A Modular SMT-based Approach for Data-aware Conformance Checking

Paolo Felli[1], Alessandro Gianola[1], Marco Montali[1], Andrey Rivkin[1] and Sarah Winkler[1]

[1]*Free University of Bozen-Bolzano, Italy*

Abstract

In the last years, automated reasoning techniques have been gaining momentum in the BPM community. One of their recent applications is related to Process Mining. In this work we briefly describe how the conformance checking problem can be addressed using satisfiability modulo theories (SMT).

Keywords

Conformance Checking, SMT, Data-Aware Processes, Data Petri Nets

1. Introduction

In this survey paper, we give an overview of recent results on the application of techniques based on Satisfiability Modulo Theories (SMT) in the context of multi-perspective Process Mining: specifically, we present an approach that leverage SMT-based techniques for performing conformance checking of data-aware processes over logs with (in)complete data.

Process mining (PM) is an active research field that originally stems from business process management (BPM), and involves frameworks and methods taken from data science, formal methods and AI. The original idea of PM is to provide a series of techniques supporting analysis of operational processes based on event log data. Such techniques can be subdivided into three main groups [1]: *(i) process discovery* focuses on constructing a process model that is representative of all the behaviors observed in the event log; *(ii) conformance checking* searches for deviations and commonalities between a given process model and an event log; *(iii) enhancement*, offers a set of on-line techniques using incoming event data so as to improve and optimize an already existing process. For more information on the process mining techniques we refer to [2].

In this paper, we deal with the conformance checking [3] task whose main idea, as we have mentioned above, is the detection of behavioral discrepancies between the process model and a

OVERLAY 2022: 4th Workshop on Artificial Intelligence and Formal Verification, Logic, Automata, and Synthesis, November 28, 2022, Udine, Italy

pfelli@inf.unibz.it (P. Felli); gianola@inf.unibz.it (A. Gianola); montali@inf.unibz.it (M. Montali); rivkin@inf.unibz.it (A. Rivkin); winkler@inf.unibz.it (S. Winkler)

https://www.pfelli.xyz/ (P. Felli); https://gianola.people.unibz.it/ (A. Gianola); https://www.inf.unibz.it/~montali/ (M. Montali); https://rivkin.people.unibz.it/ (A. Rivkin); http://cl-informatik.uibk.ac.at/users/swinkler/verona/ (S. Winkler)

0000-0001-9561-8775 (P. Felli); 0000-0003-4216-5199 (A. Gianola); 0000-0002-8021-3430 (M. Montali); 0000-0001-8425-2309 (A. Rivkin); 0000-0001-8114-3107 (S. Winkler)

© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

log. Notably, one may choose among different process representations, ranging from classical workflow nets [4] to declarative data-aware processes [5], as well as logs, considering those with additional data on activity payloads, incomplete information, relations between objects used by operational processes etc. In this multitude of approaches, notice that conformance checking quite often becomes more challenging when one goes beyond the control-flow perspective and, as we are interested here, tries to account for, e.g., unbounded data.

Here, we use data Petri nets (DPNs for short) as the reference process models. DPNs have been extensively studied in the context of formal verification [6, 7] as well as process mining [8, 9, 10]. Notably, various techniques proposed for DPNs are rather ad-hoc and work under restrictive assumptions on the data dimension (e.g., limited support of data types).

To alleviate such restrictions, we introduced in [11] a framework based on well-established automated reasoning techniques such as the ones provided by Satisfiability Modulo Theories (SMT). SMT provides a more universal, algorithmically robust and flexible approach, whose use allows us to support complex forms of unbounded data constrained by expressive first-order theories: SAT-based approaches would be sufficient to capture the control-flow behavior in isolation, but not its interaction with data, which causes the system to be infinite-state. Instead, by relying on an SMT backend, we can support data and operations from a variety of theories such as arithmetics, without significantly restricting the data dimension. We also discuss how this framework can be adapted to address data-aware conformance checking over uncertain logs, by providing a suitably extended notion of alignment. In this case, event logs incorporate complex forms of uncertainty, regarding the recorded timestamps, data values, and/or events.

