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

Diego Calvanese
Faculty of Computer Science
Free University of Bozen-Bolzano, Bolzano, Italy
calvanese@inf.unibz.it

Silvio Ghilardi
Dipartimento di Matematica
Università degli Studi di Milano, Milan, Italy
silvio.ghilardi@unimi.it

Alessandro Gianola
Faculty of Computer Science
Free University of Bozen-Bolzano, Bolzano, Italy
gianola@inf.unibz.it

Marco Montali
Faculty of Computer Science
Free University of Bozen-Bolzano, Bolzano, Italy
montali@inf.unibz.it

Andrey Rivkin
Faculty of Computer Science
Free University of Bozen-Bolzano, Bolzano, Italy
rivkin@inf.unibz.it

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 [21].

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 [35], UML Activity Diagrams [26], and EPCs [1]. 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.

To appear in EPTCS.

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 [38, 20, 37].

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 [41, 8]. 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 [18, 6, 32], *(ii)* relational data with numerical values and arithmetics [15, 19], *(iii)* tree-structured data [7]. 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 [5, 2], 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 [28, 14].

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 [39, 31], *(ii)* data registers with numerical and non-numerical values [16], *(iii)* tokens carrying tree-structured data [4], and/or *(iv)* persistent relational data manipulated with the full power of FOL/SQL [17, 34].

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) [3] and explicit (token-driven) [22] 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, soundness and liveness, to sophisticated formulae expressed in linear- and branching-time first-order temporal logics [8]. 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 very promising.

2 The Concrete Case 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., [18, 15, 7, 19, 11, 12, 9]). 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 [19, 32].

In [11], 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 [24]. This is done in two steps. First, the schema of a read-only database is represented in a 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 the top of this formal model, various problems arise that can be effectively attacked using techniques and solutions within the automated reasoning community in general, and the SMT community in particular. We briefly discuss next some of them.

1. By focusing on model checking of safety properties via symbolic backward reachability [24, 25], the main problem is that of avoiding the existential prefix to grow in an uncontrolled way. This, in turn, calls for some form of symbol elimination. This is rather easily achieved for second-order variables – at least when backward search is employed – because, as mentioned above, updates are often functional modulo first-order parameters; however, it is not clear what happens if alternative search strategies are employed, or other relevant properties are checked. At the first-order level, specific challenges instead arise already in this setting. In fact, while numerical existentially quantified variables can be eliminated via well-known methods (such as predicate abstraction [23], interpolation [33, 30], model elimination [29, 36], or even quantifier elimination), the manipulation of variables ranging over the read-only database appears to require completely different techniques, like cover computation [27].

Thanks to cover computation, one can in particular overcome the fact that quantifier elimination is not directly applicable to variables pointing to elements of the read-only database. More technically, the computation of covers is nothing but quantifier elimination in the model completions of the theory used to capture the schema and the constraints of the read-only database schema, as shown in [12]. The idea of using model completions when quantifier elimination is not directly available is present also in [40]. Notably, differently from quantifier elimination in linear arithmetics, cover computation in the restricted "unary" case required for RASs turns out to be tractable [27, 12].

2. Different types of properties are usually required to be verified in the context of data-aware processes, where safety is one of the most typical. Nevertheless, a comprehensive research dedicated to SMT-based techniques for the effective verification of data-aware processes should also consider richer forms of verification going beyond safety (e.g., liveness and fairness), and richer classes of artifact systems incorporating concrete data types and arithmetic operations that should explicitly appear in the specification language.
3. Database instances are typically built on top of *finite* structures (although they may contain elements from "value" sorts ranging over infinite domains). Depending on the specific setting under investigation, this feature may require to introduce specific techniques from finite model theory. In particular, advanced applications will presumably require: from the foundational perspective, to investigate suitable versions of the finite model property; from the applied perspective, to integrate common solvers with model finders.
4. Interesting variants of RASs arise when the data stored therein are interpreted under incomplete information, and in the presence of complex ontological constraints expressing background, structural knowledge of the organisational domain. Transferring model checking techniques such as those recalled in point 1 above to this richer setting is not at all trivial, as reasoning must now be carried out tackling two dimensions at once: the temporal dimension along which the artifact systems evolves, and the structural dimension constraining the manipulated data objects and their mutual relationships.
5. Towards enabling the concrete exploitation of verification techniques, logic-based formalisms used to formalize RASs or other types of data-aware processes need to be connected to end user-oriented process modeling languages. This interconnection paves the way towards practical reasoning tasks that are relevant for end users, but have not yet addressed by the automated reasoning community. In addition, by considering specific modeling guidelines and methodologies, interesting subclasses of general formal models such as that of RASs may naturally emerge. It would be then important to assess whether such subclasses come with interesting computational guarantees for the corresponding automated reasoning tasks.

