PlatSmt in SMT-COMP 2024

David Ewert (MSc. University of British Columbia)

June 14, 2024

1 Introduction

PlatSmt is an open-source SMT solver implemented in Rust. It is available on GitHub at `https://github.com/dewert99/plat-smt`. The solver currently only supports uninterpreted functions and uses the standard CDCL(T) [5] architecture.

2 Description

PlatSmt uses a data structure developed in section 2.1 of Nieuwenhuis and Oliveras [4] as part of its explanations. Instead of always using the oldest explanation, PlatSmt sometimes collapses subpaths of the oldest explanation when they have corresponding literals in the SAT solver. For example, given the trail: `a=b,b=c,a=c,a=x,c=d,x=d,a=d`, the oldest explanation of `a=d` would be `a=b,b=c,c=d`, but PlatSmt might compress the subpath `a=b,b=c` and instead give the explanation: `a=c,c=d`, but it wouldn't give the explanation: `a=x,x=d` since `x` isn't in the oldest explanation. If multiple subpaths with the same start and end points miss being compressed because the SAT-solver does not have a literal corresponding to the equality of the start and end points, this literal gets added to the SAT-solver so these subpaths can be compressed in the future. Adding these literals allows PlatSmt to avoid exponential blowup when given the diamond problem (Figure 5 from Rozanov and Strichman [6]).

3 Acknowledgements

PlatSmt uses PlatSat (tag: `SMT-COMP2024`) as its SAT solver and PlatEgg (tag: `SMT-COMP2024`) as its e-graph library. PlatSat is a fork I created from BatSat [1], which is a fork of RatSat [3], which is a re-implementation of MiniSat [2] in Rust. PlatEgg is a fork I created from egg [7].

3.1 Other Dependencies

- rustc-hash

- hashbrown
- log
- thiserror-no-std
- env_logger
- perfect-derive
- smallvec
- no-std-compat

4 Competition Version

In the SMT competition 2024, I am participating with the most recent development version of PlatSmt (tag: `SMT-COMP2024-v2`) This version is entering the `QF_UF` logic in the single query, incremental, model-validation, and unsat-core tracks. The binary can be compiled using:

```
> rustup install 1.78.0
> cargo +1.78.0 install --git https://github.com/dewert99/plat-smt
  --tag SMT-COMP2024
```

References

[1] S. Cruanes, N. Mometto, M. Bray, and N. Braud. Batsat. GitHub, Aug. 2018. URL `https://github.com/c-cube/batsat`.

[2] N. Eén and N. Sörensson. *An Extensible SAT-solver*, pages 502–518. Springer Berlin Heidelberg, 2004. ISBN 9783540246053. doi:10.1007/978-3-540-24605-3_37.

[3] M. Hara. Ratsat. GitHub, Mar. 2018. URL `https://github.com/qnighy/ratsat`.

[4] R. Nieuwenhuis and A. Oliveras. *Proof-Producing Congruence Closure*, pages 453–468. Springer Berlin Heidelberg, 2005. ISBN 9783540320333. doi:10.1007/978-3-540-32033-3_33.

[5] R. Nieuwenhuis, A. Oliveras, and C. Tinelli. *Abstract DPLL and Abstract DPLL Modulo Theories*, pages 36–50. Springer Berlin Heidelberg, 2005. ISBN 9783540322757. doi:10.1007/978-3-540-32275-7_3.

[6] M. Rozanov and O. Strichman. Generating minimum transitivity constraints in p-time for deciding equality logic. *Electronic Notes in Theoretical Computer Science*, 198(2):3–17, May 2008. ISSN 1571-0661. doi:10.1016/j.entcs.2008.04.077.

[7] M. Willsey, C. Nandi, Y. R. Wang, O. Flatt, Z. Tatlock, and P. Panchekha. egg: Fast and extensible equality saturation. *Proceedings of the ACM on Programming Languages*, 5(POPL):1–29, Jan. 2021. ISSN 2475-1421. doi:10.1145/3434304.