Uniform Interpolation for the Automated Verification of Data-Aware Business Processes

Alessandro Gianola[1]

[1]Free University of Bozen-Bolzano, Italy

Abstract

In the last two decades, Craig interpolation has emerged as a powerful tool in formal verification. Interpolants are largely exploited as an efficient method to approximate the reachable states and for invariant synthesis. In this survey, we report recent results on "stronger interpolants", called *uniform interpolants*, and we discuss how they can be used to develop effective techniques for computing the reachable states in an exact way. Uniform interpolants can be efficiently computed when verifying *data-aware processes*, where the control flow of a (business) process can interact with a data storage. This is significant since integrating data and processes is a long-standing problem in business process management.

Keywords

Uniform Interpolation, Formal Verification, BPM, Data-Aware Processes, SMT

1. Overview

In this survey, we report recent results on the use of *uniform interpolants* (UIs) in automated reasoning [1, 2] and in the context of *formal verification* of the so-called *data-aware (business) processes* [3, 4, 5]. These results originate from the confluence of two well-established research fields: *model-theoretic algebra* [6] in mathematical logic, where UIs were originally investigated for non-classical logics, and *Satisfiability Modulo Theories* (SMT) [7], where UIs provide a 'light form' of quantifier elimination. We discuss how such apparently quite distant scientific paradigms can cooperate in verification of infinite-state systems such as data-aware processes [8, 4, 9, 10], where the control flow of a (business) process can interact with a data storage.

Verification of Data-Aware Business Processes. In recent years, a growing number of AI application domains asks for process modeling languages paired with automated verification techniques that do not just consider the control flow dimension of a process, but also take into consideration the data dimension [11, 12, 13]. From a *theoretical perspective*, this has led to the development of formal frameworks for attacking the problem of verifying data-aware processes (DAPs) [14, 15, 16, 17, 18]. What all these frameworks have in common is that they strive to focus on general DAP models that formalize abstract dynamic systems (i.e., the *process*

OVERLAY 2022: 4th Workshop on Artificial Intelligence and Formal Verification, Logic, Automata, and Synthesis, November 28, 2022, Udine, Italy

gianola@inf.unibz.it (A. Gianola)
https://gianola.people.unibz.it/ (A. Gianola)
0000-0003-4216-5199 (A. Gianola)

component) interacting with data persistent storage (i.e., the *data* component). DAP verification should reflect the possibility of expressing properties that simultaneously account for the data and the process perspectives, and most importantly for their interaction. Because of data, DAP models are *intrinsically infinite-state*: a database is *finite* but its size is *unbounded* and *unknown* a priori (since new tuples can always be added to relations using elements from infinite domains).

From a more *applied* perspective, a huge body of research has been dedicated to the problem of reconciling data and business process management (BPM [19]) within contemporary organizations [20, 21, 22]. 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 [23, 14]: in contrast with abstract DAPs, these models are more focused on concrete processes as they are interpreted by stakeholders and BPM practitioners. One important challenge that naturally arises is how to formally verify such business processes enriched with data [24, 23]: since they are complex infinite-state systems, this requires to develop sophisticated symbolic techniques.

The Problem of Quantifiers. Verification of infinite-state systems, and in particular of DAPs, when studied in *declarative* terms, requires to *symbolically* represent *transitions* and *reachable states*: contrary to finite-state model checking [25], an explicit and exhaustive exploration of the state space is not possible, because of the presence of infinitely many states.

Several techniques based on SMT have been studied to attack the general problem: for instance, there are prominent methods that are based on *forward* reachability, such as K-induction [26], or on *backward* reachability [27]. Methods based on backward reachability *symbolically* explore the state space starting from 'unsafe' states via an iterative computation of predecessors (i.e., the *(pre-)image*), until a fixpoint is reached or the initial state(s) are intersected. There exist various SMT-based model checkers implementing these methods (e.g., Kind 2 [28] or mcmt [29]).

