

SMT-based Safety Verification of Data-Aware Processes: Foundations and Applications (Extended Abstract)

Alessandro Gianola^{1,*}

¹Free University of Bozen-Bolzano, Bolzano, Italy

Abstract

Integrating data and processes to understand their concrete interplay is a long-standing problem in business process management. We present a general framework for the formalization and verification of infinite-state transition systems that can interact with a persistent storage represented as relational databases. We develop sophisticated algorithmic techniques, based on automated reasoning and SMT solving, for the verification of *safety* properties. Building on top of these techniques, we apply our framework to business process management: we introduce general models for the safety verification of complex business processes enriched with real data. For those models, we devise formal and operational methods that are based on standard languages and/or can capture advanced modeling capabilities, considering in particular the BPMN standard and variants of Petri nets for the process control-flow, and SQL queries and updates for the interaction with data.

1. Overview of the Problem

The thesis of our work is that principles, methods and techniques from automated reasoning and Satisfiability Modulo Theories (SMT) can be effectively employed to lay solid foundations and to develop concrete tools for the *formalization* and *verification* of infinite-state systems arising from the interplay between the *control flow* of a *business process* and the *data* it queries and manipulates.

Finite-State Model Checking. *Formal verification* has the goal of providing automated methods for verifying complex systems against some property of interest. Since the eighties, *model checking* [1] has gained increasing attention, becoming one of the most celebrated approaches. It consists of exhaustively and automatically checking whether the (abstracted) models of interest meet some given temporal specification. Traditional model checking relies on the assumption that the systems can be abstractly described using *finitely many states*, so as to exploit *explicit* search procedures in the *finite* space of all configurations. In many software contexts, where the systems are only considered from the point of view of the control-flow, sophisticated techniques for a smart exploration of the state space guarantee that these approaches are successful [2].

BPM 2022: Best BPM Dissertation Award 2022, September 11–16, 2022, Münster, Germany

*Corresponding author.


✉ gianola@inf.unibz.it (A. Gianola)

🌐 <https://gianola.people.unibz.it/> (A. Gianola)

🆔 0000-0003-4216-5199 (A. Gianola)



© 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)

Data-Aware Processes. In recent years, a growing number of application domains asks for process modeling languages paired with automated verification techniques that do not just consider the control flow dimension, but also take into consideration the data dimension. From a *theoretical perspective*, the development of formal, abstract frameworks for attacking the problem of verifying so-called *Data-Aware Processes (DAPs)* has consequently flourished [3, 4], leading to a plethora of formal models that differentiate depending on how the data and process components, as well as their interplay, is actually represented. What all these frameworks have in common is that they strive to focus on very general DAP models that formalize abstract dynamic systems (i.e., the *process* component) interacting with data persistent storage (i.e., the *data* component). In these models, the concept of “process” should be interpreted in an abstract sense, as a (possibly undetermined) mechanism that guides the evolution of a system.

One may wonder whether formal frameworks for verifying DAPs can employ the techniques used in finite-state model checking. Verification of DAPs should reflect the possibility of expressing properties that simultaneously account for the data and the process perspectives, and most importantly for their interaction. However, due to the presence of data, DAP models are *intrinsically infinite-state*: the content of a relational database is *finite* but its size is *unbounded* and *unknown* a priori (since a new tuple can always be added to some relation using data elements taken from infinite domains); in addition, many real-world scenarios often require the presence of possibly fresh data values (e.g., string, new identifiers, integer or real numbers) injected into the process by external users. In such sophisticated settings, explicit model checking is not possible anymore: due to the infinity of the state space, exhaustive search cannot be applied. Moreover, traditional approaches in data management do not help here, since they typically investigate *static*, structural aspects of a domain of interest, disregarding dynamics [5, 6]. The results obtained for the verification of DAPs are quite fragmented, since they consider a variety of different assumptions on the model and on the analysis tasks [3, 4], assumptions that are usually too restrictive when compared with both concrete business process and data management models.

Business Processes enriched with Data. While theoretical DAP frameworks were studied, a huge body of research has been dedicated to the challenging problem of reconciling data and business process management within contemporary organizations [7, 8, 9]. More specifically, in the BPM context it has become more and more important to study multi-perspective models that do not just conceptually account for the control-flow dimension of business processes, but also consider the interplay with data [3]: in contrast with abstract DAPs, these models are more focused on concrete processes as they are interpreted by stakeholders and BPM practitioners. Traditionally, BPM models are formalized using pure control-flow models, ignoring the effects caused by data to the evolution of the process, i.e., neglecting how data are manipulated during the tasks execution, and how the executability of tasks is affected by data-related conditions.

