

Verification of Data-Aware Processes: Challenges and Opportunities for Automated Reasoning

Diego Calvanese¹, Silvio Ghilardi², Alessandro Gianola¹, Marco Montali¹, and
Andrey Rivkin¹

¹ Free University of Bozen-Bolzano, Bolzano, Italy
`surname@inf.unibz.it`

² Università degli Studi di Milano, Milan, Italy
`silvio.ghilardi@unimi.it`

Abstract

We briefly introduce the line of research on the verification of data-aware processes, with the intention of raising more awareness of it within the automated reasoning community. On the one hand, data-aware processes constitute a concrete setting for validating and experimenting with automated reasoning techniques. On the other hand, they trigger new genuine research challenges for researchers in automated reasoning.

1 Introduction

Contemporary organizations rely more and more on business processes to describe, analyze, and regulate their internal work. Business process management (BPM) is now a well-assessed discipline at the intersection between operations management, computer science, and IT engineering. Its grand goal is to support managers, analysts, and domain experts in the design, deployment, enactment, and continuous improvement of processes [?].

One of the essential concepts in BPM is that of a *process model*. A process model explicitly describes which tasks have to be performed within the organization (such as *check order*) in response to external events (such as *receive order request*), and what are the allowed courses of execution (such as *deliver order* can only be executed if *check order* has been successfully completed). Several process modeling languages have been proposed for this purpose, such as BPMN [?], UML Activity Diagrams [?], and EPCs [?]. Verification and automated reasoning techniques are in this respect instrumental to formally analyze process models and ascertain their correctness before their actual deployment into corresponding BPM systems.

Traditionally, formal analysis of process models is limited to the process control flow, represented using variants of bounded Petri nets or finite-state transition systems (depending on how concurrency is interpreted). This, however, does not reflect the intrinsic, multi-perspective nature of processes and their corresponding models. In particular, process tasks are executed by *resources* based on *decisions* that depend on background and process-related *data*, in turn manipulated upon task execution.

In this multi-perspective spectrum, the last two decades have seen a huge body of research dedicated to the integration of *data* and *process* management to achieve a more comprehensive understanding on how data influence behavior, and how behavior impact data [?, ?, 27].

The corresponding development of formal frameworks for the verification of data-aware processes has consequently flourished, leading to a wide plethora of formal models depending on how the data and process components, as well as their interplay, is actually represented.

One stream of research followed the artifact-centric paradigm, where the main focus is that of persistent business objects (such as orders or loans) and their lifecycle [32, 2]. Here, variants of the same model are obtained depending on how such business objects are represented. Notable examples are: (i) relational data with different kinds of constraints [13, ?, 21], (ii) relational data with numerical values and arithmetics [?, 14], (iii) tree-structured data [1]. Also more minimalistic models have been brought forward, capturing data-aware processes as a persistent data storage evolved through the application of (conditional) actions that may inject external, possibly fresh values through service calls reminiscent of uninterpreted functions. Two variants of this model have been studied, the first considering persistent relational data with constraints [?, ?], the second operating over description logic knowledge bases whose extensional data are interpreted under incomplete information, and updated in the style of Levesque functional approach [?, ?].

Another stream of research followed instead the more traditional activity-centric approach, relying on Petri nets as the underlying control-flow backbone of the process. Specifically, Petri net-based models have been enriched with: (i) data items locally carried by tokens [30, 20], (ii) data registers with numerical and non-numerical values [11], (iii) tokens carrying tree-structured data [?], and/or (iv) persistent relational data manipulated with the full power of FOL/SQL [12, 24].

Last but not least, the interplay between data and processes has been studied to build solid foundations for “many-to-many” processes, that is, processes whose tasks co-evolve multiple different objects related to each other (such as e-commerce companies where each order may correspond to multiple shipped packages, and each package may contain items from different orders). Implicit (data-driven) [?] and explicit (token-driven) [?] coreference and synchronization mechanisms have been proposed for this purpose.