In many first-order (FO) declarative formalisms (e.g., [27]), sets of states are represented using *quantifier-free* logical formulae, called *state formulae*: the content of variables in a state formula characterizes the global state of the system and may change during its evolution. The verification procedure symbolically computes *reachable* sets of states, which should be represented again as state formulae: indeed, images are intended to describe sets of states, and, as such, they should be *quantifier-free*. Reachable sets of states need to be manipulated, because in general are *not* described by state formulae: this is because, when computing images, FO (mostly, *existential*) quantifiers may be introduced by the transitions, which are usually quantified formulae. For instance, this is the case of DAP models, whose transitions contain as guards existential queries over a relational database [8, 4]. These quantifiers appear in image computation, breaking the quantifier-free format of state formulae. Hence, a sort of *quantifier elimination (QE)* is needed to represent reachable states as state formulae. QE is also essential for efficiency reasons: e.g., using approaches à la Bounded Model Checking [30], where paths with incremental length are encoded, QE guarantees that the size of formulae does not increase, avoiding their blow-up.

Dealing with quantifiers is a genuine problem when using frameworks based on SMT. For instance, declarative versions of backward reachability [31, 27] require discharging to SMT solvers proof obligations in the form of satisfiability tests for quantified formulae with a restricted shape. Although SMT solvers natively reason about the quantifier-free fragments, FO quantifiers can be in some cases handled by *instantiation* [27], whereas in others [32], where quantifiers range over specific real values involving *light forms* of arithmetic, proper QE can be used [33].

To summmarize, the *precise* computation of the set of reachable states can be in principle performed via *proper* quantifier elimination. However, quantifier elimination is not always possible in generic FO theories, and, when available, is in many cases computationally intractable: for instance, this is the case of arithmetical theories. In order to cope with this problem, other methods for symbol elimination (e.g., predicate abstraction [34, 35] or *ordinary (Craig) interpolation* [36, 37, 38]) have been investigated, which do not compute precise images, but perform an *approximation* of the reachable states: this implies that images can contain 'spurious elements', i.e., states that are not properly reachable in one step. Nevertheless, these methods are quite successful and computationally efficient. The main limitation is that they usually require refinement techniques to handle the spurious traces possibly produced by the approximation.

2. Model Completions and Uniform Interpolation

We now remark how in significant cases (e.g., data-aware business processes), approximating methods can be abandoned in favor of *exact methods* that have the merits of both computing precise images and remaining computationally tractable (i.e., in polytime). These methods are based on the use of *uniform interpolants*, a strong form of interpolants having strict relationships with quantifier elimination performed in richer theories called *model completions*. We clarify why model completions come to the picture and how they are related to uniform interpolation.

Model-Theoretic Algebra and Model Completions An FO theory T has *quantifier elimination* iff for every formula ϕ in the signature Σ of T there is a quantifier-free formula ϕ' s.t. ϕ is T-equivalent to ϕ'. Eliminating quantifiers from a generic FO formula can be equivalently formulated as the problem of eliminating *existential* quantifiers from *constraints*. Elimination of existentials has an interesting interpretation: it can be seen as the logical counterpart of finding *witnesses*, i.e., *solutions*, to suitable systems of logical "equations and/or disequalities". Model-theoretic algebra, since the pioneering work by Robinson [39, 6], provides a powerful setting where to formulate this problem in algebraic terms and solve it using model theory.

The central notion is that of *existentially closed model*. A quantifier-free formula with parameters in a model M is *solvable* if there is an extension M' of M where the formula is satisfied. A model M is *existentially closed* if any solvable formula already has a solution in M itself. This is not FO definable in general. However, in significant cases, the class of existentially closed models of T are exactly the models of another FO theory T^*, called *model completion* [40] of T. Formally, a *universal* theory T has a *model completion* iff there is a stronger theory $T^* \supseteq T$ (still within the same signature Σ of T) s. t.: (i) every Σ-constraint satisfiable in a model of T is satisfiable in a model of T^*; (ii) T^* *eliminates quantifiers*. The theory T^*, if it exists, is unique. In model completions, QE holds, even in case it does not in T. The model completion of a theory identifies the class of the models where *all satisfiable existential statements can be satisfied*.

In declarative approaches to verification of infinite-state systems, the runs of a system that are 'analyzed' by image computation are identified with certain *definable paths* in the models of a suitable theory T: e.g., in the case of DAP models, the theory T formalizes the relational database that is queried and modified by the process [3]. As noticed, when performing these computations, FO quantifiers are introduced and need to be eliminated, even in case of theories T not admitting QE. Nevertheless, in significant cases where QE is not available, model completions still exist. This is the case of useful theories such as the ones used for DAP verification [3, 4].

