformance framework, i.e., for each pattern a mining algorithm was applied about 100 times, a simulated log being generated each time, from which it was possible to estimate the minimum, maximum, and average values for the G-metric. Since the framework is based on a probabilistic approach, the more we execute the framework the more precise the output will become.

	α			α^{++}			HeuristicsMiner			GeneticMiner			DWS Miner		
	min	max	avg	min	max	avg	min	max	avg	min	max	avg	min	max	avg
Sequence	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
One-loop	0.0	0.0	0.0	0.62	0.67	0.64	0.83	0.95	0.9	0.86	0.96	0.91	0.85	0.95	0.9
Two-loop	0.0	0.0	0.0	0.48	0.54	0.51	0.62	0.78	0.71	0.62	0.8	0.71	0.64	0.79	0.71
Three-loop	0.0	0.0	0.0	0.47	0.54	0.51	0.63	0.8	0.71	0.62	0.82	0.71	0.6	0.78	0.71
And Split/Join	0.98	0.99	0.99	0.98	0.99	0.99	0.39	0.64	0.55	0.43	0.63	0.53	0.43	0.64	0.55
Or Split/Join	0.98	0.99	0.99	0.98	0.99	0.99	0.99	1.0	1.0	0.99	1.0	1.0	0.99	1.0	1.0
And - Or	0.98	0.99	0.99	0.98	0.99	0.99	0.59	0.76	0.7	0.65	0.79	0.72	0.61	0.8	0.7
And One-loop	0.0	0.0	0.0	0.48	0.54	0.51	0.33	0.58	0.43	0.26	0.49	0.38	0.29	0.56	0.45
And Two-loop	0.0	0.0	0.0	0.31	0.33	0.32	0.48	0.67	0.57	0.47	0.71	0.59	0.42	0.66	0.57
And Three-loop	0.0	0.0	0.0	0.32	0.35	0.34	0.54	0.74	0.64	0.2	0.35	0.26	0.55	0.74	0.64

Table 2. Results of applying the process mining evaluation framework to the detection of a number of workflow patterns by different process mining algorithms.

The first pattern – *sequence* – is a trivial pattern that was used just to check that the basics features of the framework were working fine.

As expected, the α-algorithm is not able to identify any of the patterns containing short loops. On the other hand, the α^{++} algorithm, that was conceived to extend the α-algorithm to mine short loops presents more satisfactory results for these patterns. However, the metric values still remain low when compared to other algorithms such as the probabilistic-based HeuristicsMiner, GeneticMiner and the DWS algorithm.

Another important conclusion is that the algorithms that use Petri net models reach more satisfactory results when mining patterns that contain parallel behavior. One explanation for this is in the simulation process (step 3 of the framework). The fact is that Petri nets are structurally more formal and restrictive when compared with the heuristic nets, which in situations like this tend to allow the generation of extra behavior.

Finally, it also stands out that the results from the HeuristicsMiner, the GeneticMiner and the DWS algorithm are very similar. Due to their characteristics, the difference between them should be more obvious when comparing their metric results. In this study the workflow patterns that have been considered are not complex enough to provide a more interesting comparison of these algorithms in particular. Nevertheless, they suffice to demonstrate that the proposed framework can actually be used as a reliable approach to the evaluation of process mining techniques.

5 Case Studies

5.1 Simulating a Messaging Routing and Processing System

This case study is based on a message processing and routing system, extracted from the article [15], where the benefits of meta-models in the simulation of stochastic models. This system has five different routers, that exchange messages with each other, as illustrated in Figure 3.

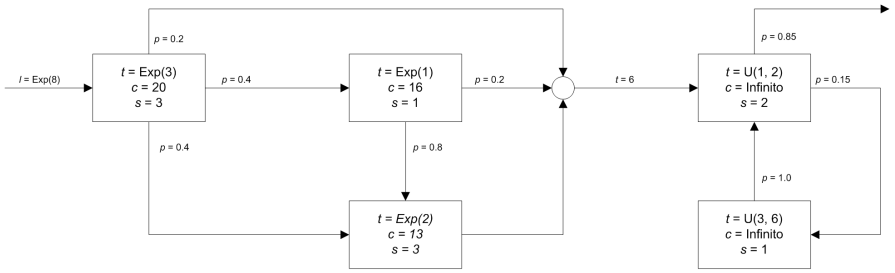

Fig. 3. Messaging routing and processing system: c = queue capacity, s = number of servers, t = processing time, l = time between arrivals, and p = path probability [15].