We tried to address only some of the most simple problems from the above list. In particular, we have used RASs as a basis for formalizing a data-aware extension of the de-facto process modeling standard BPMN [10], and used the resulting approach to conduct an initial benchmark using some process models from [32], with very encouraging results [13, 11, 10].

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 (1999): *Formalization and Verification of Event-driven Process Chains*. *Information and Software Technology* 41(10), pp. 639–650, doi:10.1016/S0950-5849(99)00016-6.

[2] P. A. Abdulla, C. Aiswarya, M. F. Atig, M. Montali & O. Rezine (2016): *Recency-Bounded Verification of Dynamic Database-Driven Systems*. In: Proc. PODS, ACM Press, pp. 195–210, doi:10.1145/2902251.2902300.

[3] A. Artale, A. Kovtunova, M. Montali & W. M. P. van der Aalst (2019): *Modeling and Reasoning over Declarative Data-Aware Processes with Object-Centric Behavioral Constraints*. In: Proc. BPM, Springer, pp. 139–156, doi:10.1007/978-3-030-26619-6_11.

[4] E. Badouel, L. Hélouët & C. Morvan (2016): *Petri Nets with Structured Data*. Fundam. Inform. 146(1), pp. 35–82, doi:10.3233/FI-2016-1375.

[5] B. Bagheri Hariri, D. Calvanese, G. De Giacomo, A. Deutsch & M. Montali (2013): *Verification of Relational Data-centric Dynamic Systems with External Services*. In: Proc. PODS, pp. 163–174, doi:10.1145/2463664.2465221.

[6] F. Belardinelli, A. Lomuscio & F. Patrizi (2012): *An Abstraction Technique for the Verification of Artifact-Centric Systems*. In: Proc. of KR. Available at `http://www.aaai.org/ocs/index.php/KR/KR12/paper/view/4531`.

[7] M. Bojańczyk, L. Segoufin & S. Toruńczyk (2013): *Verification of database-driven systems via amalgamation*. In: Proc. of PODS, pp. 63–74, doi:10.1145/2463664.2465228.

[8] D. Calvanese, G. De Giacomo & M. Montali (2013): *Foundations of Data Aware Process Analysis: A Database Theory Perspective*. In: Proc. PODS, pp. 1–12, doi:10.1145/2463664.2467796.

[9] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali & A. Rivkin (2018): *Verification of Data-Aware Processes via Array-Based Systems (Extended Version)*. Technical Report arXiv:1806.11459, arXiv.org. Available at `https://arxiv.org/abs/1806.11459`.

[10] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali & A. Rivkin (2019): *Formal Modeling and SMT-Based Parameterized Verification of Data-Aware BPMN*. In: Proc. BPM, Springer, pp. 157–175, doi:10.1007/978-3-030-26619-6_12.

[11] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali & A. Rivkin (2019): *From Model Completeness to Verification of Data Aware Processes*. In: Description Logic, Theory Combination, and All That, Springer, pp. 212–239, doi:10.1007/978-3-030-22102-7_10.

[12] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali & A. Rivkin (2019): *Model Completeness, Covers and Superposition*. In: Proc. of CADE, pp. 142–160, doi:10.1007/978-3-030-29436-6_9.

[13] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali & A. Rivkin (To appear): *SMT-based Verification of Data-Aware Processes: a Model-Theoretic Approach*. Mathematical Structures in Computer Science.

[14] D. Calvanese, M. Montali & A. Santoso (2015): *Verification of Generalized Inconsistency-Aware Knowledge and Action Bases*. In: Proc. IJCAI, AAAI Press, pp. 2847–2853. Available at `http://ijcai.org/Abstract/15/403`.

[15] E. Damaggio, A. Deutsch & V. Vianu (2012): *Artifact Systems with Data Dependencies and Arithmetic*. ACM TODS 37(3), pp. 22:1–22:36, doi:10.1145/2338626.2338628.

[16] M. de Leoni, P. Felli & M. Montali (2018): *A Holistic Approach for Soundness Verification of Decision-Aware Process Models*. In: Proc. ER, pp. 219–235, doi:10.1007/978-3-030-00847-5_17.

[17] R. De Masellis, C. Di Francescomarino, C. Ghidini, M. Montali & S. Tessaris (2017): *Add Data into Business Process Verification: Bridging the Gap between Theory and Practice*. In: Proc. AAAI, AAAI Press, pp. 1091–1099. Available at `http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14627`.

[18] A. Deutsch, R. Hull, F. Patrizi & V. Vianu (2009): *Automatic Verification of Data-Centric Business Processes*. In: Proc. of ICDT, pp. 252–267, doi:10.1145/1514894.1514924.

[19] A. Deutsch, Y. Li & V. Vianu (2016): *Verification of Hierarchical Artifact Systems*. In: Proc. PODS, ACM Press, pp. 179–194, doi:10.1145/2902251.2902275.