Model completions become the crucial tool to exploit: one of the main results from [3] is that one can *restrict the analysis to paths within existentially closed models*, thus taking profit from the properties of the model completion T^*, first of all QE. For instance, in [3] it is shown that, in the case of safety verification, performing backward reachability for systems whose models live in T is *equivalent* to performing backward reachability for systems whose models live in T^*: favorably, in T^* quantifiers can be eliminated, even when QE is not available in T.

UIs and QE in Model Completions. We give a general definition of UIs and we discuss their strict relationship with model completions. Let T be a logic or a theory and let L be a suitable fragment (propositional, FO quantifier-free, etc.) of its language. Given an L-formula $\phi(\underline{x}, \underline{y})$ ($\underline{x}, \underline{y}$ are the variables occurring in ϕ), a *(L-)uniform interpolant (UI)* of ϕ (w.r.t. $\underline{y}$) is an L-formula $\phi'(\underline{x})$ where only the $\underline{x}$ occur, satisfying these two properties: ***(i)*** $\phi(\underline{x}, \underline{y}) \vdash_T \phi'(\underline{x})$; ***(ii)*** for any further L-formula $\psi(\underline{x}, \underline{z})$ such that $\phi(\underline{x}, \underline{y}) \vdash_T \psi(\underline{x}, \underline{z})$, we have $\phi'(\underline{x}) \vdash_T \psi(\underline{x}, \underline{z})$.

For every pair of L-formulae $\phi(\underline{x}, \underline{y})$ and $\psi(\underline{x}, \underline{z})$ s.t. $\phi \vdash_T \psi$, a UI of ϕ is in particular an *ordinary (Craig) interpolant* for the pair (ϕ, ψ) [41]. Hence, whenever UIs exist, one can compute an ordinary interpolant for $\phi \vdash_T \psi$ in a way that is *independent* of ψ, i.e., *uniformly*.

UIs were originally investigated in non-classical logics, since the work by Pitts [42]. They are stronger than ordinary interpolants: even in case Craig interpolants exist, UIs may not exist. Since the nineties, they have been extensively studied in a large literature (e.g., [43, 44, 45]).

Recently, the automated reasoning community has developed an increasing interest in UIs, where L is the *quantifier-free* fragment of an FO theory T: from now on, we restrict our attention to this case. This is witnessed by various talks by Kapur (FLoC 2010, ISCAS 2013-14, SCS 2017 [46]), as well as by Gulwani and Musuvathi in [47], where UIs are called *covers*. The use of UIs in model checking to compute *exact* images of states was first shown in that work, and then further motivated by DAP verification [8, 1]. The first formal *proof* about the *existence* of UIs in $\mathcal{EUF}$ (Equality and Uninterpreted Functions) was published in [48, 1], where was also proved that computing UIs in T is *equivalent* to eliminating quantifiers in its model completion T^*. Hence, instead of investigating paths in T^* and eliminating quantifiers in model completions, reachability can be performed in T itself by computing UIs there. In [48], a UI algorithm for $\mathcal{EUF}$ relying on the *constrained Superposition Calculus* was proposed: in the case that is used to verify DAP models, a quadratic bound in time for UI computation can be given [1]. Two simpler UI algorithms are in [49].

DAP verification also suggests the study of UI transfer to combined theories: it is natural to consider the combination of theories accounting for different datatypes contained in the persistent storage [2, 8]. The UI transfer problem is: *if UIs exist in theories T_1 and T_2, under which conditions do they exist also in the combined theory $T_1 \cup T_2$?* In [50, 2] combined UIs are shown to exist in the disjoint-signatures convex case under the same hypothesis, i.e., the *equality interpolating condition* [51], guaranteeing the transfer of quantifier-free ordinary interpolation; in [50, 2], a combined UI algorithm, based on the use of *Beth definability*, is also provided.

Acknowledgments

This work is partially supported by the Italian Ministry of University and Research under the PRIN program, grant B87G22000450001 (PINPOINT), and by the Free University of Bozen-Bolzano with the SMART-APP and ADAPTERS projects.

References

[1] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Model completeness, uniform interpolants and superposition calculus, Journal of Automated Reasoning 65 (2021) 941–969.

[2] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Combination of uniform interpolants via Beth definability, Journal of Automated Reasoning 66 (2022) 409–435.

[3] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, SMT-based verification of data-aware processes: a model-theoretic approach, Mathematical Structures in Computer Science 30 (2020) 271–313.

[4] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Formal modeling and SMT-based parameterized verification of data-aware BPMN, in: Proceeding of BPM 2019, volume 11675 of *LNCS*, Springer, 2019, pp. 157–175.

