Aligning Real Process Executions and Prescriptive Process Models through Automated Planning

M. de Leoni, A. Marrella

Eindhoven University of Technology, The Netherlands
Sapienza University of Rome, Italy

A R T I C L E   I N F O

Article history:
Received 17 September 2016
Revised 21 March 2017
Accepted 22 March 2017
Available online 31 March 2017

Keywords:
Conformance checking
Automated planning
Business process management
Process mining

A B S T R A C T

Modern organizations execute processes to deliver product and services, whose enactment needs to adhere to laws, regulations and standards. Conformance checking is the problem of pinpointing where deviations are observed. This paper shows how instances of the conformance checking problem can be represented as planning problems in PDDL (Planning Domain Definition Language) for which planners can find a correct solution in a finite amount of time. If conformance checking problems are converted into planning problems, one can seamlessly update to the recent versions of the best performing automated planners, with evident advantages in terms of versatility and customization. The paper also reports on results of experiments conducted on two real-life case studies and on eight larger synthetic ones, mainly using the Fast-downward planner framework to solve the planning problems due to its performances. Some experiments were also repeated though other planners to concretely showcase the versatility of our approach. The results show that, when process models and event logs are of considerable size, our approach outperforms existing ones even by several orders of magnitude. Even more remarkably, when process models are extremely large and event log traces very long, the existing approaches are unable to terminate because they run out of memory, while our approach is able to properly complete the alignment task.

© 2017 Elsevier Ltd. All rights reserved.

1. Introduction

Process mining is about extracting knowledge from event logs commonly available in today’s information systems. These techniques provide new means to discover, monitor and improve processes in a variety of application domains.

Whereas large attention has been paid to discovery models that summarize the behavior observed in the event log, less effort has been put on conformance checking, which aims to find commonalities and discrepancies between the modeled and the observed behavior.

However, a number of previous works advocates the use of conformance checking for business alignment and auditing (see, e.g., Accorsi & Stocker, 2012; Hosseinpour & Jans, 2016); It allows one to check whether business processes are executed within certain boundaries set by managers, governments, laws, national and international regulations and standards, etc. Conformance checking starts from an event log, which records the actual process executions, and a model that encodes the boundaries and the constraints mentioned above. The practical relevance of conformance checking is demonstrated through several successful case studies in multiple domains, of which a partial list is reported in Section 6.1.

Several notions were introduced for conformance checking, such as token-based replay and comparison of footprints (see, e.g., van der Aalst, 2016) but they were unable to exactly pinpoint the deviations causing nonconformity. To overtake this limitation, the notion alignment has been introduced (van der Aalst, Adrian-syah, & van Dongen, 2012). An alignment between a recorded process execution and a process model is a pairwise matching between activities recorded in the log and activities allowed by the model. Sometimes, activities as recorded in the log (events) cannot be matched to any of the activities allowed by the model (process activities).

For instance, an activity is executed when not allowed (e.g., a loan is opened before assessing the applicant). In this case, we match the event with a special null activity (hereafter, denoted as ⊙), thus resulting in so-called moves in lag. Other times, an activity should have been executed but is not observed in the event log (e.g., a loan is not opened after positively assessing the applicant). This results in a process activity that is
matched to a $\Rightarrow$ event, thus resulting in a so-called move in model.

Alignments are powerful artifacts to detect nonconformity between the observed behavior as recorded in the event log and the prescribed behavior as represented by process models. If an alignment between a log trace and process model contains at least one move in log/model, it means that the trace refers to a process execution that is not compliant with the allowed behavior represented by the process model. As a matter of fact, the moves in log/model indicate where the execution is not conforming by pinpointing the deviations that have caused this nonconformity. Pinpointing the actual reasons of nonconformity is crucial for, e.g., auditors.

In general, a large number of possible alignments exist between a process model and a log trace, since there may exist manifold explanations why a trace is not conforming. It is clear that one is interested in finding the most probable explanation. In van der Aalst et al. (2012), an approach is proposed that is based on the principle of the Occam’s razor: the most parsimonious explanation is preferable. Therefore, one should not aim to find any alignment but, in fact, one of the alignments with the least expensive deviations (one of the so-called optimal alignments), according to some function assigning costs to deviations.

The existing techniques to compute optimal alignments (Adriansyah, van Dongen, & van der Aalst, 2013a; Adriansyah, Sidoroa, & van Dongen, 2011) provide ad-hoc implementations of the A* algorithm based on an ad-hoc heuristics. This paper starts from the belief that re-implementing standard planning techniques for solving specific planning problems in an ad-hoc way is not ideal. The main drawback is that it is not possible to seamlessly plug in new planning algorithms and evaluate the performance levels. As a consequence, the possibility of evaluating different alternatives is hampered; also, if the literature proposes new algorithms that clearly overtake the existing ones for solving instances of the alignment problem, a massive amount of work is needed to modify the implementation.

