A Probabilistic Unified Framework for Event Abstraction and Process Detection from Log Data

Bettina Fazzinga[2(✉)], Sergio Flesca[1], Filippo Furfaro[1], Elio Masciari[2], and Luigi Pontieri[2]

[1] DIMES, University of Calabria, Rende, Italy
{flesca,furfaro}@dimes.unical.it
[2] ICAR-CNR, Rende, Italy
{fazzinga,masciari,pontieri}@icar.cnr.it

Abstract. We consider the scenario where the executions of different business processes are traced into a log, where each trace describes a process instance as a sequence of low-level events (representing basic kinds of operations). In this context, we address a novel problem: given a description of the processes' behaviors in terms of high-level activities (instead of low-level events), and in the presence of uncertainty in the mapping between events and activities, find all the interpretations of each trace Φ. Specifically, an interpretation is a pair $\langle \sigma, W \rangle$ that provides a two-level "explanation" for Φ: σ is a sequence of activities that may have triggered the events in Φ, and W is a process whose model admits σ. To solve this problem, we propose a probabilistic framework representing "consistent" Φ's interpretations, where each interpretation is associated with a probability score.

1 Introduction

Thanks to the diffusion of various automated process management and tracing platforms, many process logs (i.e., collections of execution traces) have become available. Log data can be exploited to analyze and improve the processes, by possibly using process mining techniques [2], such as those for inducing a process model [1], for checking whether log traces comply to a model [3], and for quantifying "how much" a log and a model conform one to the other [7].

All of these techniques, however, require log events to be univocally mapped to activity concepts corresponding to some high-level view of the process, suitable for the analysis. For example, in order to evaluate the compliance (or the conformance) of a log w.r.t. a given process model, it is necessary that each event in the log refers to one of the activities that appear in the model. Unfortunately, this assumption does not hold often in practice. As a matter of fact, in logs of many real systems, the recorded events just represent low-level operations, with no clear reference to the business activities that were carried out through these operations, as shown in the following example.

C. Debruyne et al. (Eds.): OTM 2015 Conferences, LNCS 9415, pp. 320–328, 2015.
DOI: 10.1007/978-3-319-26148-5_20

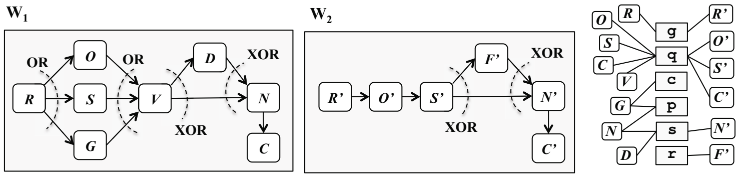

Fig. 1. Two process models (left and middle), and all the possible mappings between their activities and the low-level event types, shown inside dashed squares, that occur in the log (right).

Example 1. Consider the case of a phone company, where two business processes are carried out: 1) service activation, and 2) issue management. Assume that an abstract description of the behaviors of the two processes above is available, encoded by two models, named W_1 and W_2 and sketched in Fig. 1 in the form of *precedence graphs*. Therein, nodes and edges (some of which are labeled with typical split/join constraints) represent activities and precedence links, respectively. Basically, these models describe the corresponding processes in terms of high-level activities. In particular, W_1 (service activation) features the set of activities $\mathcal{A}_{W_1} = \{$ R (*Receive a service request*), O (*Open a new case*), S (*Select a service package*), G (*Gather more information on the customer*), V (*Validate the package*), D (*Dispatch a contract proposal*), N (*Notify the request's outcome*), C (*Close the case*)$\}$. Analogously, W_2 (issue notification) features $\mathcal{A}_{W_2} = \{$ R' (*Receive a request of help*), O' (*Open a ticket*), S' (*Search the issue in a knowledge base*), F' (*Fix the issue*), N' (*Notify the solution*), C' (*Close the ticket*)$\}$[1].

