SMT and Functional Equation Solving over the Reals: Challenges from the IMO*

Chad E. Brown[1], Karel Chvalovský[1], Mikoláš Janota[1], Mirek Olšák[2], and Stefan Ratschan[3]

[1] Czech Technical University in Prague, Prague, Czechia
[2] University of Cambridge
[3] Institute of Computer Science Academy of Sciences of the Czech Republic

1 Introduction

This abstract describes a contribution accepted for CADE 2025 [2], which does not use any statistical-based learning. Such extensions are described in Section 4, as future work.

This work investigates the use of Satisfiability Modulo Theories (SMT) to solve functional equations over the reals—a category of problems that frequently appears in International Mathematical Olympiad (IMO) and other similar competitions. These problems involve determining all functions $f : \mathbb{R} \to \mathbb{R}$ that satisfy a given constraint, such as:

$$\forall x, y.\ f(x + y) = xf(y) + yf(x).$$

The only solution in this case is $f = \lambda x.\, 0$ (constant 0), which can be seen by substituting $\{x \mapsto x, y \mapsto 0\}$ and $\{x \mapsto 0, y \mapsto 0\}$.

While the mathematical content is at a high-school level, such problems are difficult for state-of-the-art SMT solvers due to the combination of uninterpreted functions and nonlinear real arithmetic.

The paper tackles two main challenges:

1. **Function synthesis**: Finding all functions f that solve the constraint within a predefined template.
2. **Completeness verification**: Proving that no other functions outside the template satisfy the given constraints.

2 Main Contributions

1. Template-Based Solution Search with Quantifier Elimination

Our core technique involves fixing a template—typically a polynomial such as $f(x) = ax^2 + bx + c$—and solving for the parameters a, b, c, using real quantifier elimination (QE). We use the Tarski [8] system for QE but also avail of special cases and use Symbolic algebra tool SymPy [6].

*Supported by the Czech MEYS under the ERC CZ project no. LL1902 *POSTMAN*, by the European Union under the project *ROBOPROX* (reg. no. CZ.02.01.01/00/22_008/0004590).

2. Proving Completeness via SMT

Once a candidate solution Ψ is found, to prove that it captures *all* valid solutions, we attempt to prove the implication $\Phi \Rightarrow \Psi$, or equivalently, that $\Phi \wedge \neg\Psi$ is unsatisfiable. For the motivating example, prove:

$$(\forall x, y.\ f(x+y) = xf(y) + yf(x)) \Rightarrow (\forall x.\ f(x) = 0)$$

However, SMT solvers struggle with this step.[1] To overcome this, the paper introduces:

- **Lemma generation**: Automatically proposing and proving auxiliary ground equalities (e.g., $f(0) = 0$) that assist in solving the main goal.
- **Quantifier instantiation techniques**:
  - *Partial instantiations*: Assigning small constants (e.g., 0 or 1) to some variables to simplify formulas.
  - *Theory-guided substitutions*: Solving symbolic equations to find helpful instantiations, particularly when terms do not syntactically appear in the original formula.

3 Experimental Results

A benchmark suite of 422 functional equation problems (79 from Musil's dataset [3, 7], 343 from AoPS scraping) is used. The techniques led to substantial gains. With lemma generation and instantiation techniques, the number of solved problems more than doubled. The system also successfully solved problems that previously required manual insight, demonstrating the power of integrating reasoning, symbolic computation, and SMT.

4 Future Work

The current implementation does not rely on any machine learning. There are two main directions where we plan to extend the work. Currently, candidate lemmas are generated a exhaustively according to hand-designed rules. Instead, we plan to propose the candidate lemmas by ML. A second main direction is to attempt solving the problems with an LLM directly. If this is successful, we will still not be sure that the answer is correct but in such case, we will try to extract useful lemmas from the LLM's answer.

References

[1] Haniel Barbosa, Clark W. Barrett, Martin Brain, Gereon Kremer, Hanna Lachnitt, Makai Mann, Abdalrhman Mohamed, Mudathir Mohamed, Aina Niemetz, Andres Nötzli, Alex Ozdemir, Mathias Preiner, Andrew Reynolds, Ying Sheng, Cesare Tinelli, and Yoni Zohar. cvc5: A versatile and industrial-strength SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems, TACAS*, volume 13243 of *LNCS*, pages 415–442. Springer, 2022. doi:10.1007/978-3-030-99524-9_24.

[2] Chad E. Brown, Karel Chvalovský, Mikoláš Janota, Mirek Olšák, and Stefan Ratschan. SMT and functional equation solving over the reals: Challenges from the IMO, 2025. URL: https://arxiv.org/abs/2504.15645, arXiv:2504.15645.

[1] The evaluation uses a portfolio of various settings of cvc5 [1], z3 [4], and vampire [5].

[3] Chad E. Brown, Mikoláš Janota, and Mirek Olšák. Symbolic computation for all the fun. In C. W. Brown, D. Kaufmann, C. Nalon, A. Steen, and M. Suda, editors, *Joint Proceedings of the 9th Workshop on Practical Aspects of Automated Reasoning (PAAR) and the 9th Satisfiability Checking and Symbolic Computation Workshop (SC-Square), 2024 co-located with the 12th International Joint Conference on Automated Reasoning (IJCAR 2024)*, volume 3717 of *CEUR Workshop Proceedings*, pages 111–121. CEUR-WS.org, 2024. URL: https://ceur-ws.org/Vol-3717/paper6.pdf.

[4] Leonardo Mendonça de Moura and Nikolaj Bjørner. Z3: an efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008*, volume 4963, pages 337–340. Springer, 2008. doi:10.1007/978-3-540-78800-3_24.

[5] Laura Kovács and Andrei Voronkov. First-order theorem proving and Vampire. In Natasha Sharygina and Helmut Veith, editors, *Computer Aided Verification - 25th International Conference, CAV*, volume 8044 of *LNCS*, pages 1–35. Springer, 2013. doi:10.1007/978-3-642-39799-8_1.

[6] Aaron Meurer, Christopher P Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K Moore, Sartaj Singh, et al. SymPy: symbolic computing in Python. *PeerJ Computer Science*, 3:e103, 2017. URL: https://www.sympy.org.

[7] Vít Musil. Funkcionální rovnice, 2024. Online library of the Matematický korespondenční seminář PraSe (PRAžský SEminář), Downloaded 8 March 2024. URL: https://prase.cz/library/FunkcionalniRovniceVM/FunkcionalniRovniceVM.pdf.

[8] Fernando Vale-Enriquez and Christopher W. Brown. Polynomial constraints and unsat cores in Tarski. In James H. Davenport, Manuel Kauers, George Labahn, and Josef Urban, editors, *Mathematical Software - ICMS - 6th International Conference*, volume 10931 of *LNCS*, pages 466–474. Springer, 2018. Code obtained from https://github.com/chriswestbrown/tarski. URL: https://www.usna.edu/Users/cs/wcbrown/tarski/index.html, doi:10.1007/978-3-319-96418-8_55.