From DQBF to QBF by Dependency Elimination

– Extended Abstract –

Ralf Wimmer[1], Andreas Karrenbauer[2], Ruben Becker[2], Christoph Scholl[1], Bernd Becker[1]
[1] Albert-Ludwigs-Universität Freiburg
Freiburg im Breisgau, Germany
{wimmer, scholl, becker}@informatik.uni-freiburg.de

[2] MPI for Informatics, Saarland Informatics Campus
Saarbrücken, Germany
{karrenba, ruben}@mpi-sb.mpg.de

Abstract. Dependency quantified Boolean formulas (DQBFs) are a generalization of QBFs, which allows to have existential variables that depend on arbitrary subsets of the universal variables. We present an effective way to solve such DQBFs by eliminating individual dependencies such that an equisatisfiable QBF is obtained, which can be solved by an arbitrary QBF solver. We also present how to use dependency schemes in order to obtain smaller equisatisfiable QBFs, which can typically be solved more efficiently.

1. Introduction

Solver techniques have become one of the major workhorses in electronic design automation, in particular test and verification of digital circuits have significantly profited from extremely efficient solvers for the propositional satisfiability problem (SAT). While research on SAT has reached a certain degree of saturation, solving quantified propositional formulas has been a focus of current research during the last decade. We focus here on a generalization of the standard quantified Boolean formulas (QBFs), which are called *dependency quantified Boolean formulas (DQBFs)*. QBFs have the restriction that each existential variable depends on all universal variables in whose scope it is. For formulas in prenex normal form this leads to a linearly ordered quantifier prefix. This restriction is abolished in DQBF. Here each existential variable may depend on an arbitrary subset of the universal variables, its dependency set. The dependency sets of the existential variables are explicitly stated in the formula. Such formulas play an important role for modeling partial information, e. g., in games with incomplete information, circuit synthesis, and for verifying incomplete circuits.

Solving DQBFs is a hard problem – it is NEXPTIME complete, i. e., in the worst case, all algorithms that are currently known run in double exponential time. Different algorithms have been proposed to solve DQBFs: DPLL-style search, instantiation-based solving, and quantifier elimination.

Dependency elimination is an operation which allows to remove individual dependencies from the formula while preserving its satisfiability. The currently most successful way to solve DQBFs is

to simplify the formula in a preprocessing phase and then to eliminate dependencies such that an equisatisfiable QBF is obtained. The resulting QBF can be solved by an arbitrary QBF solver. This method is introduced in [1]. The conversion to QBF in general leads to an exponential blow-up of the formula: If one needs to eliminate n dependencies of an existential variable y, then the resulting formula contains up to 2^n different copies of this variable. Therefore, one must carefully select the dependencies that are eliminated to keep the blow-up as small as possible.

An elimination set is a set of dependencies of the formula whose elimination turns the DQBF into an equisatisfiable QBF. We call an elimination set optimal if its elimination yields a QBF with a minimum number of existential variables. We determine an optimal elimination set as a solution of an optimization problem on a bipartite tournament graph and show how to solve it efficiently. We have integrated this novel technique into the state-of-the-art DQBF solver HQS. Experiments show that dependency elimination improves the performance of the solver significantly.

References

[1] Ralf Wimmer, Andreas Karrenbauer, Ruben Becker, Christoph Scholl, and Bernd Becker. "From DQBF to QBF by Dependency Elimination". In: *Proc. of the 20th Int'l Conf. on Theory and Applicatons of Satisfiability Testing (SAT)*. Ed. by Serge Gaspers and Toby Walsh. Vol. 10491. Lecture Notes in Computer Science. Melbourne, Australia: Springer, Aug. 28–Sept. 1, 2017, pp. 326–343. DOI: 10.1007/978-3-319-66263-3_21.