To overcome this fundamental limitation, many concrete languages and corresponding formalisms have been proposed to represent business processes where the data and the control-flow dimension are both considered as first-class citizens [10]. One important challenge that naturally arises is how to formally verify such business processes enriched with data: in this respect, it seems natural to employ the abstract frameworks developed for DAPs, and to try to adapt these frameworks to the desired application domains (see for some attempts in this

direction, e.g., [11, 12]). However, this adaptation is quite challenging because of the lack of a comprehensive framework for the verification of DAPs that is able to capture most of the essential modeling features of business process data-aware extensions. The solutions provided in the literature consist either of abstracting from data to non-determinism or of bounding data to only finitely many configurations, fixed apriori. Nonetheless, these solutions are partial since only support very limited types of unboundedness, whereas real data, in general, account for two more complex types: the one due to fresh values injected into the process from the environment, and the one due to how relations (the most basic data structures) can yield new configurations. These, in turn, can easily lead to Turing-complete mechanisms.

Moreover, currently there are few theoretical frameworks that also come with a corresponding implemented tool for concrete verification, which is one of the desiderata for every model checking application domain. Finally, an orthogonal challenging dimension regards the abstract flavor of these frameworks and their modeling constructs, which often depart from those offered by process and data modeling standard languages used in practice, such as the Business Process Model and Notation (BPMN) for processes, and the Structured Query Language (SQL) for persistent data. In fact, it is essential to rely on standard languages when dealing with real applications, considering that model checking should have a big impact in guaranteeing trustworthiness of concretely deployed systems.

Infinite-state Model Checking. In parallel, in the nineties, a completely separated line of research started to investigate how to attack the problem of verifying *infinite-state systems* in general, disregarding the data perspective. A particularly interesting class of infinite-state systems is given by *parameterized* systems [13]. Such systems can be seen as transition systems that are *parametric* in the number n of (possibly independent) components, called *parameters*. *Parameterized verification* approaches provide automatic methods to prove trustworthiness of these systems regardless of the number of their components.

Another successful way for tackling infinite-state model checking consists of providing a *symbolic setting* [14] for representing states. Thanks to this representation, the exploration of the state space does not need to be *explicit*, but it can be performed *symbolically*. Satisfiability Modulo Theories (SMT) [15] is one of the most fruitful emerging areas in computational logic that provides techniques for symbolic model checking: the SMT problem consists of deciding the satisfiability of logical formulae with respect to combinations of background first-order logic theories. SMT leverages highly efficient technologies, such as those stemming from SMT solvers, and has been proved to be particularly successful for verifying infinite-state systems. There exists a wide range of methods using SMT solvers as their underlying engine. A successful framework based on SMT solving (and applied, e.g., to distributed systems) is the one of *array-based* systems [16], a declarative formalism where arrays (as known from programming languages) are the central notion.

2. Contributions

As we have seen, DAP verification has recently gained increasing attention, with strong motivation from BPM applications. Several approaches have been proposed for attacking this problem, but they present two significant limitations. First, the employed verification techniques are

developed *ad hoc*, and do not rely on *well-established methods* and *implemented and state-of-the-art technologies*. Second, these frameworks are usually studied at a quite abstract level, which makes it challenging to connect them to the standard front-end languages used in practice. To summarize, there is the lack of a *universal* and *comprehensive* approach to verify DAPs via a sufficiently expressive formalism that natively supports effective techniques and can encode “front-end” languages and models.

At the same time, many approaches to symbolic reasoning for *infinite-state systems* have been successfully applied to different models, such as parameterized systems. Most of these approaches provide not only a solid theoretical framework, but also the support of computationally efficient and highly engineered tools. This is the case of SMT solvers and model checkers based on them. Nevertheless, importing SMT-based model checking in the context of DAP verification is extremely challenging and cannot be done *as is*: it requires carrying out genuinely novel research for handling the data perspective, and to develop algorithmic techniques. In this work, *we bridge the gap between DAP verification and infinite-state model checking*: (i) we propose a **theoretical SMT-based framework** for DAP verification and, to make it effective in practice, we devise efficient **algorithmic techniques** based on automated reasoning; (ii) we apply the previous machinery to provide **practical approaches** for modeling and verifying business processes enriched with real data.

DAP Verification via SMT. We define a *general model-theoretic/algebraic framework*, called (Universal) RASs, for automated verification of DAPs based on SMT solving, focusing in particular on *parameterized safety*: this amounts to establishing whether a system can reach an undesired, unsafe configuration, irrespective of the specific content of its background data storage. Our approach relies on array-based systems: we incorporate the needed capabilities to formalize and reason about relational databases. We exhibit how arrays can represent read-write relations that can be queried and updated during the evolution of the system.