2. The framework

In this section we present our SMT-based approach to conformance checking [11]. In a nutshell, the conformance checking problem takes as input a log/a trace and a process model, and computes various artefacts linking observed and modeled behaviors. Such artefacts are then used for such purposes for deviation detection, (process model) quality metrics computation etc. In this section, we first briefly discuss the type of logs we are interested in as well as the process models we use, and then introduce the conformance checking problem together with the artefacts of interest. Finally, we discuss how to encode all these elements into SMT.

Logs with uncertainties and Petri nets with payloads. Given a set A of activity labels and a set of attributes names Y, an event is a tuple $(b, \alpha, \mathsf{c}, \mathsf{LA}, \mathsf{TS})$, where $b \in A$, α is a (partial) payload function associating values to attributes from Y, $\mathsf{c} \in (0, 1]$ is a confidence that the event actually happened, LA is a set of activities with related confidence values, and TS is a finite set of elements from (or an interval over) a totally ordered set of timestamps. The last three values are crucial for specifying the following uncertainties: *(i) uncertain events*, which come with confidence values describing how certain we are about the fact that recorded events actually happened during the process execution; *(ii) uncertain timestamps* that are either coarse or come as intervals, which in turn affect the ordering of the events in the log; *(iii) uncertain activities* that in bundle (together with corresponding confidence values) are assigned to a concrete event when we are not certain which of the activities actually caused it; *(iv) uncertain data values* which, for an attribute, come as a set or an interval of possible values. For a set of events $\mathcal{E}$, a

log trace is a sequence $\mathbf{e} \in \mathcal{E}^*$ and a log is a multiset of log traces.

As we have shown above, events come with activity payloads, which differs from the majority of process mining tasks in which one essentially deals with processes characterized by timestamps and activity labels only [4]. Processes with payloads can be represented using data Petri nets (DPNs for short) [8, 9, 10], which are our reference process model. A DPN is a labeled Petri net extended with *read* and *write* guards defined over a finite set of (typed) net variables V. Every guard is a boolean combination of atoms $x_1 \odot x_2$, where $\odot$ is a type-specific predicate.

The execution semantics of DPNs extends the one of place-transition nets with the ability to check and manipulate the net's variables via transition guards. A transition is *enabled* iff all the input places of the transition contain sufficiently many tokens to consume, and its read and write guards are satisfied under a given "firing mode" β assigning values only to variables of the guard. For the read guards, values are picked from the current net state variable assignment, whereas for write guards values are provided by β. An enabled transition may *fire*, consuming the necessary amount of tokens from its input places and producing tokens in its output places, and also updating the state using β. All other values assigned to V remain the same.

Here DPNs are considered relaxed data sound, i.e., there exists at least one sequence of transition firings bringing to a given final marking. Such transitions are called valid.

The conformance checking problem. Conformance checking looks into comparing the observed behavior in the log with the one allowed by the process model. This comparison can be achieved by computing various analytic (or conformance) artefacts [3]. One of the main artefacts is given by (optimal) alignments, where an *alignment* is a finite sequence of *moves* (event-firing pairs) s.t. each (e, f) is *(i)* a log move, if $e \in \mathcal{E}$ and $f = \gg$ ($\gg$ is a symbol denoting skipping); *(ii)* a model move, if $e = \gg$ and f is a valid transition firing; *(iii)* a synchronous move otherwise. Given a log trace and a process model, a corresponding alignment is computed by finding a complete process run that is "the best" among all those the process model can generate. This is done by searching for an *optimal* alignment using a specific cost function, which can be realized as a distance between two finite strings [12]. In case of logs with uncertainties, one looks into alignments obtained for *realizations* – possible sequentializations of (a subset of the) events with uncertainty in a given trace so that timestamp uncertainties are resolved [13].

Encoding into SMT. We now present how the conformance checking problem and related artefacts can be *modularly* encoded using SMT. The detailed encoding can be found in [11, 13], whereas here we focus on how conformance checking can be solved using SMT and on the modular organization of the encoding, where each module accounts for one of the problem components and can be seamlessly replaced when, for example, other types of process models or conformance artefacts are required. The main modules of this encoding are as follows: ***(i)*** the executable process model (**EPM**) module, which realizes the execution semantics of the input net by symbolically encoding all possible valid process runs; ***(ii)*** the log trace (**LT**) module, which accounts for the trace realization in case it contains any uncertainties; ***(iii)*** the conformance artefacts (**CA**) module, which accounts for a conformance artefact of interest as well as for its related *decision/optimization problem*, i.e., the task that needs to be solved (or decided) in order to compute such artefact. Each module (but the part of **CA** related to the decision problem) is represented as a conjunction of SMT constraints. It is important to notice that the process model **EPM** is usually given by DPNs, and encodes process runs whose length does not exceed

a given upper bound. This, however, does not affect the optimality of the final solution and has been extensively discussed in [11, 13]. The **LT** module naturally encodes all the aforementioned uncertainties (if any) and, in case of the uncertainty on timestamps, for related events it provides an additional encoding to find a suitable ordering for them.

