Search-Space Partitioning for Parallelizing SMT Solvers

Matteo Marescotti, Antti E. J. Hyvärinen, and Natasha Sharigina

Faculty of Informatics, University of Lugano
Via Giuseppe Buffi 13, CH-6904 Lugano, Switzerland

The *Satisfiability Modulo Theories* (SMT) problem is the decision problem of determining whether a propositional formula is satisfiable, given that some of the variables have an interpretation with respect to background theories. The expressiveness of SMT makes it suitable for a vast range of application domains, and for that reason it has recently attained significant interest from both industry and academia.

The computational cost of SMT can be very high given that just the problem of determining the propositional satisfiability, the *Boolean Satisfiability Problem* (SAT), is proven to be NP-complete, and the introduction of the background theories can only make the problem harder. Nevertheless relatively little research exists on how parallel computing can be used to speed up SMT solving.

Two important approaches can be used to parallelize SMT solving: algorithm portfolio and the divide-and-conquer approach.

Algorithm portfolio is widely used because the heuristics that guides the solver's choices are intrinsically inaccurate, and small changes can result in significant differences in run times. In this work we introduce such changes by randomly initializing the seeds that each solver of the portfolio uses. However algorithm portfolio alone seems to hit a scalability limit where adding more CPUs does not provide more speed-up.

The divide-and-conquer approach partitions the search-space of the original instance; this results in many instances all different from each other. This approach leads to an inherent problem when the original instance is unsatisfiable: we must wait for all the instances to be solved in order to prove the unsatisfiability of the original one. Those instances, despite being usually easier than the original one, still might be challenging, and since we have to wait for the "unluckiest" one to be solved, an exclusive use of this approach can lead to worse performances.

In this work we address the challenge of parallelizing SMT solving introducing an abstract algorithmic framework called *parallelization tree*. The parallelization tree can be used to combine the two parallel approaches in order to exploit the positive aspects of both. We show experimentally, that several instantiations of the parallelization tree framework lead to an increment on the number of instances solved within a given amount of time.

All the experimentations have been made using the solver OpenSMT2, developed at the University of Lugano, that is capable of handling the theory of quantifier free uninterpreted functions with equalities (QF_UF). We run all the experiments in a cluster consisting of nodes with two AMD quad-core Opteron

2344 HE CPUs, and each node was running at most four solver instances. We used 16 nodes, hence 64 solver instances running concurrently.

We noted that the best instantiation of the parallelization tree is the one that divides the main instance in 8 instances, each of those solved with a portfolio made of 8 solvers; this approach also provides a general and significant speed-up on the other instances. Among our benchmark set, this parallel approach was able to solve 6 more instances within the timeout. The article with all the details has been accepted at the conference SAT2015 [1].

Acknowledgements. This work was financially supported by SNF project number 153402.

References

1. Hyvärinen, A.E.J., Marescotti, M., Sharygina, N.: Search-space partitioning for parallelizing SMT solvers (2015)