Each router has his own storage capacity of messages, in queue, while they wait to be processed and sent. When the queue capacity is reached, the new messages that arrive start to be discarded[3]. The decision of the messages routing is defined by a probabilistic function, which varies for each router.

Therefore, the main goal of this case study was to study routing path, of different messages, and try to discover the main patterns, by using process mining algorithms. Finally, making use of the process mining evaluation framework, presented in this work, the results produce by the mining algorithms will be evaluated and compared with the inicial model.

For that purpose, the the system was simulated, using the Awesim software [16]. The simulation produced a log of all the different states that the messages had, throughout its route. This step was repeated six times, with different overload parameters, i.e., the regularity with which the messages were introduced into the network, thus influencing a greater or lesser traffic on the network.

Therefore, therefore six logs were created with 10.000 entries, each. For each log, the evaluation framework was applied. The results of each simulation are in the Table 3. Analyzing this results, it is possible to see that, in all of the six experiments, the five algorithms were extremely efficient, being the results between 85% and 100%. In order to confirm the validity of the results, there was a comparison between the models, extracted by the process mining algorithms, and the model resulting from the Figure 3.

Although all the algorithms had manage to extract the correct behavior of the message routing, the deterministic algorithms, that use Petri nets, had worst results when compared with the probabilistic algorithms. This happen mainly because of the simulation process (third step of the evaluation framework). Unlike with the probabilistic algorithms, the deterministic algorithms fail to recreate in the process of simulation, the sequences in suitable proportions contributing to worse results, even when they extract correctly the behavior recorded in the log. Hence, despite the results obtained by the algorithms α and α^{++} be satisfactory, they are always below the other algorithms.

Another important conclusion was that, for different experiments with different message overload, the results were always the same.

[3] When a message is discarded it means that its sequence of events will stay incomplete, creating incorrect behavior

	α			α^{++}			Heuristic Miner			Genetic Miner			DWS Miner		
	min	**avg**	max	min	**avg**	max	min	**avg**	max	min	**avg**	max	min	**avg**	max
Log 1	0.85	**0.86**	0.88	0.85	**0.87**	0.88	0.99	**1.0**	1.0	0.99	**1.0**	1.0	0.99	**1.0**	1.0
Log 2	0.85	**0.86**	0.88	0.84	**0.86**	0.89	0.99	**1.0**	1.0	0.99	**1.0**	1.0	0.99	**1.0**	1.0
Log 3	0.85	**0.87**	0.89	0.85	**0.87**	0.89	0.99	**1.0**	1.0	0.99	**1.0**	1.0	0.99	**1.0**	1.0
Log 4	0.84	**0.86**	0.88	0.85	**0.87**	0.88	0.99	**1.0**	1.0	0.99	**1.0**	1.0	0.99	**1.0**	1.0
Log 5	0.85	**0.87**	0.89	0.84	**0.87**	0.89	0.99	**1.0**	1.0	0.99	**1.0**	1.0	0.99	**1.0**	1.0
Log 6	0.85	**0.87**	0.89	0.85	**0.87**	0.89	0.99	**1.0**	1.0	0.99	**1.0**	1.0	0.99	**1.0**	1.0

Table 3. Resultados obtidos aps a aplicao da metodologia de avaliao recorrendo mtrica *naive*

5.2 Issue Manager Process

This case study is based on a case presented in the article [17], where it was studied a medium-sized IT company whose the main product is an advanced software platform to facilitate and accelerate the development of custom business applications, while reducing their operating costs.