All these activities are performed via low-level operations (provided by different collaborative-work tools, including databases and communication services) that correspond to instances of the following event types: `g` (resp., `s`, `c`, `p`): *getting an email from* (resp., *sending an email to*, *opening a chat session with*, *making a phone call to*) *a customer*; `r`: *delivering a report*; `q`: *posing a query against a database*. Thus, performing any activity results in the execution of one of the above events, and the correspondence between event and activity types is *many to many*, as depicted on the right-hand side of Fig. 1. For instance, activity G (*Gather more information on the customer*) can be accomplished by either *opening a chat session with the customer* (event `c`) or *making a phone call to the customer* (event `p`). Analogously, an instance of event `p` can be generated by the execution of either activity G or N.

The enactments of the processes is monitored by a tracing system that stores the executions of the low-level operations triggered during the process execution. Hence, an instance of process W_1 consisting in the execution of the sequence $R\,S\,G\,V\,N\,C$ can be stored in the log as the (low-level event) *trace* `g q p q s q`, or as the trace `g q c q p q`, depending on the low-level operations triggered by each activity instance. □

[1] Two disjoint sets of activities are here shown for the two models only for the sake of readability. In fact, our approach can deal with activities shared by an arbitrary number of process models.

In such a setting, we address a novel challenging analysis problem: given a log L (possibly containing traces generated by different processes), and a set $\mathcal{W}$ of candidate process models (encoded each as a set of behavioral constraints), we want to assess whether each trace in L can be regarded as an instance of each model in $\mathcal{W}$, and to what degree of confidence. In order to support such analysis, we need to *interpret* each step of the trace (i.e. each low-level event registered in the trace) as the execution of an activity from one of the process models, so that the whole trace can be eventually viewed as a instance of that model.

Example 2 (contd.). Consider the trace Φ= `gqqssq`. In order to interpret Φ as the result of an execution of the models in Fig. 1, we should first translate Φ into a sequence of activities σ (by interpreting each of its events as the execution of a single activity), and then interpreting σ as an instance of W_1 or W_2. Actually, as each event can have different causes (in terms of activities that have generated it), Φ may admit multiple sequences of activities "explaining" it. In turn, the same sequence of activities may be compatible with different process models. Hence, there can be multiple reasonable interpretations of Φ as a pair $\langle\sigma, W\rangle$ (where σ is a sequence of activities and W a process): those where the mappings event/activity used for obtaining σ agree with the right-hand side of Fig. 1, and σ complies with the model of W.

It is easy to see that, based on this reasoning, the only valid interpretations for Φ consist in viewing it as generated by either the sequence $ROVDNC$ or $RSVDNC$, and viewing both these sequences as generated by an instance of process W_1. In fact, any other translation of Φ into a sequence of activities consistent with the event-activity mappings in Fig. 1 does not conform to any process model. □

Contribution. We propose a probabilistic framework for facing the interpretation problem above in a uniform synergistic way, at both mapping levels (i.e. events vs. activities, and traces vs. process models). As a matter of fact, current log abstraction techniques [4–6] do not approach this problem adequately, as they do not address the specific "interpretation" problem faced in this paper.

The ultimate result of our approach is a "conditioned interpretation set" (named *ci-set*) for each input trace Φ, which stores, for each candidate model W, the probability that Φ is an instance of W, along with probabilistically-scored mappings between the steps of Φ and instances of the activities in W. Technically, we first introduce a model for formally describing the structure of each process (via precedence constraints), and for probabilistically ranking the event-activity and trace-process mappings (via a-priori probabilities). Next, based on this model, we describe the conditioning approach for encoding all and only the interpretations of the input trace Φ, which are consistent with the structure of the processes, and the probabilities assigned to the mappings from each step of Φ to an activity, and from the whole Φ to a process. Preliminary experiments on a real-life scenario, prove that the approach is effective yet not efficient enough.