[5] A. Gianola, SMT-based safety verification of data-aware processes: Foundations and applications, in: Proc. of the Best Dissertation Award, Doctoral Consortium, and Demonstration & Resources Track at BPM 2022, volume 3216 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022, pp. 20–24.

[6] A. Robinson, Introduction to model theory and to the metamathematics of algebra, Studies in logic and the foundations of mathematics, North-Holland, 1963.

[7] C. W. Barrett, C. Tinelli, Satisfiability modulo theories, in: Handbook of Model Checking., Springer, 2018, pp. 305–343.

[8] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Verification of data-aware processes: Challenges and opportunities for automated reasoning, in: Proceedings of ARCADE 2019, volume 311, EPTCS, 2019.

[9] S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Delta-BPMN: A concrete language and verifier for Data-Aware BPMN, in: Proceedings of BPM 2021, volume 12875 of *LNCS*, Springer, 2021, pp. 179–196.

[10] S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Petri net-based object-centric processes with read-only data, Information Systems 107 (2022).

[11] B. B. Hariri, D. Calvanese, G. De Giacomo, R. De Masellis, P. Felli, Foundations of relational artifacts verification, in: Proceedings of BPM 2011, volume 6896 of *LNCS*, Springer, 2011, pp. 379–395.

[12] G. D. Giacomo, R. D. Masellis, M. Grasso, F. M. Maggi, M. Montali, Monitoring business metaconstraints based on LTL and LDL for finite traces, in: Proc. of BPM 2014, volume 8659 of *Lecture Notes in Computer Science*, Springer, 2014, pp. 1–17. URL: https://doi.org/10.1007/978-3-319-10172-9_1. doi:10.1007/978-3-319-10172-9_1.

[13] G. D. Giacomo, X. Oriol, M. Estañol, E. Teniente, Linking data and BPMN processes to achieve executable models, in: Proc. of CAiSE, LNCS, Springer, 2017, pp. 612–628.

[14] D. Calvanese, G. De Giacomo, M. Montali, Foundations of data-aware process analysis: A database theory perspective, in: Proceedings of PODS 2013, 2013, pp. 1–12.

[15] B. Bagheri Hariri, D. Calvanese, G. De Giacomo, A. Deutsch, M. Montali, Verification of relational data-centric dynamic systems with external services, in: Proceedings of PODS 2013, 2013, pp. 163–174.

[16] A. Deutsch, R. Hull, Y. Li, V. Vianu, Automatic verification of database-centric systems, ACM SIGLOG News 5 (2018) 37–56.

[17] A. Deutsch, Y. Li, V. Vianu, Verification of hierarchical artifact systems, in: Proceedings of PODS 2016, 2016, pp. 179–194.

[18] A. Deutsch, Y. Li, V. Vianu, Verification of hierarchical artifact systems, ACM Transactions on Database Systems 44 (2019) 12:1–12:68.

[19] M. Dumas, M. L. Rosa, J. Mendling, H. A. Reijers, Fundamentals of Business Process Management, Second Edition, Springer, 2018.

[20] 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.

[21] M. Dumas, On the convergence of data and process engineering, in: Proceedings of ADBIS 2011, volume 6909 of *LNCS*, Springer, 2011, pp. 19–26.

[22] 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.

[23] E. Damaggio, A. Deutsch, R. Hull, V. Vianu, Automatic verification of data-centric business processes, in: Proceedings of BPM 2011, volume 6896 of *LNCS*, Springer, 2011, pp. 3–16.

[24] A. Deutsch, R. Hull, F. Patrizi, V. Vianu, Automatic verification of data-centric business processes, in: Proceedings of ICDT 2009, 2009, pp. 252–267.

[25] E. M. Clarke, O. Grumberg, D. A. Peled, Model checking, MIT Press, 2001.

[26] M. Sheeran, S. Singh, G. Stålmarck, Checking safety properties using induction and a SAT-solver, in: Proceedings of FMCAD 2000, volume 1954 of *LNCS*, Springer, 2000, pp. 108–125.

[27] S. Ghilardi, S. Ranise, Backward reachability of array-based systems by SMT solving: Termination and invariant synthesis, Logical Methods in Computer Science 6 (2010).

