Argumentation meets Process Mining: an architecture for log interpretation

Bettina **Fazzinga**[1], Sergio **Flesca**[2], Filippo **Furfaro**[2] and Luigi **Pontieri**[3]

[1]*DICES - University of Calabria, Italy*

[2]*DIMES - University of Calabria, Italy*

[3]*ICAR - CNR, Italy*

Abstract

We consider the scenario where there is an abstraction gap between the "low-level" events composing the traces in a business process log and the "high-level" activities in terms of which the analysts typically reason on the process behavior. We address the *online interpretation problem* of translating the event that has just been generated within a business process into the step of the activity instance it corresponds to. We present the architecture of a novel tool that models this interpretation problem as a dispute, encoded into an *Abstract Argumentation Framework* (AAF) [1], and that translates the computation of the valid interpretations, as well of the explanations on why the other interpretations are not valid, into instances of the AAF acceptance problem.

1. Introduction

Thanks to the increasing diffusion of automated tracing systems, the analysis of log data describing executions of business processes has gained momentum with the growth of the Process Mining research field [2]. However, all the approaches and tools developed in this field require that each log event can be mapped to well-defined activities, corresponding to some high-level view of the process. As a matter of fact, this assumption often does not hold in practice: in the logs of many processes, the 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.

Example Consider the scenario of a hospital where patient medical records are stored by keeping track of the low-level events describing the exams and the checks performed by doctors and nurses. Suppose that a trace consists of $\Phi = e_1, e_2, e_3, e_4$, where e_1 is the event *Blood sample taken*, e_2 is *Blood pressure measurement*, e_3 is *Temperature measurement*, and e_4 is *Cannula insertion*, and that each of the first 3 events can be performed during any of the high-level activities A_1 =*pre-hospitalization*, A_2 =*pre-surgery*, A_3 =*post-surgery*, while e_4 can be performed only during activity A_2. In order to reconstruct the medical history of patients, there

is the need to interpret the low-level trace in terms of high-level activities, and due to the many-to-many correspondence between events and activities, the high-level trace interpretations are many: $I_1 = A_1 A_1 A_1 A_2$, $I_2 = A_1 A_1 A_2 A_2$, $I_3 = A_2 A_2 A_2 A_2$, and so on. □

In the above-described scenario, we aim at providing the business analyst with a tool supporting *explorative analyses* of low-level traces, that can give insights on the "high-level" behavior exhibited by the monitored process. We assume the presence of some background knowledge, that is we assume that the many-to-many mapping between activities and events is known, and some information about the dependencies between the high-level activities is available, in the form of precedence rules among activities and of composition of activities. Example Continuing the previous example, suppose that we know that A_2 must always be immediately preceded by A_1, and that A_1 must always start with *Blood sample taken* and end with *Blood pressure measurement*. By using these rules, it is possible to filter out some of the high-level interpretations, and find out that the only interpretation that is consistent with the rules is I_2. □

In this context, given a trace Φ, expressed in terms of low-level events and representing an execution of a business process, we focus on the *online interpretation problem* of evaluating the answers and the explanations of *Interpretation Queries* over Φ, i.e. queries that seek for a "high"-level description of the current event in terms of the high-level activities of the business process at hand. We model the interpretation problem as an AAF, and translate the computation of interpretations and explanations on why the other interpretations are not valid into instances of the AAF acceptance problem.

2. Related work

The form of abstraction gap described above affects process logs collected in a variety of contexts, and several efforts have been devoted to provide techniques able to interpret logs in terms of activities [3, 4, 5, 6, 7, 8, 9, 10]. In our opinion, two issues undermine the applicability and usefulness of current log-abstraction solutions:

I1) The analyst is not allowed to evaluate the different alternative interpretations that likely exist for a log event/trace. Indeed, existing methods just return one "optimal" interpretation, which represents an incomplete biased view of the event/trace. This makes these methods unsuitable for operational settings (e.g., concerning security management, auditing or business-critical decision making), where the risk of incorrectly interpreting log data should be minimized. However, when many interpretations are possible, it is not sufficient to just compute and return them all directly to the analyst (e.g., as done in [3]), since without providing the analyst with effective tools for exploring and analyzing these interpretations, they may become useless.