[20] M. Dumas (2011): *On the Convergence of Data and Process Engineering*. In: Proc. of ADBIS, pp. 19–26, doi:10.1007/978-3-642-23737-9_2.

[21] M. Dumas, M. La Rosa, J. Mendling & H. A. Reijers (2013): *Fundamentals of Business Process Management*. Springer, doi:10.1007/978-3-642-33143-5.

[22] D. Fahland (2019): *Describing Behavior of Processes with Many-to-Many Interactions*. In: Proc. of PETRI NETS, LNCS 11522, Springer, pp. 3–24, doi:10.1007/978-3-030-21571-2_1.

[23] C. Flanagan & S. Qadeer (2002): *Predicate abstraction for software verification*. In: Proc. of POPL, pp. 191–202, doi:10.1145/503272.503291.

[24] S. Ghilardi, E. Nicolini, S. Ranise & D. Zucchelli (2008): *Towards SMT Model Checking of Array-Based Systems*. In: Proc. of IJCAR, pp. 67–82, doi:10.1007/978-3-540-71070-7_6.

[25] S. Ghilardi & S. Ranise (2010): *Backward Reachability of Array-based Systems by SMT Solving: Termination and Invariant Synthesis*. Logical Methods in Computer Science 6(4), doi:10.2168/LMCS-6(4:10)2010.

[26] Object Management Group (2013): *OMG Unified Modeling Language 2.5*. Http://www.omg.com/uml/.

[27] S. Gulwani & M. Musuvathi (2008): *Cover Algorithms and Their Combination*. In: Proc. of ESOP, Held as Part of ETAPS, pp. 193–207, doi:10.1007/978-3-540-78739-6_16.

[28] B. Bagheri Hariri, D. Calvanese, G. De Giacomo, R. De Masellis, P. Felli & M. Montali (2012): *Verification of Description Logic Knowledge and Action Bases*. In: Proc. ECAI, pp. 103–108, doi:10.3233/978-1-61499-098-7-103.

[29] K. Hoder & Nikolaj Bjørner (2012): *Generalized Property Directed Reachability*. In: Proc. of SAT, pp. 157–171, doi:10.1007/978-3-642-31612-8_13.

[30] L. Kovács & A. Voronkov (2009): *Interpolation and Symbol Elimination*. In: Proc. of CADE, pp. 199–213, doi:10.1007/978-3-642-02959-2_17.

[31] S. Lasota (2016): *Decidability Border for Petri Nets with Data: WQO Dichotomy Conjecture*. In: Proc. of PETRI NETS, LNCS 9698, Springer, pp. 20–36, doi:10.1007/978-3-319-39086-4_3.

[32] Y. Li, A. Deutsch & V. Vianu (2017): *VERIFAS: A Practical Verifier for Artifact Systems*. PVLDB 11(3), pp. 283–296, doi:10.14778/3157794.3157798.

[33] K.L. McMillan (2006): *Lazy Abstraction with Interpolants*. In: Proc. of CAV, pp. 123–136, doi:10.1007/11817963_14.

[34] M. Montali & A. Rivkin (2017): *DB-Nets: on The Marriage of Colored Petri Nets and Relational Databases*. TOPNOC 12, pp. 91–118, doi:10.1007/978-3-662-55862-1_5.

[35] OMG (2009): *Business Process Model and Notation (BPMN) - Version 2.0, Beta 1*.

[36] O. Padon, N. Immerman, S. Shoham, A. Karbyshev & M. Sagiv (2016): *Decidability of inferring inductive invariants*. In: Proc. of POPL, pp. 217–231, doi:10.1145/2837614.2837640.

[37] M. Reichert (2012): *Process and Data: Two Sides of the Same Coin?* In: Proc. of the On the Move Confederated Int. Conf. (OTM 2012), LNCS 7565, Springer, doi:10.1007/978-3-642-33606-5_2.

[38] C. Richardson (2010): *Warning: Don't Assume Your Business Processes Use Master Data*. In: Proc. of BPM, LNCS 6336, Springer, doi:10.1007/978-3-642-15618-2_3.

[39] F. Rosa-Velardo & D. de Frutos-Escrig (2011): *Decidability and complexity of Petri nets with unordered data*. Theor. Comput. Sci. 412(34), pp. 4439–4451, doi:10.1016/j.tcs.2011.05.007.

[40] V. Sofronie-Stokkermans (2016): *On Interpolation and Symbol Elimination in Theory Extensions.* In: Proc. of IJCAR, Lecture Notes in Computer Science, Springer, pp. 273–289, doi:10.1007/978-3-319-40229-1_19.

[41] V. Vianu (2009): *Automatic Verification of Database-Driven Systems: a New Frontier*. In: Proc. of ICDT, pp. 1–13, doi:10.1145/1514894.1514896.