2 Preliminaries

Logs, Traces, Processes, Activities and Events. A log is a set of *traces*. Each trace Φ describes a process instance at the abstraction level of basic *events*, each generated by the execution of an activity. That is, an instance w of a *process* W is the execution of a sequence $A_1, \ldots, A_m$ of *activities*; in turn, the execution of each activity A_i generates an instance e_i of an event E_i; hence, the trace describing w consists in the sequence $e_1, \ldots, e_m$ of event instances. For any event e_i occurring in a trace, we assume that the starting time point of its execution is represented in the log, and denote it as $e_i.t_s$.

In the following, we assume given the sets $\mathcal{W}$, $\mathcal{A}$, $\mathcal{E}$ of (types of) processes, activities, and events, respectively, and denote their elements with upper-case alphabetical symbols (such as W, A, E). Process-, activity-, and event- instances will be denoted with lower-case symbols (such as w, a, e).

Given an event instance e, we denote the event of which it is an instance as $E\text{-}type(e)$.

Composition Rules for the Processes (Process Models). We assume that every process $W \in \mathcal{W}$ is "regulated" by a "*composition rule*", that restricts the sequences of activities that are allowed to be executed when W is enacted. Basically, the composition rule associated with W is a pair $\langle ActSet, \mathcal{IC}\rangle$ (whose members will be also denoted as $W.ActSet$ and $W.\mathcal{IC}$, respectively), where *Act-Set* is the set of activities that are allowed to be executed within any instance of W, and $\mathcal{IC}$ is a set of constraints of the form $A \Rightarrow_T B$ (called *must-constraint*) or $A \Rightarrow_T \neg B$ (called *not-constraint*), where A, $B \in \mathcal{A}$ and T is of the form '$\leq c$', where c is a constant. Basically, $A \Rightarrow_T B$ (resp., $A \Rightarrow_T \neg B$) imposes that, within every instance of W, the beginning of an instance a of A always (resp., never) precedes the beginning of an instance b of B such that the width of the interval between the starting times of a and b satisfies T. Omitting T is equivalent to specifying $T =$'$\leq \infty$'. The set of composition rules associated with the processes in $\mathcal{W}$ will be denoted as $\mathcal{CR}$.

Correspondence Between Events and Activities. Under the above-discussed assumption that the execution of an activity generates one event instance, we consider the general case that the correspondence between activities and event types is *many to many*: different instances of the same event can be generated by executions of different activities, and vice versa. For instance, with reference to Example 1 (see Fig.1), activity G (*Gather more information on the customer*) can generate an instance of either event $\mathtt{c}$ (*opening a chat session...*) or of event $\mathtt{p}$ (*making a phone call...*), while an instance of event $\mathtt{p}$ can be generated by both the activities G and N (*Notify the request's outcome*).

Given an event $E \in \mathcal{E}$, we denote as *cand-act*(E) the set of activities whose execution is known to possibly generate an instance of E. In turn, given an instance e of E, we use *cand-act*(e) as a shorthand for *cand-act*$(E\text{-}type(e))$: basically, *cand-act*(e) contains every "candidate" activity A such that e can be viewed as the result of executing A.

3 The Interpretation Problem and Our Approach for Solving It

Problem Statement. The problem addressed in this paper is that of "interpreting" an input trace $\Phi = e_1 \dots e_m$ from an event log. This means deciding, for each e_i of Φ, the activity $A_i \in \textit{cand-act}(e_i)$ whose execution generated e_i, and, in turn, the process W whose execution caused the sequence of activities $A_1 \dots A_m$ to be performed. When deciding this, the models of the processes encoded by the composition rules in $\mathcal{CR}$ must be taken into account.

More formally, the solution of an instance $\Phi = e_1 \dots e_m$ of this problem is called *interpretation for Φ consistent with $\mathcal{CR}$* (or, simply, *consistent interpretation*, when Φ and $\mathcal{CR}$ are understood), and is a pair $\langle \sigma, W \rangle$, where:

– σ is called *sequence-interpretation* and is a sequence $A_1 \dots A_m$ of activities, where each A_j is in $\textit{cand-act}(e_i)$, meaning that each e_j is interpreted as the result of executing an instance of the activity A_j;
– W is called *process-interpretation*, meaning that Φ is interpreted as the result of an execution of the process W;
– σ conforms to the composition rule of W.