On top of these formal models, several verification tasks have been studied. On the one hand, they consider different types of properties, ranging from fundamental properties such as reachability, safety, and soundness, to sophisticated formulae expressed in linear- and branching-time first-order temporal logics [2]. On the other hand, they place different assumptions regarding how data can be manipulated, and whether there are read-only data whose configuration is not known. The resulting verification problems are all undecidable in general, and require to properly tame the infinity arising from the presence of data.

All in all, we believe this wide spectrum of verification problems constitutes an extremely interesting application area for automated reasoning techniques. On the one hand, data-aware processes constitute a concrete setting for experimenting symbolic techniques developed within automated reasoning, so as to enable reasoning on the evolution of data without explicitly representing them. In addition, given the applied flavor of BPM, the feasibility of assumptions and conditions imposed towards guaranteeing good computational properties (such as decidability or tractability) can be assessed in the light of end user-oriented modeling languages and their corresponding modeling methodologies. On the other hand, data-aware processes trigger new, genuine research challenges for researchers in automated reasoning, arising from the subtle, yet necessary interplay between control-flow aspects and volatile and persistent data with constraints.

To substantiate this claim, we briefly describe next one particular verification problem where automated reasoning techniques are proving very successful.

2 The Success Story of Relational Artifact Systems

Artifact systems formalize data-aware processes using three main components: (i) a read-only database that stores fixed, background information; (ii) a working memory that stores the evolving state of artifacts throughout their lifecycle; (iii) actions that inspect the read-only memory and the working memory, and consequently update the working memory. Different variants of this model, obtained via a careful tuning of the relative expressive power of its three components, have been studied towards decidability of verification problems parameterized over the read-only database (see, e.g., [13, ?, 1, 14, 6, 5]). These are verification problems where a property is checked for every possible configuration of the read-only database, thus guaranteeing that the overall process operates correctly no matter how the read-only data are instantiated.

In the most recent variants of this model, the read-only database is equipped with key and foreign key constraints relating the content of different relations. At the same time, the working memory is relational, with each relation representing an artifact, in principle capable of storing unboundedly many tuples denoting instances of that artifact [14, 21].

In [6], we took inspiration from this approach, studying the model of so-called *relational artifact systems* (RASs). Notably, we connected RASs to the well-established model of array-based systems within the SMT tradition [15]. This is done in two steps. First, the schema of a read-only database is represented in functional, algebraic fashion, where relations and constraints are captured using multiple sorts and unary functions. Second, each artifact relation within the working memory is treated as a set of arrays, where each array accounts for one component of the corresponding artifact relation. A tuple (i.e., artifact instance) in an artifact relation is then reconstructed by accessing all such arrays with the same index.

With these notions at hand, from a logical point of view the behavior of a RAS is specified via: (i) second order variables for artifacts components; (ii) first order variables for “data”, ranging both on the sorts of the read-only database and on numerical (real, integer) domains. Thus, suitable combinations of (linear) arithmetics and EUF can be employed for reasoning about RAS systems. Non-determinism in system evolution is captured via first-order parameters, that is, further existentially quantified variables occurring in transition formulae, whereas second-order variables updates are functionally determined by such non-determinism at the first-order level.

On top of this formal model, we have in particular studied model checking of safety properties, starting from the symbolic backward reachability approach implemented in the MCMT model checker [15, 16] and extending it towards native support for RASs. In our approach, sets of system states are specified via formulae constraining both artifact components and data variables; such constraints may again include further existentially quantified parameter variables. From the point of view of model checking search, the main problem is that of avoiding the existential prefix to grow in an uncontrolled way, thus some form of symbol elimination is needed. This is rather easily achieved for second-order variables, at least if backward search is employed (because, as recalled before, there updates are functional modulo first order parameters). However, it is non trivial at all at the first-order level. Numerical existentially quantified variables can be eliminated via well-known methods (predicate abstraction [?], interpolation [?, ?], model elimination [19, ?], or even quantifier elimination), however for variables ranging over the read-only database our suggestion is to employ a different technique, namely cover computation [18].