To contribute in the improvement of this platform, a software was developed to register new issues reported by the support team – called Issue Manager. This software allows to the support team to register information regarding each issue (such as date, description, submitter, status, priority, risk, severity, etc.). These data can be filled with whatever the support team finds appropriate, except for the status field which is allowed to have one of a limited set of possible states.

Therefore, during handling, the issue goes through a number of different states. Some of these states may actually be skipped for issues that can be solved immediately, while other issues may get to the point of generating a request for change, which will then trigger a separate development process. This scenario is illustrated in Figure 4.

The goal of this study is to study the behavior recorded, by the Issue Manager, and see if it comply with the predefined behavior. For that it was used the process conformance evaluation framework, presented in this work.

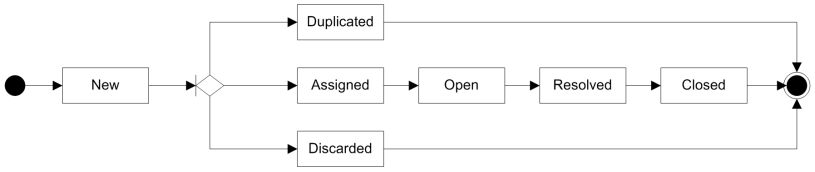

Fig. 4. Descrio do Processo pr-definido de tratamento de Incidentes [17]

In order to study how well the issues, reported by the support team, were being handling, the information about each one status were extracted from the Issue Manager's data base. From that was possible to infer that, beyond the default behavior, new patterns have emerged (see Table 4).

Thus, applying the process conformance evaluation framework it is possible to calculate if the two behaviors conform, applying the G-metric.

$$G(p \parallel q) \triangleq \sum_{\mathbf{z} \in \mathbb{Z}} \sqrt{p(\mathbf{z}) \cdot q(\mathbf{z})} \quad \approx 0,65$$

Although the result may be considered positive, it proves that significant changes had occur in the registration process and, as such, it should be reviewed. In this case the G^*-metric would produce similar results, because all the predefined behaviors were correctly recorded.

However, consider the hypothetical scenario where both sequences "New, Assigned, Open, Resolved, Closed" and "New, Duplicated" had not been recorded. In this scenario the G-metric would only give $0,23$, because there were only one common behavior sequence ("New,Discarded"), even considering

Predefined Behavior		Recorded Behavior	
New,Discarded	1	New,Open,Resolved,Closed	1145
New,Assigned,Open,Resolved,Closed	1	New, Assigned, Open, Resolved, Closed	1129
		New, Duplicated	1043
		New,Discarded	733
		New,Approved	626
		Assigned,Closed	554
		New,Assigned,Resolved,Closed	495
		New,Assigned	455
		New,Resolved,Closed	313
		New,Approved,Duplicated	273

Table 4. List of the predefined behavioral sequences and the most common sequences recorded by the Issue Manager.

that there was a strong link behavioral. It is in this situations, where the G^*-metric can be extremely helpful, being more flexible. Thus, the applying the G^*-metric the result would be:

$$G(p \parallel q) \triangleq \sum_{\mathbf{z_1} \in \mathbb{Z}} \sum_{\mathbf{z_2} \in \mathbb{Z}} \delta(\mathbf{z_1}, \mathbf{z_2}) \cdot \sqrt{p(\mathbf{z_1}) \cdot q(\mathbf{z_2})} \quad \approx 0,73$$

From this case study, it is possible to take two important conclusions. First that, sometimes the it is important to use more flexible metrics, when compare only the common behavior between the two sources, is not enough. Sometimes it is important to analyze the similar behavior, too.

However, the G^*-metric should be used as a complement to the G-metric, when it produce reduced results.

6 Conclusions and Future Work

In this paper we have presented a framework for process conformance that can also be used as a tool to evaluate different process mining algorithms. The framework is based on model simulation and on the direct comparison of event logs via specific metrics. In general, there will be one event log obtained from the run-time behavior recorded in an information system, and another event log generated by simulation from a predefined process model. Conformance can be checked by comparing both event logs. If, however, the second event log is obtained by simulating mined models, then the proposed approach can be used as an evaluation framework for mining algorithms.

