Binary Discovery of Declarative Business Processes with ASP Preferences

Federico Chesani[1], Chiara Di Francescomarino[2], Chiara Ghidini[3], Daniela Loreti[1], Fabrizio Maria Maggi[4], Paola Mello[1], Marco Montali[4], Sergio Tessaris[4]

[1]DISI - University of Bologna, Italy
[2]DISI - University of Trento, Italy
[3]Fondazione Bruno Kessler, Trento, Italy
[4]Free University of Bozen/Bolzano, Italy

Abstract

Process discovery techniques focus on learning a process model starting from a given set of logged traces. The majority of the discovery approaches, however, only consider one set of examples to learn from, i.e., the log itself. Some recent works on declarative process discovery, instead, advocated the usefulness of taking into account two different sets of traces (a.k.a. positive and negative examples), with the goal of learning a set of constraints that is able to discriminate which trace belongs to which set. In this paper we recall our recent work on the discovery of process models from positive and negative examples, with the goal of learning a set of declarative constraints that is able to discriminate which trace belongs to which set, also taking into account user preferences on activities and constraint templates to be used to build the final set of constraints. The approach is grounded in a logic-based framework that provides a sound and formal meaning to the notion of expert preferences.

Introduction

Process discovery is one of the most investigated process mining techniques (van der Aalst, et al. 2012). It deals with the automatic learning of a process model from a given set of logged traces, each one representing the digital footprint of the execution of a case.

If we focus on the way process discovery techniques see the model-extraction task, we can divide them into two broad categories. The first category is constituted by works that tackle the problem of process discovery with one-class supervised learning techniques (see, e.g., (van der Aalst et al. 2010; van der Aalst, Weijters, and Maruster 2004; Günther and van der Aalst 2007; Weijters and van der Aalst 2003; van der Aalst et al. 2017)). These works are driven by the assumption that all available log traces are instances of the process to be discovered and constitute the wast majority of works in the process discovery spectrum. The second category comprises works that intend model-extraction as a two-class supervised task, which is driven by the possibility of partitioning the log traces into two sets according to some business or domain-related criteria. Usually, these sets are referred to as *positive* and *negative* examples, and the goal is to learn a model that characterizes one set w.r.t. the other (see (Chesani et al. 2009; Goedertier et al. 2009; Maruster et al. 2006)). These works are traditionally less represented in the BPM community. Nonetheless, few recent works (Chesani et al. 2021; de León et al. 2018; Slaats, Debois, and Back 2021) have highlighted the importance of performing model-extraction as a two-class supervised task with different motivations: first, the actual existence of *positive* and *negative* examples in real use cases (de León et al. 2018; Slaats, Debois, and Back 2021); second, the need to balance *accuracy* and *recall* (Slaats, Debois, and Back 2021); and third, the need to discover a particular process variant (e.g., the process characterizing "fast" traces) against the one that characterizes other variants, thus using the labels *positive* and *negative* to distinguish between two classes of examples (Chesani et al. 2021). Hereafter, we refer to miners of the first and second category as *unary* and *binary* miners, respectively.

In this paper we recall our recent work on the discovery of process models from positive and negative examples, with the goal of learning a set of declarative constraints that is able to discriminate which trace belongs to which set, also taking into account user preferences on activities and declarative constraint templates to be used to build the final process model. We believe that the grounding on positive and negative examples, which is typical of AI settings and of supervised machine learning (and Inductive Logic Programming in particular), together with the grounding of our approach in a logic-based framework that provides a sound and formal meaning to the notion of user preferences, provides a clear example of bridge between the needs of the BPM domain and the contribution that AI techniques can provide.

The contributions that we have provided are briefly listed in the following, and are contained in (Chesani et al. 2021) (the original framework); (Chesani et al. 2022c,b) (the extension with user preferences); and (Chesani et al. 2022a) (the algorithmic and tool support):

- A novel discovery approach, `NegDis`, based on the underlying logic semantics of Declare, which makes use of the information brought by the positive and negative example sets to produce declarative models.
- The adoption of a satisfiability-based technique to identify the models.
- The introduction of two types of user preferences on the declarative language used to express the model: the first one on the `Declare` patterns to be used in the discovery

task, and the second one on the activities appearing in the output model. Moreover, we discuss also a third type of preference coming from the combination of the first two.
- Heuristics to select the preferred models according to input parameters dealing with (i) generalisation; (ii) simplicity; and (iii) user preferences on the Declare language.
- The exploitation of Answer Set Programming so as to enable the combination of deduction rules within the optimisation algorithm exploited to identify optimal sets of constraints according to preference criteria. This is done in order to take into account not only the user preferences but also the implicit semantics of the formal language.
- An encoding of the process discovery problem using the ASPrin framework for qualitative and quantitative optimisation in ASP.
- An evaluation of the performance of NegDis w.r.t. other relevant works in the same field.

Discovering Business Processes from Positive & Negative Traces

For lack of space we provide here only a description of the basic NegDis binary miner (Chesani et al. 2021). Given two input sets of positive and negative examples, it aims at extracting a model accepting all positive traces and rejecting all negative ones.