Example 3. Given $\mathcal{W} = \{W_1, W_2\}$, $\mathcal{A} = \{A, B, C\}$, $\mathcal{E} = \{E_1, E_2, E_3\}$, let $\langle \mathcal{A}, \{A \Rightarrow B; A \Rightarrow C\} \rangle$ be the composition rule of W_1, and $\langle \mathcal{A}, \{B \Rightarrow A; B \Rightarrow \neg C\} \rangle$ the composition rule of W_2. Moreover, assume that $\textit{cand-act}(E_1) = \{A\}$, $\textit{cand-act}(E_2) = \{B\}$, $\textit{cand-act}(E_3) = \{A, B, C\}$. Consider the trace $\Phi = e_1\, e_2\, e_3$, where $\textit{E-type}(e_1) = E_1$, $\textit{E-type}(e_2) = E_2$, $\textit{E-type}(e_3) = E_3$. If we consider only the correspondence between events and activities encoded by $\textit{cand-act}(\cdot)$, Φ can be interpreted as the result of executing one of the following sequences of activities: $\sigma_1 = A\,B\,A$, or $\sigma_2 = A\,B\,B$, or $\sigma_3 = A\,B\,C$. It is easy to see that $\sigma_1 = A\,B\,A$ is inconsistent with the composition rule of W_1, but consistent with that of W_2; $\sigma_2 = A\,B\,B$ is inconsistent with both the composition rules of W_1 and W_2; finally, $\sigma_1 = A\,B\,C$ is consistent with the composition rule of W_1, but inconsistent with that of W_2. Hence, there are two consistent interpretations for Φ: $\langle A\,B\,C, W_1 \rangle$ and $\langle A\,B\,A, W_2 \rangle$. □

Dealing with the Interpretation Problem in Probabilistic Terms. The above example describes an instance of the interpretation problem admitting multiple solutions. This situation is likely to happen frequently, for the following reasons: 1) the correspondence between events and activities is many to many, thus an event instance can be interpreted as the result of different activity executions; 2) the description of the process models provided by composition rules can be rather "loose", thus there can be process instances consistent with the composition rules of different processes: that is, the execution of a sequence of activities can be interpreted as the execution of different processes.

The presence of this uncertainty suggests to address the interpretation problem in probabilistic terms. Under the probabilistic perspective, the problem becomes that of finding all the consistent interpretations, and associating each of them with the probability of being the actual one. A starting point for probabilistically addressing the interpretation problem could be that of modeling the correspondence between events and activities probabilistically. In this direction, we assume that the many-to-many correspondence between events and activities is modeled by a probability distribution function (pdf) $p^a(A|E)$ returning the *a-priori* probability that an instance of event E is generated by an execution of A. For instance, $p^a(A_1|E_1) = 0.75$ and $p^a(A_2|E_1) = 0.25$ means that any instance of event E_1 is generated by the execution of either A_1 or A_2, and that the former case is three times more probable than the latter. This pdf is said to be *a-priori* since it assigns probabilities to the different ways of interpreting an event in terms of an activity execution, without looking at the context where the event instance happened. That is, it provides an interpretation for the events that occur in a trace without looking at the other events occurring in the trace.

Analogously, we assume given a pdf $p^a(W)$ over $\mathcal{W}$, assigning to each $W \in \mathcal{W}$ the *a-priori* probability that a trace in a log encodes an instance of W. For instance, given $\mathcal{W} = \{W_1, W_2\}$, $p^a(W_1) = p^a(W_2) = 0.5$ means that, in the absence of further knowledge, W_1 and W_2 are equi-probable interpretations of any trace.

It is worth noting that assuming that $p^a(A|E)$ and $p^a(W)$ are known is realistic: as done in our experiments, these pdfs can be obtained by computing statistics taken from historical data, or by encoding domain-expert knowledge.