Experiments with a set of common workflow patterns suggest that this approach is valid for both purposes. Nonetheless, many improvements are still possible. As such, our future work includes: evaluating a set of more flexible metrics; developing a new model for simulation.

References

1. van der Aalst, W.M.P., van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G., Weijters, A.J.M.M.: Workflow mining: A survey of issues and approaches. Data Knowl. Eng. **47**(2) (2003) 237–267
2. Rozinat, A.: Conformance testing: Measuring the alignment between event logs and process models. Master's thesis, University of Technology, Eindhoven, Netherlands (2005)
3. Rozinat, A., van der Aalst, W.M.P.: Conformance checking of processes based on monitoring real behavior. Inf. Syst. **33**(1) (2008) 64–95
4. Governatori, G., Milosevic, Z., Sadiq, S.: Compliance checking between business processes and business contracts. In: EDOC '06: Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference, Washington, DC, USA, IEEE Computer Society (2006) 221–232
5. van Dongen, B.F., Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.: The ProM framework: A new era in process mining tool support. In Ciardo, G., Darondeau, P., eds.: Application and Theory of Petri Nets 2005. Volume 3536 of Lecture Notes in Computer Science., Springer-Verlag, Berlin (2005) 444–454

6. Rozinat, A., Mans, R.S., Song, M., van der Aalst, W.M.P.: Discovering simulation models. Inf. Syst. **34**(3) (2009) 305–327
7. Medeiros, A.K.A., Günther, C.W.: Process mining: Using CPN tools to create test logs for mining algorithms. In Jensen, K., ed.: Proceedings of the Sixth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools. Volume 576 of DAIMI., University of Aarhus (2005) 177–190
8. Günther, C.W., van der Aalst, W.M.P.: A generic import framework for process event logs. In: Business Process Management Workshops. Volume Volume 4103/2006 of Business Process Management Workshops. (2006) 81–92
9. Rozinat, A., Medeiros, A.K.A., Günther, C.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The need for a process mining evaluation framework in research and practice. In: Business Process Management Workshops. Volume Volume 4928/2008 of Lecture Notes in Computer Science., Springer Berlin / Heidelberg (2007) 84–89
10. Medeiros, A.K.A., van Dongen, B.F., van der Aalst, W.M.P., Weijters, A.J.M.M.: Process mining: Extending the α-algorithm to mine short loops. BETA Working Paper Series WP 113, Eindhoven University of Technology (2004)
11. Medeiros, A.K.A.: Genetic Process Mining. PhD thesis, Eindhoven University of Technology, Netherlands (2006)
12. Weijters, A.J.M.M., van der Aalst, W.M.P., Medeiros, A.K.A.: Process mining with the HeuristicsMiner algorithm. BETA Working Paper Series WP 166, Eindhoven University of Technology, Eindhoven (2006)
13. Medeiros, A.K.A., Guzzo, A., Greco, G., van der Aalst, W.M.P., Weijters, A.J.M.M., van Dongen, B.F., Saccà, D.: Process mining based on clustering: A quest for precision. In ter Hofstede, A.H.M., Benatallah, B., Paik, H.Y., eds.: Business Process Management Workshops. Volume 4928 of Lecture Notes in Computer Science., Springer (2007) 17–29
14. Mitchell, T.M.: Machine learning. McGraw-Hill (1997)
15. dos Santos, P.M.R., dos Santos, M.I.R.: Using subsystem linear regression metamodels in stochastic simulation. European Journal of Operational Research **vol. 196, n. 3** (August 2009) pp. 1031–1040
16. Pritsker, A.A., O'Reilly, J.J.: Simulation with Visual Slam and Awesim. John Wiley & Sons, Inc., New York, NY, USA (1999)
17. Ferreira, D.R., da Silva, M.M.: Using process mining for itil assessment: a case study with incident management. In: Proceedings of the 13th Annual UKAIS Conference, Bournemouth University (April 10-11 2008)