All the SMT constraints are then put into one conjunction, which is passed as input to an SMT solver in order to solve the associated decision problem. For the case of alignments, **CA** encodes a selected distance-based cost function defined as the sum of contributions that only depend on the local discrepancies in the moves of the considered alignment (i.e., inductively), and this is done using a finite set of variables that are later used to extract the optimal alignment solution. However, since there are in principle many possible satisfiable solutions, in order to find the 'best/optimal one', we solve a constrained optimization problem for which a satisfying assignment to all the variables in the encoding is so that the total cost (represented as a single variable carrying the final result of the aforementioned inductive cost encoding) is minimal. This allows us to find a model run which is the 'closest one' to the trace.

In the same manner, one can adjust **CA** by changing the decision problem associated to the specific conformance artefact of interest. Like that we proposed in [11] slight encoding modifications so as to account for multi- and anti-alignments [14, 15]. To capture different results while computing conformance artefacts, one can also modify **CA** so as to account for different cost functions. For example, in [13] we demonstrated that one can employ a generic cost function whose components can be flexibly instantiated to homogeneously account for a variety of concrete situations where uncertainty come into play.

After an SMT solver finds a satisfiable solution, we use the known correspondence between an edit distance and alignments [16] to decode the solution and obtain the optimal alignment.

3. Conclusions and future directions

The discussed approach has been implemented and tested on publicly available DPN models and logs. More on that can be found on the tool webpage: https://github.com/bytekid/cocomot.

We see a few advantages of using SMT for conformance checking. First, SMT solving provides a *universal* and *flexible* setting where sophisticated reasoning tasks can be accomplished. In the case of conformance checking, one obtains a tool-agnostic framework in which the reasoning part is delegated to an SMT solver, whereas the user needs only to provide suitable encodings. However, it is important to investigate meta-properties of such encodings in order to understand which conformance checking scenarios can be addressed in the SMT-based setting. For example, the current encodings consider bounded model runs and can be used only for inductively-definable artefacts. Another advantage of SMT is a multitude of supported *background theories* that can be used for capturing manipulated data and expressing sophisticated cost functions, e.g., involving background knowledge coming from additional data sources (in line of [17, 18]).

In the future, we plan to investigate how to employ richer SMT theories for the case of DPNs and other object/data-aware process models [19, 20, 21, 22, 23]. Given that SMT can be also used for encoding declarative processes [24], we can use our approach to attack conformance checking for Declare and its data-aware extensions [19, 20, 21]. Another interesting future task is a study of the limitations of the SMT technique in the context of conformance checking.

Acknowledgments

This work is partially supported by the Italian Ministry of University and Research under the PRIN program, grant B87G22000450001 (PINPOINT), and by the Free University of Bozen-Bolzano with the SMART-APP and ADAPTERS projects.

References

[1] W. M. P. van der Aalst, Process Mining - Data Science in Action, Second Edition, Springer, 2016. URL: https://doi.org/10.1007/978-3-662-49851-4. doi:10.1007/978-3-662-49851-4.

[2] W. M. P. van der Aalst, J. Carmona (Eds.), Process Mining Handbook, volume 448 of *Lecture Notes in Business Information Processing*, Springer, 2022. URL: https://doi.org/10.1007/978-3-031-08848-3. doi:10.1007/978-3-031-08848-3.

[3] J. Carmona, B. F. van Dongen, A. Solti, M. Weidlich, Conformance Checking - Relating Processes and Models, Springer, 2018.

[4] W. M. P. van der Aalst, Process Mining – Discovery, Conformance and Enhancement of Business Processes, Springer, 2011.

[5] A. Burattin, F. M. Maggi, A. Sperduti, Conformance checking based on multi-perspective declarative process models, Expert Syst. Appl. 65 (2016) 194–211. URL: https://doi.org/10.1016/j.eswa.2016.08.040. doi:10.1016/j.eswa.2016.08.040.