NegDis starts from a certain *language bias*: given a set of Declare templates D and a set of activities A, we indicate with $D[A]$ the set of all possible groundings of templates in D w.r.t. A, i.e., all the constraints that can be built using activities in A.

We respectively denote with L^+ and L^- the sets of positive and negative examples in the input event log. NegDis starts by considering a, possibly empty, initial model P, that is a set of Declare constraints known to characterize the examples in L^+. The goal of NegDis is to refine P taking into account both the positive and the negative examples.

Definition 1. Given the initial model P, a candidate solution for the discovery task is any set of constraints $S \subseteq D[A]$ s.t. *(i)* $P \subseteq S$; *(ii)* $\forall t \in L^+$ we have $t \models S$; *(iii)* S maximizes the set $\{t \in L^- \mid t \not\models S\}$.

Declare templates can be organized into a hierarchy of *subsumption* (Di Ciccio et al. 2017) according to the logical implications derivable from their semantics. Consistently with this concept, we introduce the following definition of *generality* relation between models.

Definition 2. A model $M \subseteq D[A]$ is more general than $M' \subseteq D[A]$ (written as $M \succeq M'$) when for any $t \in A^*$, $t \models M' \Rightarrow t \models M$, and strictly more general (written as $M \succ M'$) if M is more general than M' and there exists $t' \in A^*$ s.t. $t' \not\models M'$ and $t' \models M$.

NegDis integrates the *subsumption* rules introduced in (Di Ciccio et al. 2017), into the *deductive closure operator*.

Definition 3. Given a set R of subsumption rules, a deductive closure operator is a function $cl_R : \mathcal{P}(D[A]) \rightarrow \mathcal{P}(D[A])$ that associates any set $M \in D[A]$ with all the constraints that can be logically derived from M by applying one or more deduction rules in R.

For brevity, in the rest of the paper, we will omit the set R and we will simply write $cl(M)$ to indicate the deductive closure of M. The complete set of employed deduction rules is available in the source code (Tessaris, Di Francescomarino, and Chesani 2021).

Conceptually, the NegDis approach can be seen as a two-step procedure: in the first step, a set of candidate constraints is built, and then solutions are selected among subsets of candidates via an optimization algorithm. The set of candidate constraints is composed of those in $D[A]$ that accept all positive examples and reject at least a negative one. To build this set, NegDis constructs a *compatibles* set, i.e., the set of constraints that accept all traces in L^+:

$$compatibles(D[A], L^+) = \{c \in D[A] \mid \forall t \in L^+,\ t \models c\} \tag{1}$$

Then, it defines the *sheriffs* function to associate to any trace t in L^- the constraints of *compatibles* that reject t:

$$\textit{sheriffs}(t) = \{c \in compatibles \mid t \not\models c\} \tag{2}$$

The *sheriffs* function is used to construct the set of all candidate constraints from which a discovered model is derived, i.e., the set $\mathcal{C} = \bigcup_{t \in L^-} \textit{sheriffs}(t)$ of all the constraints in $D[A]$ accepting all positive traces and rejecting at least one negative trace. The solution space is therefore:

$$\mathcal{Z} = \{M \in \mathcal{P}(\mathcal{C}) \mid \forall t \in L^- \ t \not\models M \cup P \text{ or } \textit{sheriffs}(t) = \emptyset\} \tag{3}$$

Due to the fact that not all the pairs of negative and positive sets of traces can be perfectly separated using Declare (Slaats, Debois, and Back 2021), there can be traces in L^- for which the *sheriffs* is empty, meaning that those traces cannot be excluded by any model that guarantees the acceptance of all the positive ones.

The second step of NegDis uses an optimization strategy to identify the solutions; in (Chesani et al. 2021), two different criteria were taken into account: *generality* (or conversely, *specificity*), and *simplicity*. If the user is interested in the most general model, then NegDis employs the closure operator cl to select the models $S \in \mathcal{Z}$ with the less restrictive behavior. If the user wants the simplest model, NegDis looks for the solutions with minimal closure size. In case of ties, the solution with the minimal size is preferred.

Concluding Remarks

The evaluation reported in (Chesani et al. 2021) shows that when compared to state-of-the-art techniques for the *unary* discovery of Declare models, NegDis is able to return a small number of constraints satisfying all traces in L^+ without decreasing the percentage of violated traces in L^-. This is extended, in (Chesani et al. 2022b) to the evaluation of *binary* discovery with preferences using *ASPrin*, a general framework for computing optimal ASP models with preferences. Future works will include a wider evaluation, which will also involve end-users. This will enable the assessment of the potential benefits of involving users (through their preferences) in the loop of process discovery.

References

Chesani, F.; Di Francescomarino, C.; Ghidini, C.; Loreti, D.; Maggi, F. M.; Mello, P.; Montali, M.; and Tessaris, S. 2021. Process discovery on deviant traces and other stranger things. *IEEE Trans. on Knowledge and Data Engineering*. Under review.