[28] A. Champion, A. Mebsout, C. Sticksel, C. Tinelli, The Kind 2 model checker, in: Proceedings of CAV 2016, volume 9780 of *LNCS*, Springer, 2016, pp. 510–517.

[29] S. Ghilardi, S. Ranise, MCMT: A model checker modulo theories, in: Proceedings of IJCAR 2010, volume 6173 of *LNCS (LNAI)*, Springer, 2010, pp. 22–29.

[30] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, Y. Zhu, Bounded model checking, Adv. Comput. 58 (2003) 117–148. URL: https://doi.org/10.1016/S0065-2458(03)58003-2. doi:10.1016/S0065-2458(03)58003-2.

[31] S. Ghilardi, E. Nicolini, S. Ranise, D. Zucchelli, Towards SMT model checking of array-based systems, in: Proceedings of IJCAR 2008, volume 5195 of *LNCS (LNAI)*, Springer, 2008, pp. 67–82.

[32] F. Alberti, R. Bruttomesso, S. Ghilardi, S. Ranise, N. Sharygina, An extension of lazy abstraction with interpolation for programs with arrays, Formal Methods in System Design 45 (2014) 63–109.

[33] D. C. Cooper, Theorem proving in arithmetic without multiplication., in: Machine Intelligence, volume 7, Edinburgh University Press, 1972, pp. 91–100.

[34] R. Jhala, R. Majumdar, Software model checking, ACM Comput. Surv. 41 (2009) 21:1–21:54.

[35] R. Jhala, A. Podelski, A. Rybalchenko, Predicate abstraction for program verification, in: Handbook of Model Checking, Springer, 2018, pp. 447–491.

[36] K. L. McMillan, Interpolation and SAT-Based Model Checking, in: Proceedings of CAV 2003, volume 2725 of *LNCS*, Springer, 2003, pp. 1–13.

[37] K. L. McMillan, Lazy Abstraction with Interpolants, in: Proceedings of CAV 2006, volume 4144 of *LNCS*, Springer, 2006, pp. 123–136.

[38] L. Kovács, A. Voronkov, Interpolation and symbol elimination, in: Proceedings of CADE 2009, volume 5663 of *LNCS (LNAI)*, Springer, 2009, pp. 199–213.

[39] A. Robinson, On the metamathematics of algebra, Studies in Logic and the Foundations of Mathematics, North-Holland Publishing Co., Amsterdam, 1951.

[40] C.-C. Chang, J. H. Keisler, Model Theory, third ed., North-Holland Publishing Co., Amsterdam-London, 1990.

[41] W. Craig, Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory, Journal of Symbolic Logic 22 (1957) 269–285.

[42] A. M. Pitts, On an interpretation of second order quantification in first order intuitionistic propositional logic, Journal of Symbolic Logic 57 (1992) 33–52.

[43] S. Ghilardi, M. Zawadowski, Sheaves, games, and model completions, volume 14 of *Trends in Logic—Studia Logica Library*, Kluwer Academic Publishers, Dordrecht, 2002.

[44] T. Kowalski, G. Metcalfe, Uniform interpolation and coherence, Annals of Pure and Applied Logic 170 (2019) 825–841.

[45] G. Metcalfe, L. Reggio, Model completions for universal classes of algebras: necessary and sufficient conditions, Journal of Symbolic Logic (2022) 1–34.

[46] D. Kapur, Nonlinear polynomials, interpolants and invariant generation for system analysis, in: Proceedings of SC-Square 2017 (co-located with ISSAC), volume 1974, CEUR Workshop Proceedings, 2017.

[47] S. Gulwani, M. Musuvathi, Cover algorithms and their combination, in: Proceedings of ESOP 2008, volume 4960 of *LNCS*, Springer, 2008, pp. 193–207.

[48] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Model completeness, covers and superposition, in: Proceedings of CADE 2019, volume 11716 of *LNCS (LNAI)*, Springer, 2019, pp. 142–160.

[49] S. Ghilardi, A. Gianola, D. Kapur, Uniform interpolants in EUF: Algorithms using DAG-representations, Logical Methods in Computer Science 18 (2022).

[50] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Combined Covers and Beth Definability, in: Proceedings of IJCAR 2020, volume 12166 of *LNCS (LNAI)*, Springer, 2020, pp. 181–200.

[51] G. Yorsh, M. Musuvathi, A combination method for generating interpolants, in: Proceedings of CADE 2005, volume 3632 of *LNCS*, Springer, 2005, pp. 353–368.