[6] M. de Leoni, P. Felli, M. Montali, A holistic approach for soundness verification of decision-aware process models, in: Proc. 37th International Conference on Conceptual Modeling, volume 11157 of *LNCS*, 2018, pp. 219–235. doi:10.1007/978-3-030-00847-5_17.

[7] P. Felli, M. de Leoni, M. Montali, Soundness verification of decision-aware process models with variable-to-variable conditions, in: Proc. 19th ACSD, IEEE, 2019, pp. 82–91.

[8] F. Mannhardt, M. de Leoni, H. A. Reijers, W. M. P. van der Aalst, Decision mining revisited - discovering overlapping rules, in: Proc. 28th CAiSE, volume 9694 of *LNCS*, 2016, pp. 377–392.

[9] F. Mannhardt, M. de Leoni, H. Reijers, W. van der Aalst, Balanced multi-perspective checking of process conformance, Computing 98 (2016) 407–437. doi:10.1007/s00607-015-0441-1.

[10] F. Mannhardt, Multi-perspective Process Mining, Ph.D. thesis, Technical University of Eindhoven, 2018.

[11] P. Felli, A. Gianola, M. Montali, A. Rivkin, S. Winkler, CoCoMoT: Conformance checking of multi-perspective processes via SMT, in: Proc. of BPM 2021, volume 12875 of *LNCS*, Springer, 2021, pp. 217–234. doi:10.1007/978-3-030-85469-0_15.

[12] A. Adriansyah, Aligning observed and modeled behavior, Ph.D. thesis, Technische Universiteit Eindhoven, 2014.

[13] P. Felli, A. Gianola, M. Montali, A. Rivkin, S. Winkler, Conformance checking with uncertainty via SMT, in: Proc. of BPM 2022, volume 13420 of *LNCS*, Springer, 2022, pp. 199–216.

[14] T. Chatain, J. Carmona, Anti-alignments in conformance checking – the dark side of

process models, in: Proc. 37th Petri Nets, volume 9698 of *LNCS*, 2016, pp. 240–258. doi:10.1007/978-3-319-39086-4_15.

[15] M. Boltenhagen, T. Chatain, J. Carmona, Encoding conformance checking artefacts in SAT, in: Proc. Business Process Management Workshops 2019, volume 362 of *LNCS*, 2019, pp. 160–171. doi:10.1007/978-3-030-37453-2_14.

[16] S. B. Needleman, C. D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins, Journal of Molecular Biology 48 (1970) 443–453. doi:https://doi.org/10.1016/0022-2836(70)90057-4.

[17] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, SMT-based verification of data-aware processes: a model-theoretic approach, Math. Struct. Comput. Sci. 30 (2020) 271–313.

[18] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Formal modeling and SMT-based parameterized verification of data-aware BPMN, in: Proc. BPM 2019, volume 11675 of *LNCS*, Springer, 2019, pp. 157–175.

[19] R. D. Masellis, F. M. Maggi, M. Montali, Monitoring data-aware business constraints with finite state automata, in: H. Zhang, L. Huang, I. Richardson (Eds.), Proc. ICSSP '14, ACM, 2014, pp. 134–143. doi:10.1145/2600821.2600835.

[20] A. Burattin, F. M. Maggi, A. Sperduti, Conformance checking based on multi-perspective declarative process models, Expert Syst. Appl. 65 (2016) 194–211.

[21] A. Polyvyanyy, J. M. E. M. van der Werf, S. Overbeek, R. Brouwers, Information systems modeling: Language, verification, and tool support, in: Proc. CAiSE 2019, volume 11483 of *LNCS*, 2019, pp. 194–212.

[22] F. M. Maggi, M. Montali, U. Bhat, Compliance monitoring of multi-perspective declarative process models, in: Proc. EDOC 2019, IEEE, 2019, pp. 151–160. doi:10.1109/EDOC.2019.00027.

[23] S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Petri net-based object-centric processes with read-only data, Information Systems 107 (2022).

[24] V. Fionda, A. Guzzo, Control-flow modeling with declare: Behavioral properties, computational complexity, and tools, IEEE Trans. Knowl. Data Eng. 32 (2020) 898–911. doi:10.1109/TKDE.2019.2897309.