An ILASP-Based Approach to Repair Petri Nets (Extended Abstract)

Francesco Chiariello[1][0000−0001−7855−7480], Antonio Ielo[2][0009−0006−9644−7975], and Alice Tarzariol[3][0000−0001−6586−3649]

[1] ANITI, IRIT, University of Toulouse, France
[2] University of Calabria, Italy
[3] University of Klagenfurt, Austria

1 Introduction

Process Mining (PM) [3] is an interdisciplinary field at the intersection of Business Process Management (BPM) and Data Mining that aims at getting insight into operational processes by analyzing event logs as recorded by enterprise information systems. An operational process, or simply a *process*, expresses the relationships among the activities an organization performs to achieve a goal. Several formalisms have been proposed to model and reason about processes, including Petri nets [2, 4], the *de facto* standard model in BPM. An *event log*, i.e., the collection of activities the organization executes while enacting the process, can then be analyzed to perform various PM tasks. In particular, *Model Repair* is the task of revising a process model to make it conformant to the log [9]. Multiple solutions exist to repair Petri nets [10, 12, 7]. However, none of them allow the user to easily specify the edit operations, the structural properties of the net [1], or to exploit domain knowledge [13], e.g., by specifying portions of the net that should not be involved with the edits. Besides, those approaches do not guarantee full conformance to the log. In the paper referred by this extended abstract [8], we work towards overcoming these limitations, by proposing a fully declarative, user-customizable framework for Petri net model repair based on the Inductive Logic Programming (ILP) system ILASP [11]. In particular, we illustrate how to set the elements of a learning task for ILASP to tackle the model repair of a Petri net. The learned solution represents edit operations that, applied to $\mathcal{M}$, make it satisfy the structural requirements and accept all the traces in L. Our experiments show the proposed approach is effective on medium-sized Petri nets where domain knowledge about repairs is available.

2 Model Repair in ILASP

ILASP is an ILP system for learning ASP programs. An ILP task comprises four elements: the hypothesis space, H; the background knowledge, B; and two sets of positive and negative examples, E^+ and E^-, respectively. The learned solution is a subset $S \subseteq H$ such that $B \cup S$ covers all the examples in E^+, and no examples in E^-.

The hypothesis space of our ILP task contains *edit operations* allowed on the original Petri net $\mathcal{M}$. We define it explicitly in ASP, as this setting yields a flexible framework where users can customize the model repair actions in a declarative fashion. The background knowledge is defined by three components: $\Pi_{\mathcal{M}}$, encoding in ASP the model $\mathcal{M}$, Π_{edit}, which produces a new Petri net by applying the edit operations to $\mathcal{M}$, and lastly $\Pi_{semantics}$, the program modelling the Petri net semantics by adapting the one introduced in [6]. The positive and negative examples of our learning task derive from two elements: the event log L and the structural requirements. The former can be split into two sets, L^+ and L^-, containing respectively traces that should or should not characterize the considered process. We allow this flexibility since it has recently been observed that negative examples are a valuable source of information, and thus being increasingly adopted in industry [5, 14]. However, our setting allows performing model repair even when L^- is empty. Besides conformance on L, we also exploited negative examples to enforce structural properties to the revised model by defining in their context the ASP rules containing constraints that are triggered when a desired condition is not met.

2.1 Experiments

After defining our approach for tackling the model repair problem, we implemented and assessed its feasibility. Our code, settings, and detailed results from experiments are available online [4]. The experiments consider a test case to evaluate the quality of the repairs found, as well as aggregate results for evaluating the performance of our approach with respect to the percentage of edits introduced on a synthetic model originating from real logs. The description of the experiments and the results can be found in our original paper [8].