I2) The analyst is not given the possibility to get explanations for the interpretations returned and, more importantly, for any other alternative interpretation that has been discarded (and may yet look plausible to her/him). This deficiency makes existing methods hardly trustable to "risk-averse" business users (as noticed in [11] for the whole class of Process Mining tools), since: (i) the analyst is unlikely to be capable of extracting the explanations by her/himself by looking into the raw log data and the received results, and (ii) these
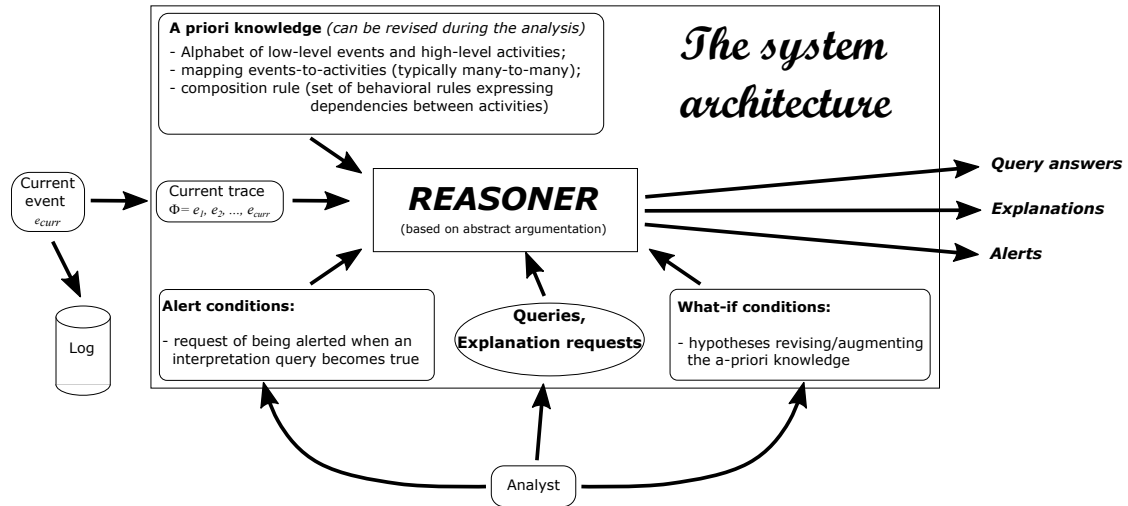
Figure 1: The system supporting the interactive exploration of ongoing low-level traces

methods typically incorporate black-boxes or combinations of heuristics that hide the rationale underlying the returned results (e.g., owing to the use of automatically-discovered sequence models [5, 6], clustering algorithms [7, 8, 9], optimal alignments [10]).

3. Our framework

Our system, whose architecture is shown in Figure 1, supports an interactive exploration of ongoing low-level traces: it provides fast answers to interpretation queries and explanation requests: for example, considering the previous example, and interpretation query could be *Can e_2 of Φ be a step of an instance of A_2?*, and its answer, along with the explanation, would be *No, as it must be a step of an instance of A_1 due to the composition rules*.

Our system, also, offers further amenities. For instance, it allows the user to set *alerts*: the analyst can ask to be warned when an interpretation query becomes true, e.g., when the system detects that the current event is (possibly or certainly) the first step of an execution of a specific activity A.

In our system, query answers and explanations are computed thanks to a reasoner, that properly exploits some background knowledge on the activities and events involved in the process. In this regard, we rely on two forms of knowledge. First, we assume that the alphabet of high-level activities is known, along with the many-to-many mapping between activities and events: that is, for each activity type A, we assume that the set of events that can be raised during any execution of A is known. Second, we assume that a declarative behavioral model of the underlying process, describing what is known about the dependencies between the high-level activities, is available. We point out that we allow activities to be complex, i.e., activities whose instances can be composed by many events, and that we allow more than one instance of the same activity to be executed at the same time.

Given this, the process of answering interpretation queries, i.e., deciding which interpre-

tations of the steps of a trace Φ make sense, is modeled as a dispute. Here, virtual agents participate by proposing either *interpretation arguments*, i.e., arguments providing possible interpretations of the trace events (according to the event-to-activity mapping), or *attacking arguments*, i.e., arguments claiming that some interpretation arguments may not be considered as valid interpretations of the corresponding event, since they may raise violations of the behavioral rules. In particular, this dispute is encoded into an instance $F(\Phi)$ of an *Abstract Argumentation Framework* (AAF) [1], that is a popular paradigm for modeling debates. In fact, we translate the problem of answering an interpretation query over Φ into the problem of verifying whether the argument proposing the interpretation specified in the query is "accepted" in $F(\Phi)$. In fact, the prototype implementation of our framework invokes the state-of-the-art argumentation solver μ-*toksia* [12] to answer interpretation queries and compute explanations, and benefits from its well-known efficiency. In particular, we model the process of reasoning on the actual cause of the current event e_{curr} of Φ as an AAF $F(\Phi)$, whose arguments can be:

1. *interpretation arguments*: arguments that propose an interpretation for a single step e_i of Φ (such as the argument α: "*Given the current composition of Φ, I believe that the event e_i occurred as the first step of the 3rd execution of activity B*");

2. *undermining arguments*: arguments that attack interpretation arguments for an event on the basis of how the other events have been interpreted (such as the argument β_1: "*Since a composition rule of the process says that an execution of activity A cannot be followed by an execution of B, and since the event preceding e_{curr} was the last step of an execution of A, the argument e_{curr} cannot be interpreted as a step of an instance of B*", or β_2: "*Since the number of instances of A started before the arrival of e_{curr} is less than 3, the event e_{curr} cannot be interpreted as the first step of the 4-th instance of A.*")

These arguments are suitably connected by means of attacks: an attack is from an argument to another one if the first argument undermines or rebuts or undercuts the second. For instance, for the above arguments α, β_1, an attack is put from β_1 to α. $F(\Phi)$ is built progressively: as soon as a new event e_i is detected and appended to Φ, the AAF $F(\Phi)$ is built by adding to the "old" $F(\Phi)$ new interpretation arguments regarding the current step e_{curr}, as well as new undermining arguments and attacks.

The point is that, once $F(\Phi)$ is constructed, it can be used to "simulate" a complex form of reasoning on the interpretation of the trace steps. This is allowed by the property that an interpretation argument α for e_{curr} is credulously (resp., skeptically) accepted if and only if the interpretation $I(e_{curr})$ coinciding with α is a credulously (resp., skeptically) valid interpretation for e_{curr}. Thus, the analyst can elaborate on the reasonability of a candidate interpretation of e_{curr} by solving an instance of the acceptance problem over $F(\Phi)$. Analogously, an explanation of why an interpretation is not accepted can be obtained by solving other instances of the acceptance problem.

4. Conclusion

We have presented an architecture for supporting an interactive analysis of low-level process logs, that aims at interpreting each event of a given trace as a step of an activity instance, based on a loose description of the process activities. We model the interpretation process as

an AAF and compute the valid interpretations of the trace and the explanations of why other interpretations are not valid by solving an instance of the acceptance problem.

References

[1] P. M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, Artif. Intell. 77 (1995) 321–358.

[2] W. Van Der Aalst, Data science in action, Springer, 2016.

[3] B. Fazzinga, S. Flesca, F. Furfaro, E. Masciari, L. Pontieri, Efficiently interpreting traces of low level events in business process logs, Inf. Syst. 73 (2018) 1–24.

[4] B. Fazzinga, F. Folino, F. Furfaro, L. Pontieri, An ensemble-based approach to the security-oriented classification of low-level log traces, Expert Syst. Appl. 153 (2020) 113386.

[5] N. Tax, Human activity prediction in smart home environments with lstm neural networks, in: 2018 14th International Conference on Intelligent Environments (IE), IEEE, 2018, pp. 40–47.

[6] D. R. Ferreira, F. Szimanski, C. G. Ralha, Improving process models by mining mappings of low-level events to high-level activities, Journal of Intelligent Information Systems 43 (2014) 379–407.

[7] T. Baier, J. Mendling, M. Weske, Bridging abstraction layers in process mining, Information Systems 46 (2014) 123–139.

[8] T. Baier, C. Di Ciccio, J. Mendling, M. Weske, Matching events and activities by integrating behavioral aspects and label analysis, Software and Systems Modeling 17 (2018). doi:10.1007/s10270-017-0603-z.

[9] G. Tello, G. Gianini, R. Mizouni, E. Damiani, Machine Learning-Based Framework for Log-Lifting in Business Process Mining Applications, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 11675 LNCS, 2019.

[10] F. Mannhardt, M. de Leoni, H. A. Reijers, W. M. P. van der Aalst, P. J. Toussaint, From low-level events to activities-a pattern-based approach, in: Proc. 14th Int. Conf. on Business Process Management (BPM), 2016, pp. 125–141.

[11] S. T. K. Jan, V. Ishakian, V. Muthusamy, AI Trust in Business Processes: The Need for Process-Aware Explanations. BT - The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth A, 2020.

[12] A. Niskanen, M. Järvisalo, μ-toksia: An efficient abstract argumentation reasoner, in: Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020), AAAI Press, United States, 2020.