Thanks to cover computation, we overcome the fact that quantifier elimination is not applicable to data variables pointing to elements of the read-only database. In particular, the computation of covers is nothing but quantifier elimination in the model completions of the theory used to capture the schema and constraints of the read-only database schema, as shown in [5]. The idea of using model completions when quantifier elimination is not available is present also in [?]. Interestingly, differently from quantifier elimination in linear arithmetics, cover computation in the restricted “unary” case required for RASs turns out to be tractable [18, 5]. Tractability has been also confirmed by our initial experiments with the latest version of MCMT, which handles RASs natively. In particular, we have used RASs as a basis for formalizing a data-aware extension of the de-facto process modeling standard BPMN [?], and used the resulting approach to conduct an initial benchmark using some process models from [21], with very encouraging results [6, ?].

To sum up, we believe that by employing both well-established and relatively new techniques, the automated reasoning community is ready to face the challenges raised by the emerging area of verification of data-aware processes, providing foundational, algorithmic, and applied advancements.

References

- [1] W.M.P. van der Aalst. Formalization and Verification of Event-driven Process Chains. *Information and Software Technology*, 41(10):639–650, 1999.
- [2] P. A. Abdulla, C. Aiswarya, M. F. and M. Montali Atig, and O. Rezine. Recency-bounded verification of dynamic database-driven systems. In *Proc. of PODS*. ACM Press, 2016.
- [3] A. Artale, A. Kovtunova, M. Montali, and W. M. P. van der Aalst. Modeling and reasoning over declarative data-aware processes with object-centric behavioral constraints. In *Proc. BPM*. Springer, 2019. To appear.
- [4] E. Badouel, L. Hélouët, and C. Morvan. Petri nets with structured data. *Fundam. Inform.*, 146(1):35–82, 2016.
- [5] B. Bagheri Hariri, D. Calvanese, G. De Giacomo, A. Deutsch, and M. Montali. Verification of relational data-centric dynamic systems with external services. In *Proc. of PODS*, 2013.
- [6] F. Belardinelli, A. Lomuscio, and F. Patrizi. An abstraction technique for the verification of artifact-centric systems. In *Proc. of KR*, 2012.
- [7] M. Bojańczyk, L. Segoufin, and S. Toruńczyk. Verification of database-driven systems via amalgamation. In *Proc. of PODS*, pages 63–74, 2013.
- [8] D. Calvanese, G. De Giacomo, and M. Montali. Foundations of data aware process analysis: A database theory perspective. In *Proc. of PODS*, 2013.
- [9] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, and A. Rivkin. Formal modeling and SMT-based parameterized verification of data-aware BPMN. In *Proc. BPM*. Springer, 2019. To appear.
- [10] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, and A. Rivkin. Model completeness, covers and superposition. In *Proc. of CADE*, 2019.
- [11] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, and A. Rivkin. From model completeness to verification of data aware processes. In *Description Logic, Theory Combination, and All That*. Springer, 2019. To appear.
- [12] D. Calvanese, M. Montali, and A. Santoso. Verification of generalized inconsistency-aware knowledge and action bases. In *Proc. IJCAI*. AAAI Press, 2015.
- [13] E. Damaggio, A. Deutsch, and V. Vianu. Artifact systems with data dependencies and arithmetic. *ACM TODS*, 37(3), 2012.