3 Conclusions

In this work, we introduced an ILP-based approach to repair Petri net process models using ILASP. Our approach allows users to define edit operations (as logic programs), can be easily extended to handle noisy examples and to support different firing semantics. Experiments confirm this is a feasible approach for medium-sized models and models where domain knowledge is available. Our ASP-based approach makes it easy to seamlessly support many interesting features. For example, users can choose which parts of the input model should be kept as-is (e.g., unaffected by edit operations), as well as define constraints on the underlying bipartite graph topology of the Petri net (e.g., force the repaired model to be a WF-net, or having at most n outgoing arcs). Another interesting aspect, assuming a fixed set of edit operations, is that ILASP can process examples *incrementally* and asynchronously (making use of caches to store intermediate optimal solutions). This makes our approach a good basis for an interactive and (counter-)example-driven process model repair tool [7].

[4] `www.github.com/ainnoot/ilp-pn-repair`

Acknowledgments. Francesco Chiariello was partially supported by the program France 2030 under the Grant agreement n°ANR-19-PI3A-0004. Antonio Ielo was partially supported by the Italian Ministry of Research (MUR) under PRIN projects PINPOINT - CUP H23C22000280006 and FAIR "Future AI Research" - Spoke 9 - WP9.1 - CUP H23C22000 860006.

References

[1] Wil M. P. van der Aalst. "Discovering Directly-Follows Complete Petri Nets from Event Data". In: *A Journey from Process Algebra via Timed Automata to Model Learning*. Vol. 13560. Lecture Notes in Computer Science. Springer, 2022, pp. 539–558.

[2] Wil M. P. van der Aalst. "The Application of Petri Nets to Workflow Management". In: *J. Circuits Syst. Comput.* 8.1 (1998), pp. 21–66.

[3] Wil M. P. van der Aalst and Josep Carmona, eds. *Process Mining Handbook*. Vol. 448. Lecture Notes in Business Information Processing. Springer, 2022.

[4] Wil M. P. van der Aalst and Christian Stahl. *Modeling Business Processes - A Petri Net-Oriented Approach*. Cooperative Information Systems series. MIT Press, 2011.

[5] Simone Agostinelli et al. "Process mining meets model learning: Discovering deterministic finite state automata from event logs for business process analysis". In: *Inf. Syst.* 114 (2023), p. 102180.

[6] Saadat Anwar, Chitta Baral, and Katsumi Inoue. "Encoding Petri Nets in Answer Set Programming for Simulation Based Reasoning". In: *Theory Pract. Log. Program.* 13 (2013).

[7] Abel Armas-Cervantes et al. "Incremental and Interactive Business Process Model Repair in Apromore". In: *BPM (Demos)*. Vol. 1920. CEUR Workshop Proceedings. CEUR-WS.org, 2017.

[8] Francesco Chiariello, Antonio Ielo, and Alice Tarzariol. "An ILASP-Based Approach to Repair Petri Nets". In: *Logic Programming and Nonmonotonic Reasoning*. Ed. by Carmine Dodaro, Gopal Gupta, and Maria Vanina Martinez. Cham: Springer Nature Switzerland, 2025, pp. 85–97. ISBN: 978-3-031-74209-5.

[9] Dirk Fahland and Wil M. P. van der Aalst. "Model repair - aligning process models to reality". In: *Inf. Syst.* 47 (2015), pp. 220–243.

[10] Dirk Fahland and Wil M. P. van der Aalst. "Repairing Process Models to Reflect Reality". In: *BPM*. Vol. 7481. Lecture Notes in Computer Science. Springer, 2012, pp. 229–245.

[11] Mark Law. "Conflict-Driven Inductive Logic Programming". In: *Theory Pract. Log. Program.* 23.2 (2023), pp. 387–414.

[12] Artem Polyvyanyy et al. "Impact-Driven Process Model Repair". In: *ACM Trans. Softw. Eng. Methodol.* 25.4 (2017), 28:1–28:60.

[13] Daniel Schuster, Sebastiaan J. van Zelst, and Wil M. P. van der Aalst. "Utilizing domain knowledge in data-driven process discovery: A literature review". In: *Comput. Ind.* 137 (2022), p. 103612.

[14] Tijs Slaats, Søren Debois, and Christoffer Olling Back. "Weighing the Pros and Cons: Process Discovery with Negative Examples". In: *BPM*. Vol. 12875. Lecture Notes in Computer Science. Springer, 2021, pp. 47–64.