For this purpose, we develop *sophisticated algorithmic techniques* building on top of well-known methods in automated reasoning. We also demonstrate the feasibility of our approach by showing these techniques in action thanks to the implementation in the state-of-the-art MCMT model checker, testing it against a concrete DAP benchmark: in this respect we provide a first, preliminary experimental evaluation of our verification machinery.

Modeling and Verification of Data-Aware Business Processes. As a second, key contribution, we exploit the developed machinery to provide *BPM-oriented practical approaches*, strictly linked to other major research lines on modeling and verifying processes with data.

First, within the general framework of RASs, we propose Data-Aware BPMN models, called DABs, based on standard languages used in the BPM literature. These models natively combine process modeling and querying languages used in practice (BPMN+SQL) in terms of a significant subset of this combination, exhibiting how this subset can be both automatically translated into the formal verification framework of RASs. These models are implemented and executed in an operational framework called delta-BPMN, where a standard process execution engine (provided by Camunda) is enriched with data management features, i.e., with the support of relational databases and data manipulation capabilities.

Then, we introduce a theoretical formalism, called COA-nets, that captures expressive modeling capabilities in the recently born spectrum of so-called object-centric processes. COA-nets are

in line with process modeling principles studied in process research, specifically Petri nets and their extensions, and represent an extension of Colored Petri nets that is able both to model data flow in tokens and to access a relational database. Once again, the translation to the verification framework of Universal RASs is provided. We explicitly discuss ways of modeling process and data interactions along a comprehensive setting, where expressive modeling patterns are introduced and shown how to be employed in practice without hampering verification guarantees.

References

- [1] E. M. Clarke, T. A. Henzinger, H. Veith, R. Bloem (Eds.), *Handbook of Model Checking*, Springer, 2018.
- [2] J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, L. J. Hwang, Symbolic model checking: 10^{20} states and beyond, *Inf. Comput.* 98 (1992) 142–170.
- [3] D. Calvanese, G. De Giacomo, M. Montali, Foundations of data-aware process analysis: A database theory perspective, in: *Proc. of PODS 2013*, 2013, pp. 1–12.
- [4] A. Deutsch, R. Hull, Y. Li, V. Vianu, Automatic verification of database-centric systems, *ACM SIGLOG News* 5 (2018) 37–56.
- [5] S. Abiteboul, R. Hull, V. Vianu, *Foundations of Databases*, Addison-Wesley, 1995.
- [6] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider (Eds.), *The Description Logic Handbook: Theory, Implementation, and Applications*, Cambridge University Press, 2003.
- [7] C. Richardson, Warning: Don't assume your business processes use master data, in: *Proceedings of BPM 2010*, volume 6336 of *LNCS*, Springer, 2010, pp. 11–12.
- [8] M. Dumas, On the convergence of data and process engineering, in: *Proc. of ADBIS 2011*, 2011, pp. 19–26.
- [9] M. Reichert, Process and data: Two sides of the same coin?, in: *Proceedings of OTM 2012*, volume 7565 of *LNCS*, Springer, 2012, pp. 2–19.
- [10] A. Meyer, L. Pufahl, D. Fahland, M. Weske, Modeling and enacting complex data dependencies in business processes, in: *Proc. of BPM 2013*, 2013, pp. 171–186.
- [11] E. Damaggio, A. Deutsch, R. Hull, V. Vianu, Automatic verification of data-centric business processes, in: *Proc. of BPM 2011*, 2011, pp. 3–16.
- [12] D. Solomakhin, M. Montali, S. Tessaris, R. De Masellis, Verification of artifact-centric systems: Decidability and modeling issues, in: *Proc. of ICSOC 2013*, 2013, pp. 252–266.
- [13] P. A. Abdulla, G. Delzanno, Parameterized verification, *Int. J. Softw. Tools Technol. Transf.* 18 (2016) 469–473.
- [14] K. L. McMillan, *Symbolic Model Checking*, Kluwer Academic Publishers, Norwell, MA, USA, 1993.
- [15] C. W. Barrett, C. Tinelli, Satisfiability modulo theories, in: *Handbook of Model Checking*, 2018, pp. 305–343.
- [16] S. Ghilardi, S. Ranise, Backward reachability of array-based systems by SMT solving: Termination and invariant synthesis, *Log. Methods Comput. Sci.* 6 (2010).