- [14] M. de Leoni, P. Felli, and M. Montali. A holistic approach for soundness verification of decision-aware process models. In *Proc. ER*, pages 219–235, 2018.
- [15] R. De Masellis, C. Di Francescomarino, C. Ghidini, M. Montali, and S. Tessaris. Add data into business process verification: Bridging the gap between theory and practice. In *Proc. of AAAI*. AAAI Press, 2017.
- [16] A. Deutsch, R. Hull, F. Patrizi, and V. Vianu. Automatic verification of data-centric business processes. In *Proc. of ICDT*, pages 252–267, 2009.
- [17] A. Deutsch, Y. Li, and V. Vianu. Verification of hierarchical artifact systems. In *Proc. of PODS*, pages 179–194. ACM Press, 2016.
- [18] M. Dumas. On the convergence of data and process engineering. In *Proc. of ADBIS*, volume 6909 of *LNCS*. Springer, 2011.
- [19] M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers. *Fundamentals of Business Process Management*. Springer, 2013.
- [20] D. Fahland. Describing behavior of processes with many-to-many interactions. In *Proc. PETRI NETS*, volume 11522 of *LNCS*, pages 3–24. Springer, 2019.
- [21] C. Flanagan and S. Qadeer. Predicate abstraction for software verification. In *POPL*, pages 191–202, 2002.
- [22] S. Ghilardi, E. Nicolini, S. Ranise, and D. Zucchelli. Towards SMT model checking of array-based systems. In *Proc. of IJCAR*, pages 67–82, 2008.
- [23] S. Ghilardi and S. Ranise. Backward reachability of array-based systems by SMT solving: Termination and invariant synthesis. *Logical Methods in Computer Science*, 6(4), 2010.
- [24] Object Management Group. Omg unified modeling language 2.5, 2013. <http://www.omg.com/uml/>.
- [25] S. Gulwani and M. Musuvathi. Cover algorithms and their combination. In *Proc. of ESOP, Held as Part of ETAPS*, pages 193–207, 2008.
- [26] B. Bagheri Hariri, D. Calvanese, G. De Giacomo, R. De Masellis, P. Felli, and M. Montali. Verification of description logic knowledge and action bases. In *Proc. ECAI*, pages 103–108, 2012.
- [27] K. Hoder and Nikolaj Bjørner. Generalized property directed reachability. In *Proc. of SAT*, pages 157–171, 2012.
- [28] L. Kovács and A. Voronkov. Interpolation and symbol elimination. In *Proc. of CADE*, pages 199–213, 2009.
- [29] S. Lasota. Decidability border for petri nets with data: WQO dichotomy conjecture. In *Proc. PETRI NETS*, volume 9698 of *LNCS*, pages 20–36. Springer.
- [30] Y. Li, A. Deutsch, and V. Vianu. VERIFAS: A practical verifier for artifact systems. *PVLDB*, 11(3):283–296, 2017.
- [31] K.L. McMillan. Lazy Abstraction with Interpolants. In *CAV*, 2006.
- [32] M. Montali and A. Rivkin. DB-Nets: on the marriage of colored Petri Nets and relational databases. *TOPNOC*, 28(4), 2017.
- [33] OMG. *Business Process Model and Notation (BPMN) - Version 2.0, Beta 1*. 2009.
- [34] O. Padon, N. Immerman, S. Shoham, A. Karbyshev, and M. Sagiv. Decidability of inferring inductive invariants. In *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016, St. Petersburg, FL, USA, January 20 - 22, 2016*, pages 217–231, 2016.
- [35] M. Reichert. Process and data: Two sides of the same coin? In *Proc. of the On the Move Confederated Int. Conf. (OTM 2012)*, volume 7565 of *LNCS*. Springer, 2012.
- [36] C. Richardson. Warning: Don’t assume your business processes use master data. In *Proc. of BPM*, volume 6336 of *LNCS*. Springer, 2010.
- [37] F. Rosa-Velardo and D. de Frutos-Escrig. Decidability and complexity of petri nets with unordered

- data. *Theor. Comput. Sci.*, 412(34):4439–4451, 2011.
- [38] V. Sofronie-Stokkermans. On interpolation and symbol elimination in theory extensions. In *Proc. IJCAR*, Lecture Notes in Computer Science. Springer, 2016.
- [39] V. Vianu. Automatic verification of database-driven systems: a new frontier. In *Proc. of ICDT*, pages 1–13, 2009.