# A Benchmark for Infinite Models in SMT * (ongoing work)

Mikoláš Janota[1], Chad E. Brown[1], and Cezary Kaliszyk[2]

[1] Czech Technical University, Prague
[2] University of Innsbruck, Innsbruck, Austria and INDRC, Czech Republic

**Introduction** *Satisfiability modulo theories* (SMT) [1] require model finding for formulas. This task is conceptually easy for quantifier-free theories but is extremely difficult in the presence of quantifiers. Indeed, in the theory of linear real arithmetic, one can easily see that for the formula $2c = 1$, setting $c = 0.5$ is a model, and, the same formula has no model in the theory of integer arithmetic. However, the formula $(\forall x)(fx > x)$ requires us to construct the function $f$ (e.g. $fx = x + 1$). Unlike proof-finding, model-finding is not even semi-decidable.

There is a long-standing tradition of *finite* model finders [6, 3]—these can be implemented by search on models of increasing, but finite, size. In the case of *infinite* models, the search space is extremely unwieldy and we expect that machine learning methods will be able to help there.

In recent evaluation Parsert et al. evaluate the capabilities of several existing tools that enable infinite model finding [7]. This evaluation showed clear limitations of the existing tools but it does not provide a data set on which we could evaluate or train statistical approaches. In this ongoing work, we aim to create a benchmark of formulas with infinite models.

**Approach** In practice, users generate *many* satisfiable problems. Indeed, any incorrectly specified or implemented program leads to a satisfiability SMT (in TPTP terminology counter-satisfiability). Unfortunately, these intermediate, incorrect formulas are typically not made available. However, we make seek inspiration in the available SMT formula and look at their fragments and modifications.

In this work, we consider a simple technique to generate interesting fragments from a given SMT formula (understood as a conjunction of assertions). The idea is to consider some parameter $k$ and extract fragments that contain $k$ uninterpreted functions. Given an SMT formula, we choose $k$ uninterpreted function symbols that appear in the formula and filter out the conjuncts that are only weakly related to these $k$ symbols. Let us describe the process for $k = 2$. Consider an SMT formula $\phi$ and two uninterpreted function symbols $f$ and $g$ that occur in $\phi$. Consider subformula $\psi$ in $\phi$ (one of the conjuncts) of the form $(\forall \overline{x})\psi'$, where $\overline{x}$ is an arbitrary set of variables and $\psi'$ is quantifier free. We will say that $\psi$ is *in the $f, g$ fragment of $\phi$* if it contains at least one $f$ or $g$ and no other uninterpreted functions; there is no limitation on constants. *The global $f, g$ fragment of $\phi$* is defined as the conjunction of all the $f, g$ fragments in $\phi$. We remark that in the current implementation we only consider subformulas that are denoted by the users as separate assertions; this could be relaxed.

**Experiments** As an initial test et we use the UFNIA family from SMT-LIB. These are formulas that contain uninterpreted functions (UF) and nonlinear integer arithmetic (NIA). The family contains 13,463 SMT formulas.

*The results were supported by the Ministry of Education, Youth and Sports within the dedicated program ERC CZ under the project POSTMAN no. LL1902.

Extracting the global 2-fragments for all the formulas and all functions $f, g$ appearing in them produces 19,566 non-duplicate fragments. Each fragment can be considered as a standalone SMT problem.

Interestingly, the SMT solver Z3 [4] can solve 15,626 out of these, most of them are trivial. This leaves 3,938 challenge benchmark problems. It is worth noting, that Z3 enables finding some simple infinite models (such as identity) throughout model-based quantifier instantiation (mbqi) [5].

**Conclusions and Future Work** The proposed approach enables us to easily produce formulas that are difficult for state-of-the-art SMT solvers. We plan to extend this generation process by focusing also on different families of formulas and consider different parameterization of the process. An interesting question arises, how do we know that the formulas require infinite models? In the case of formulas that use integers/reals/string etc., this is immediate. In general, of course, this is again an undecidable problem. However, one more might try to attempt using heuristic techniques such as Infinox [2].

# References

[1] Clark Barrett and Cesare Tinelli. Satisfiability modulo theories. In *Handbook of Model Checking*, pages 305–343. Springer, 2018.

[2] Koen Claessen and Ann Lillieström. Automated inference of finite unsatisfiability. *J. Autom. Reason.*, 47(2):111–132, 2011.

[3] Koen Claessen and Niklas Sörensson. New techniques that improve MACE-style finite model finding. In *Proceedings of the CADE-19 Workshop: Model Computation - Principles, Algorithms, Applications*, 2003.

[4] Leonardo Mendonça de Moura and Nikolaj Bjørner. Z3: an efficient SMT solver. In C. R. Ramakrishnan and Jakob Rehof, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008*, volume 4963 of *LNCS*, pages 337–340. Springer, 2008.

[5] Yeting Ge and Leonardo Mendonça de Moura. Complete instantiation for quantified formulas in satisfiability modulo theories. In *Computer Aided Verification, 21st International Conference, CAV*, pages 306–320, 2009.

[6] William McCune. Mace4 reference manual and guide. *CoRR*, cs.SC/0310055, 2003.

[7] Julian Parsert, Chad E. Brown Mikoláš Janota, and Cezary Kaliszyk. Experiments on infinite model finding in smt solving. In *LPAR 2023, to appear*. EasyChair, 2023.