veriT at SMT-COMP 2021

Haniel Barbosa[1], Pascal Fontaine[2,3], Hans-Jörg Schurr[2]*

[1] Universidade Federal de Minas Gerais (UFMG)
hbarbosa@dcc.ufmg.br
[2] CNRS, Inria, and the University of Lorraine, Nancy, France
{daniel.elouraoui, pascal.fontaine, hans-jorg.schurr}@loria.fr
[3] Université de Liège, Belgium

URL : http://www.verit-solver.org — Seed : 6923

veriT is a satisfiability modulo theory (SMT) solver developed at the University of Lorraine, Inria (Nancy, France), and Université de Liège (Liège, Belgium). veriT provides an open, trustable and reasonably efficient decision procedure [7] for the logic of quantifier-free formulas over uninterpreted symbols, linear real arithmetics, and the combination thereof. It also handles linear arithmetics over integers, and has quantifier reasoning using trigger- and conflict-based instantiation [4] as well as enumerative instantiation [10]. Recently, veriT was extended to higher-order logic [5]. Also, veriT is proof producing [9, 3] and its proofs are reconstructed in both Isabelle/HOL [2] and Coq [1]. Since Isabelle/HOL's 2021 release, Sledgehammer [6] uses the reconstruction of veriT proofs for a proof tactic [11].

veriT is written in C and accepts the input formats SMT-LIB 2.6 and DIMACS. It integrates a CDCL($\mathcal{T}$)-based boolean satisfiability engine with a Nelson-Oppen like combination of decision and semi-decision procedures with propagation of model equalities, and implements simplifications such as symmetry-based reductions [8]. The tool is open-source and distributed under the BSD licence.

The veriT version competing in SMT-COMP 2021 features new quantifier preprocessing techniques and an improved strategy scheduler.

Current instantiation techniques, such as the ones implemented by veriT, struggle with quantified formulas that contain nested quantifiers. To better handle such formulas, veriT now augments the input problem with formulas where some nested quantified formulas have been eliminated by unifying them with unit assertions. The method is carefully constrained to ensure that the newly generated formulas are useful for the solver. This allows veriT to prove more quantified formulas, faster.

Since the SMT-COMP 2020, we generate the scheduling scripts for veriT automatically. To do so we first collected the necessary solving time of the SMT-LIB benchmarks for a predefined list of strategies and fix a list of allowed timeouts. We then encoded the problem of finding an optimal schedule as an integer programming problem that maximizes the number of problems solved within the total timeout. In a second step we calculated the order of the resulting option list

[*] The author order is strictly alphabetic.

that minimizes the total CPU time. Our schedule-generator now supports multiple, nested timeouts. Hence, veriT's schedule is first optimized for a 24 second timeout and then optimized to use the remaining time optimally.

veriT participates in the following divisions: ALIA AUFLIA AUFLIRA LIA UF UFIDL UFLIA UFLRA QF_ALIA QF_AUFLIA QF_IDL QF_LIA QF_LRA QF_RDL QF_UF QF_UFIDL QF_UFLIA QF_UFLRA.

Acknowledgements The development of veriT is funded by the European Research Council (ERC) starting grant Matryoshka (713999).

References

1. M. Armand, G. Faure, B. Grégoire, C. Keller, L. Théry, and B. Werner. A modular integration of SAT/SMT solvers to Coq through proof witnesses. In J.-P. Jouannaud and Z. Shao, editors, *Certified Programs and Proofs*, volume 7086 of *Lecture Notes in Computer Science*, pages 135–150. Springer, 2011.
2. H. Barbosa, J. C. Blanchette, M. Fleury, and P. Fontaine. Scalable fine-grained proofs for formula processing. *Journal of Automated Reasoning*, 64(3):485–510, 2020.
3. H. Barbosa, J. C. Blanchette, and P. Fontaine. Scalable fine-grained proofs for formula processing. In L. de Moura, editor, *Proc. Conference on Automated Deduction (CADE)*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2017.
4. H. Barbosa, P. Fontaine, and A. Reynolds. Congruence closure with free variables. In A. Legay and T. Margaria, editors, *Tools and Algorithms for Construction and Analysis of Systems (TACAS)*, volume 10206 of *Lecture Notes in Computer Science*, pages 214–230, 2017.
5. H. Barbosa, A. Reynolds, D. E. Ouraoui, C. Tinelli, and C. W. Barrett. Extending SMT solvers to higher-order logic. In P. Fontaine, editor, *Proc. Conference on Automated Deduction (CADE-27)*, volume 11716 of *Lecture Notes in Computer Science*, pages 35–54. Springer, 2019.
6. J. C. Blanchette, S. Böhme, and L. C. Paulson. Extending sledgehammer with SMT solvers. *Journal of Automated Reasoning*, 51(1):109–128, 2013.
7. T. Bouton, D. C. B. de Oliveira, D. Déharbe, and P. Fontaine. veriT: an open, trustable and efficient SMT-solver. In R. A. Schmidt, editor, *Proc. Conference on Automated Deduction (CADE-22)*, 2009.
8. D. Déharbe, P. Fontaine, S. Merz, and B. Wolzenlogel Paleo. Exploiting Symmetry in SMT Problems. In N. Bjørner and V. Sofronie-Stokkermans, editors, *Proc. Conference on Automated Deduction (CADE-23)*, 2011.
9. D. Déharbe, P. Fontaine, and B. Wolzenlogel Paleo. Quantifier Inference Rules for SMT proofs. In *First Workshop on Proof eXchange for Theorem Proving (PxTP)*, 2011.
10. A. Reynolds, H. Barbosa, and P. Fontaine. Revisiting enumerative instantiation. In D. Beyer and M. Huisman, editors, *Tools and Algorithms for Construction and Analysis of Systems (TACAS)*, volume 10806 of *Lecture Notes in Computer Science*, pages 112–131. Springer, 2018.
11. H.-J. Schurr, M. Fleury, and M. Desharnais. Reliable reconstruction of fine-grained proofs in a proof assistant. In *CADE 28*, LNCS, 2021.