We now show how the pdfs $p^a(A|E)$ and $p^a(W)$ can be exploited as a basis for probabilistically addressing the interpretation problem. We start by introducing a naive approach (that will be used in the experiments as term of comparison) and discuss its limits; then, we briefly introduce the rationale of our framework, explaining how it overcomes these limits.

Once the a-priori pdfs $p^a(A|E)$ and $p^a(W)$ are given, a naive way to solve the interpretation problem is the following: First, assume that the event instances in Φ are independent from one another; then, use the a-priori pdf $p^a(A|E)$ to probabilistically interpret each e_i in $\Phi = e_1 \dots e_m$ as an instance of an activity A_i, and the pdf $p^a(W)$ to probabilistically interpret the whole Φ as an instance of one of the processes, say W_j. As implied by the independence assumption, every sequence-interpretation $\sigma = A_1 \dots A_m$ obtained this way will be associated with the probability $\Pi_{i=1}^{m} p^a(A_i|E\text{-}type(e_i))$ (that is the product of the a-priori probabilities of the event-to-activity mappings at each step), while the process-interpretation W_j will have probability $p^a(W_j)$. Unfortunately, this naive approach is unlikely to provide reasonable results, as explained in the following example.

Example 4. Consider the scenario where the sets of processes, activities, and events are $\mathcal{W} = \{W_1, W_2\}$, $\mathcal{A} = \{A, B, C, D\}$, and $\mathcal{E} = \{E_1, E_2\}$, such that $cand\text{-}act(E_1) = \{A, B\}$ and $cand\text{-}act(E_2) = \{C, D\}$. Moreover, assume that the a-priori probabilities of the processes are $p^a(W_1) = 0.7$ and $p^a(W_2) = 0.3$, while those of the activities are $p^a(A|E_1) = p^a(B|E_1) = 0.5$; $p^a(C|E_2) = 0.8$; $p^a(D|E_2) = 0.2$.

Let $\Phi = e_1 e_2$ be the trace to be interpreted, where e_1 and e_2 are instances of E_1 and E_2, respectively. According to the a-priori pdf, if we consider the events in Φ separately, all the sequence-interpretations are: $\sigma_1 = AC$; $\sigma_2 = AD$; $\sigma_3 = BC$; $\sigma_4 = BD$.

Each sequence-interpretation is also associated with a probability, implied by p^a and the independence assumption. Thus, the probability of each sequence-interpretation σ, denoted as $p^a(\sigma)$, is the product of the a-priori probabilities of its steps. Hence:

$$p^a(\sigma_1) = p^a(A|E_1) \cdot p^a(C|E_2) = 0.4; \quad p^a(\sigma_2) = p^a(\sigma_4) = 0.1; \quad p^a(\sigma_3) = 0.4.$$

Moreover, owing to $p^a(W_1)$ and $p^a(W_2)$, we can say that Φ encodes an execution of W_1 with probability 0.7, and of W_2 with probability 0.3.

Now, assume given the context-specific knowledge that the composition rules are: $\langle \mathcal{A}, \{A \Rightarrow C; B \Rightarrow A\}\rangle$ for W_1, and $\langle \mathcal{A}, \{A \Rightarrow B; B \Rightarrow \neg C; B \Rightarrow \neg D\}\rangle$ for W_2. Given this, we have that σ_2, σ_3 and σ_4 are inconsistent with the composition rules of W_1 and W_2, and thus they turn out to be invalid sequence-interpretations. Therefore, given that σ_1 conforms to the composition of W_1 but not of W_2, σ_1 remains the only valid sequence-interpretation, and W_1 turns out to be the only valid process-interpretation. Hence, the a-priori probabilities $p^a(\sigma_1) = 0.4$ and $p^a(W_1) = 0.7$ are not reasonable, and should be revised as the sum of the probabilities of the consistent sequence-interpretations should be 1, and the same should hold for the consistent process-interpretations. □