Hence, in order to facilitate the integration of different planning algorithms, this paper illustrates how the problem of computing optimal alignments can be formulated as a planning problem in PDDL (Planning Domain Definition Language) (McDermott et al., 1998), which can be solved through off-the-shelf automated planners. PDDL is the standard encoding language for planning tasks. It allows one to explicitly represent states of the world and actions through a planning domain, and to instantiate such a domain with concrete objects, an initial state and a goal specification (planning problem).

In a nutshell, given a process model and a real process execution recorded in an event log, this paper shows how to build a planning domain and problem instance such that the solution steps of the problem are guaranteed to correspond to the alignment steps. Also, the paper illustrates how the encoding of alignment problem always generates planning problems that can be solved by planning systems in a finite amount of time.

An evaluation was performed on real-life process models and event logs and on synthetic models and logs of increasing sizes to analyze the scalability of our approach. The evaluation results show that, when the process models are larger (more than 100 activities), it is significantly faster than the existing techniques reported in Adriansyah et al. (2013a, 2011). The latter are the only techniques that compute alignments with guarantee of optimality. Even more remarkably, when process models are extremely large and event log traces very long, the existing techniques were unable to compute alignments because they were unable to operate with 16 Gb of RAM (they went out of memory after several hours of computation), while our plan-based approach was able to properly complete the alignment task.

The need of approach that scales better is also advocated by the IEEE Task Force in Process Mining (van der Aalst et al., 2011): “In some domains, mind-boggling quantities of events are recorded. [...] Therefore, additional efforts are needed to improve performance and scalability.” While experiments were largely conducted through Fast-DOWNWARD because of being the best performing, the paper also reports on experience with other planners. The intention is also to showcase how the approach is versatile and allows one to plug in new planners at basically no cost.

The rest of the paper is organized as follows. In Section 2 we provide the relevant background necessary to understand the paper. In Section 3 we illustrate our approach to convert alignment problems to planning problems, and we provide correctness results for such problems. In Section 4 we describe the architecture of the software tool implementing our planning-based alignment approach. Then, Section 5 reports on experiment results, while in Section 6 we discuss related works. Finally, Section 7 concludes the paper.

2. Preliminaries

In this section, we provide some preliminary concepts used throughout the paper. In Section 2.1 we introduce the Petri Net modeling language. In Section 2.2 we describe the problem we want to solve through planning: constructing an optimal alignment between event logs and process models represented as Petri nets. In Section 2.3 we give an overview on automated planning.

2.1. Petri nets

Many notations have been introduced to represent business processes, such as BPMN, EPC, YAWL or UML Activity Diagrams (Dumas, La Rosa, Mendling, & Reijers, 2013), and some of those are characterized by an ambiguous semantics. Since we need a simple language with clear semantics to explain our technique, we opted for Petri nets. Despite of its simplicity, it has been proven to be sufficiently adequate to model crucial aspects of business processes (van der Aalst, 1998) and to check their conformance against real executions recorded in event logs (see, e.g., the case studies reported in van der Aalst, 2016). The practical suitability of Petri nets for conformance checking is also showcased through two real-life case studies in Section 5.1.

It is undeniable that there are business process aspects that cannot be modeled through Petri nets, including the objects manipulated by activities, the activity guards or constraints on activities to only be executed by resources belonging to certain organizational units. They can certainly be modeled by richer languages, e.g., BPMN, but this addition aspects require an increase of the language expressiveness to an extent that makes the alignment computation an undecidable problem (cf. Section 7). However, our approach can easily be applied to any other process modeling language with the same expressiveness or, for more expressive languages, if the constructs providing more expressiveness are not employed.

A Petri net is a directed graph with two node types called places and transitions. The nodes are connected via directed arcs. Connections between two nodes of the same type are not allowed. Places are represented by circles and transitions by rectangles.

Definition 1 (Petri Net). A Petri net is a tuple $(P, T, F)$ where:

- $P$ is a finite set of places;
- $T$ is a finite set of transitions;
- $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation between places and transitions (and between transitions and places).

Given a transition $t \in T$, $^*t$ is used to indicate the set of input places of $t$, which are the places $p$ with a directed arc from $p$ to
دریافت فوری متن کامل مقاله

امکان دانلود نسخه تمام متن مقالات انگلیسی
امکان دانلود نسخه ترجمه شده مقالات
پذیرش سفارش ترجمه تخصصی
امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
امکان دانلود رایگان ۲ صفحه اول هر مقاله
امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
دانلود فوری مقاله پس از پرداخت آنلاین
پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات