What is the State-of-the-Art in DQBF solving?

Gergely Kovásznai

IoT Research Center,
Eszterhazy Karoly University of Applied Sciences,
Eger, Hungary

kovasznai.gergely@ekf.hu

Dependency Quantified Boolean Formulas (DQBF) are obtained by adding Henkin quantifiers to Boolean formulas and have seen growing interest in the last years. In contrast to QBF, the dependencies of a variable in DQBF are explicitly specified instead of being implicitly defined by the order of the quantifier prefix. As a result, problem descriptions in DQBF can possibly be exponentially more succinct. While QBF is PSPACE-complete, DQBF was shown to be NEXPTIME-complete.

Many practical problems are known to be NEXPTIME-complete. This includes, e.g., partial information non-cooperative games or certain bit-vector logics [1]. More recently, DQBF has also been shown to offer a natural encoding for partial equivalence checking (PEC) problems [6].

Speaking of solving approaches, the first known direct solving approach for DQBF is an adaptation of QDPLL [7], which did not end up being efficient. Expansion-based techniques for DQBF were also investigated and yielded in an expansion-based solver in [6] that uses an underlying SAT solver. In [4], a refutational approach is proposed that is based on QBF abstraction, thus an underlying QBF solver is used. This approach can practically be used only for refuting unsatisfiable formulas. Since Effectively Propositional Logic (EPR) is another NEXPTIME-complete logic, we investigated how to adapt an instantiation-based EPR solving approach to DQBF, resulting a new instantiation-based DQBF solver, iDQ [5], that turned out to be an efficient solver. In [2], an elimination-based solving strategy is proposed and is implemented in the DQBF solver called HQS. Besides the powerful elimination strategy, HQS utilizes several optimizations, such as pure and unit literal detection, yields to an even more efficient DQBF solver than iDQ.

Apart from the solving technique we use, preprocessing techniques can speed up the solving significantly. Therefore it is worth to investigate if well-known SAT and QBF preprocessing techniques can be adapted to DQBF. Currently, such investigations were reported to be quite successful [3].

References

[1] G. Kovásznai, A. Fröhlich, A. Biere. Complexity of Fixed-Size Bit-Vector Logics. Theory of Computing Systems, Springer, pp. 1-54, 2015.

[2] K. Gitina, R. Wimmer, S. Reimer, M. Sauer, C. Scholl, B. Becker. Solving DQBF through Quantifier Elimination. Proc. DATE 2015, pp. 1617-1622, 2015.

[3] R. Wimmer, K. Gitina, J. Nist, C. Scholl, B. Becker. Preprocessing for DQBF. Proc. SAT 2015, pp. 173-190, 2015.

[4] B. Finkbeiner, L. Tentrup. Fast DQBF Refutation. Proc. SAT 2014, pp. 243-251, 2014.

[5] A. Fröhlich, G. Kovásznai, A. Biere, H. Veith. iDQ: Instantiation-Based DQBF Solving. Proc. POS 2014, aff. to SAT 2014, pp. 103-116, 2014.

[6] K. Gitina, S. Reimer, M. Sauer, R. Wimmer,C. Scholl, B. Becker. Equivalence checking of partial designs using dependency quantified Boolean formulae. Proc. ICCD'13, pp. 396-403, 2013.

[7] A. Fröhlich, G. Kovásznai, A. Biere. A DPLL Algorithm for Solving DQBF. Proc. POS'12, aff. to SAT 2012, 2012.