Our approach consists in using this naive approach to obtain some initial interpretations, and then revising them by *a-posteriori* enforcing the composition rules. In order to do this, we resort to probabilistic conditioning, that is a well-known paradigm used in the general context of forcing integrity constraints over probabilistic databases where the assumption of independence between tuples is originally used. Generally speaking, applying probabilistic conditioning for a random variable X in the presence of a set C of constraints means revising $f(X)$ into $f(X|C)$, where f is the pdf of X: that is, re-evaluating $f(X)$ conditioned to the fact that the constraints in C are satisfied. In our case, this means that we will revise the a-priori pdfs $p^a(\sigma)$ and $p^a(W)$ over the sequence- and the process-interpretations into $p^a(\sigma|\mathcal{CR})$ and $p^a(W|\mathcal{CR})$, respectively.

For instance, consider the case of Example 4. If we disregard the composition rules, the probabilities of the sequence-interpretations are the a-priori ones: $p^a(\sigma_1) = p^a(\sigma_3) = 0.4$; $p^a(\sigma_2) = p^a(\sigma_4) = 0.1$. Now, we have seen that if we take into account the set $\mathcal{CR}$ of composition rules, σ_2, σ_3 and σ_4 turn out to be inadmissible. Thus, the conditioned probabilities are:

$$p^a(\sigma_1|\mathcal{CR}) = \frac{p^a(\sigma_1)}{p^a(\sigma_1)} = 1; \qquad p^a(\sigma_2|\mathcal{CR}) = p^a(\sigma_3|\mathcal{CR}) = p^a(\sigma_4|\mathcal{CR}) = 0.$$

Thus generally, the conditioned probability of a sequence-interpretation is either 0, if it is invalid, or the ratio between its a-priori probability and the sum of the a-priori probabilities of all the valid sequence-interpretations, otherwise.

In turn, the conditioned probabilities of the process interpretations are: $p^a(W_1|\mathcal{CR}) = \frac{p^a(W_1)}{p^a(W_1)} = 1$ and $p^a(W_2|\mathcal{CR}) = 0$. This corresponds to the fact that the only valid process-interpretation of Φ is W_1.

Our approach for interpreting an input trace Φ consists in revising $p^a(\sigma)$ and $p^a(W)$ into $p^a(\sigma|\mathcal{CR})$ and $p^a(W|\mathcal{CR})$ by first enumerating all the interpretations of Φ, then discarding those not satisfying the composition rules, and finally revising the a-priori probabilities of the remaining ones. More in detail, given a trace Φ a probabilistic interpretation of Φ is the set of triples $CIS_\Phi = \{\langle\sigma, W, p^a(\sigma \wedge W|\mathcal{CR})\rangle | \langle\sigma, W\rangle$ is an interpretation for $\Phi\}$, which is named *ci-set* for Φ in the following. It is worth pointing out that CIS_Φ implicitly represent $p^a(\sigma|\mathcal{CR})$ (resp. $p^a(W|\mathcal{CR})$) as $p^a(\sigma|\mathcal{CR}) = \sum_{\langle\sigma,W',p\rangle\in CIS_\Phi} p$ (resp. $p^a(W|\mathcal{CR}) = \sum_{\langle\sigma',W,p\rangle\in CIS_\Phi} p$).

4 Experiments

Scenario, Data, and a-priori pdfs. We tested our framework over real-like data of the administrative units of a phone company (PC). In these scenario, a process instance is a collection of activities performed by the staff of the units in response to customers' requests. We were given the scheme of the processes in terms of precedence relationships (between activities), that were easily encoded into composition rules of the form used in our framework. Moreover, we were given the sets $\mathcal{W}$, $\mathcal{A}$ and $\mathcal{E}$, and, for each $E \in \mathcal{E}$, the set $cand\text{-}act(E)$ of the activities whose execution is known to possibly generate an instance of E. Besides the composition rules the phone company gave us a set of real traces along with suitable values for the a-priori pdfs, based on their knowledge of the domain. Specifically, we were given by the phone company and the service agency 100 real traces describing different process instances. Moreover, for each original trace Φ, we were given also its correct interpretation $\langle\sigma_\Phi, W_\Phi\rangle$.

