# Boolector-reasonls

Shaowei Cai, Xindi Zhang and Bing He
State Key Laboratory of Computer Science,
Institute of Software, Chinese Academy of Sciences, Beijing, China
caisw@ios.ac.cn

***Abstract*—This note introduces a SMT solver that improves Boolector for solving QF-BV instances, and the new solver is named Boolector-reasonls. The method encodes QF-BV instances to SAT problems and solves them by SAT solvers. We try to improve the ablility of Boolector for difficult instances by using our recent SAT solver ReasonLS.**

## I. Introduction

We build our solver on top of the SMT solver Boolector[1], more exactly, Boolector Github master $<$ b9827374e5613cd3d6a29406b6ac76ebd763cdf4 $>$ [1]. The idea is simple and straightforward. Since Boolector solves QF-BV instances by calling a SAT solver, improvements on SAT solvers may lead to improvement on its performance. We modify Boolector by replacing the default SAT solver with our recent SAT solver ReasonLS for solving hard (indeed we mean not very easy) instances. Specifically, if the default SAT solver in Boolector fails to solve the instance within a preset time limit $t$ (which is short), then we use another SAT solver ReasonLS [2] to solve the instance. The idea of ReasonLS is to improve CDCL solvers by relaxing the backtracking and integrating a local search solver, aiming to improve its ability to find solutions for satisfiable instances.

## II. Adjusted ReasonLS

The main idea of ReasonLS is to relax the backtracking by protecting promising partial assignments from being pruned. Specifically, when the CDCL processes reaching a node corresponding to a promising assignment and meet some conditions, ReasonLS enters a no-backtracking phase, which uses unit propagation to assign the remaining variables without backtracking until all the variables are assigned, and then fed the complete assignment to a local search solver. If the local search solver cannot find a solution with certain time, ReasonLS will back to the node where it was interrupted. In Boolector-reasonls, we use the CDCL solver Maple_LCM_Dist_ChronoBT [3] and the local search solver CCAnr [4].

## III. Main Parameters

There is one parameter $p$ $(0 \leq p \leq 1)$ for controlling the cooperation of the backtracking style procedure and the local search solver. When $p = 1$, ReasonLS can be seen as a CDCL solver, and as a local search solver when $p = 0$. Meanwhile, we limit the time of each local search call to 300 seconds , and limit the total local search time to be no more than a propotion $\xi$ of the total time limit. Finally, parameter $t$ (seconds) controls when to replace the SAT solver: if the default SAT solver fails in $t$ time, then the ReasonLS solver is called to continue to solve the instance.

For our solvers, the parameters are set as follows.
$p = 0.9$; $\xi = 0.3$; $t = 9$.

## IV. SMT Competition 2019 Specifies

Our solver is submitted to "QF-BV in the Single Query Track" and "QF_BV in the Incremental track".

Specifically, our solver need to be compiled in the root folder by running "./starexec_build".

The running command is: "./starexec_run_default $1" in folder "./bin". The parameter $1 is the absolute path of input file. For a given input file "∼/sc/a.smt", the call command is "./starexec_run_default ∼/sc/a.smt ".

## V. Acknowledgement

We would like to thank the teams of Boolector, Maple_LCM_Dist_ChronoBT and all the teams in the references.

## References

[1] A. Niemetz, M. Preiner, C. Wolf, and A. Biere, "Btor2, btormc and boolector 3.0," in *International Conference on Computer Aided Verification*. Springer, 2018, pp. 587–595.

[2] S. Cai and X. Zhang, "ReasonLS," in *Proceeding of SAT Competition 2018*, 2018, pp. 52–53.

[3] V. Ryvchin and A. Nadel, "Maple_LCM_Dist_ChronoBT: Featuring Chronological Backtracking," in *Proceeding of SAT Competition 2018*, 2018, p. 29.

[4] S. Cai, C. Luo, and K. Su, "CCAnr: A configuration checking based local search solver for non-random satisfiability," in *Proceedings of18th International Conference on Theory and Applications of Satisfiability Testing, SAT 2015, Austin, TX, USA, September 24-27, 2015*, 2015, pp. 1–8.

[1]https://github.com/Boolector/boolector/tree/b9827374e5613cd3d6a29406b6ac76ebd763cdf4