Chesani, F.; Francescomarino, C. D.; Ghidini, C.; Grundler, G.; Loreti, D.; Maggi, F. M.; Mello, P.; Montali, M.; and Tessaris, S. 2022a. Optimising Business Process Discovery Using Answer Set Programming. In *Logic Programming and Nonmonotonic Reasoning - 16th International Conference, LPNMR 2022, Genova, Italy, September 5-9, 2022, Proceedings*, volume 13416 of *Lecture Notes in Computer Science*, 498–504. Springer.

Chesani, F.; Francescomarino, C. D.; Ghidini, C.; Grundler, G.; Loreti, D.; Maggi, F. M.; Mello, P.; Montali, M.; and Tessaris, S. 2022b. Shape Your Process: Discovering Declarative Business Processes from Positive and Negative Traces Taking into Account User Preferences. In *Enterprise Design, Operations, and Computing - 26th International Conference, EDOC 2022, Bozen-Bolzano, Italy, October 3-7, 2022, Proceedings*, volume 13585 of *Lecture Notes in Computer Science*, 217–234. Springer.

Chesani, F.; Francescomarino, C. D.; Ghidini, C.; Loreti, D.; Maggi, F. M.; Mello, P.; Montali, M.; Palmieri, E.; and Tessaris, S. 2022c. Discovering Business Processes models expressed as DNF or CNF formulae of Declare constraints. In *Proceedings of the 37th Italian Conference on Computational Logic, Bologna, Italy, June 29 - July 1, 2022*, volume 3204 of *CEUR Workshop Proceedings*, 201–216. CEUR-WS.org.

Chesani, F.; Lamma, E.; Mello, P.; Montali, M.; Riguzzi, F.; and Storari, S. 2009. Exploiting Inductive Logic Programming Techniques for Declarative Process Mining. *Trans. Petri Nets Other Model. Concurr.*, 2: 278–295.

de León, H. P.; Nardelli, L.; Carmona, J.; and vanden Broucke, S. K. L. M. 2018. Incorporating negative information to process discovery of complex systems. *Inf. Sci.*, 422: 480–496.

Di Ciccio, C.; Maggi, F. M.; Montali, M.; and Mendling, J. 2017. Resolving inconsistencies and redundancies in declarative process models. *Inf. Syst.*, 64: 425–446.

Goedertier, S.; Martens, D.; Vanthienen, J.; and Baesens, B. 2009. Robust Process Discovery with Artificial Negative Events. *J. Mach. Learn. Res.*, 10: 1305–1340.

Günther, C. W.; and van der Aalst, W. M. P. 2007. Fuzzy Mining - Adaptive Process Simplification Based on Multi-perspective Metrics. In *BPM*, volume 4714 of *Lecture Notes in Computer Science*, 328–343. Springer.

Maruster, L.; Weijters, A. J. M. M.; van der Aalst, W. M. P.; and van den Bosch, A. 2006. A Rule-Based Approach for Process Discovery: Dealing with Noise and Imbalance in Process Logs. *Data Min. Knowl. Discov.*, 13(1): 67–87.

Slaats, T.; Debois, S.; and Back, C. O. 2021. Weighing the Pros and Cons: Process Discovery with Negative Examples. In Polyvyanyy, A.; Wynn, M. T.; Looy, A. V.; and Reichert, M., eds., *Business Process Management - 19th International Conference, BPM 2021*, volume 12875, 47–64.

Tessaris, S.; Di Francescomarino, C.; and Chesani, F. 2021. Negdis: code for the experiments.

van der Aalst, W. M. P.; De Masellis, R.; Di Francescomarino, C.; and Ghidini, C. 2017. Learning Hybrid Process Models from Events - Process Discovery Without Faking Confidence. In *Business Process Management - 15th International Conference, BPM 2017, Barcelona, Spain, September 10-15, 2017, Proceedings*, volume 10445 of *Lecture Notes in Computer Science*, 59–76. Springer.

van der Aalst, W. M. P.; Rubin, V. A.; Verbeek, H. M. W.; van Dongen, B. F.; Kindler, E.; and Günther, C. W. 2010. Process mining: a two-step approach to balance between underfitting and overfitting. *Software and Systems Modeling*, 9(1): 87–111.

van der Aalst, W. M. P.; Weijters, T.; and Maruster, L. 2004. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Trans. Knowl. Data Eng.*, 16(9): 1128–1142.

van der Aalst, et al. 2012. Process Mining Manifesto. In Daniel, F.; Barkaoui, K.; and Dustdar, S., eds., *Business Process Management Workshops*, 169–194. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN 978-3-642-28108-2.

Weijters, A. J. M. M.; and van der Aalst, W. M. P. 2003. Rediscovering workflow models from event-based data using little thumb. *Integr. Comput. Aided Eng.*, 10(2): 151–162.

Acknowledgments

This work has been partially supported by the European Union's H2020 projects HumaneAI-Net (g.a. 952026), StairwAI (g.a. 101017142), and TAILOR (g.a. 952215) and by MUR under PRIN project PINPOINT Prot. 2020FNEB27, CUP H23C22000280006 and H45E21000210001.