Efficiency and Effectiveness of the Conditioning Approach. In Table 1 we report the average running times of the generation of the *ci-set* (row *Times*) vs. trace length (we did not run the generation of the *ci-set* on traces longer than 30 as it would require too much time). It is easy to see that the generation of the *ci-set* is feasible only on short traces (i.e., traces of length 10); more efficient approaches should be investigated for dealing with longer traces (i.e., traces of length greater than or equal to 20).

Table 1. Running time of the generation of the *ci-set* and accuracy of activity queries.

Trace length	10	20	30
Running Times	$\approx$ 21 *msec*	$\approx$ 24 *min*	halted after 30 *min*
Accuracy	73%	72%	75%

We now analyze the accuracy of the interpretations. We measured the accuracy with which a *activity* queries can be answered over it. In the following, we denote as Φ the trace over which the queries are posed, and as $\langle\sigma_\Phi, W_\Phi\rangle)$ the corresponding correct interpretation. Given this, an *activity* query (over a step i) asks for the activity that has been performed at the i-th step of Φ, thus its answer is $\sigma_{\Phi}[i]$. The probabilistic answers to activity queries over *ci-sets* are the natural probabilistic extensions of their deterministic semantics. Specifically, given a *ci-set* CIS_{Φ} over Φ, the answer $q(CIS_{\Phi})$ of an activity query q over step i is a set of activities, where each activity A is associated with the probability encoded in CIS_{Φ} that the event e_i of Φ was generated by the execution of an instance of A. Correspondingly, given an activity query q and a *ci-set* CIS_{Φ} over Φ, the accuracy of $q(CIS_{\Phi})$ is measured as the probability associated with A^* in $q(CIS_{\Phi})$, where A^* is the answer of q evaluated on σ_{Φ}. The average accuracies of the answers of activity queries, issued over all the steps of the input traces, are reported in the row *Accuracy* of Table 1. The results show that, in terms of accuracy, our approach is rather good and insensitive to the trace length.

5 Discussion, Conclusion, and Future Work

We have presented a novel probabilistic approach to the problem of interpreting a low-level log trace as an instance of one or more given process models, which relies on computing, for each model, different probability-aware event-activity mappings, which all comply with the behavioral constraints encoded in the model. The proposed model is shown to be effective, but it is not efficient enough to be used in practice. Efficiency issues will be addressed in future work.

References

1. Van der Aalst, W., Weijters, T., Maruster, L.: Workflow mining: Discovering process models from event logs. IEEE TKDE **16**(9), 1128–1142 (2004)
2. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes, 1st edn. Springer Publishing Company, Incorporated (2011)
3. van der Aalst, W.M.P., de Beer, H.T., van Dongen, B.F.: Process mining and verification of properties: an approach based on temporal logic. In: Meersman, R., Tari, Z. (eds.) OTM 2005. LNCS, vol. 3760, pp. 130–147. Springer, Heidelberg (2005)
4. Baier, T., Di Ciccio, C., Mendling, J., Weske, M.: Matching of events and activities - an approach using declarative modeling constraints. In: Gaaloul, K., Schmidt, R., Nurcan, S., Guerreiro, S., Ma, Q. (eds.) BPMDS 2015 and EMMSAD 2015. LNBIP, vol. 214, pp. 119–134. Springer, Heidelberg (2015)
5. Baier, T., Mendling, J., Weske, M.: Bridging abstraction layers in process mining. Information Systems **46**, 123–139 (2014)
6. Baier, T., Rogge-Solti, A., Weske, M., Mendling, J.: Matching of events and activities - an approach based on constraint satisfaction. In: Frank, U., Loucopoulos, P., Pastor, Ó., Petrounias, I. (eds.) PoEM 2014. LNBIP, vol. 197, pp. 58–72. Springer, Heidelberg (2014)
7. Rozinat, A., van der Aalst, W.M.: Conformance checking of processes based on monitoring real behavior. Information Systems **33**(1), 64–95 (2008)