

Satisfiability of DQBF Using Binary Decision Diagrams



Juraj Sič (supervisor: Jan Strejček)

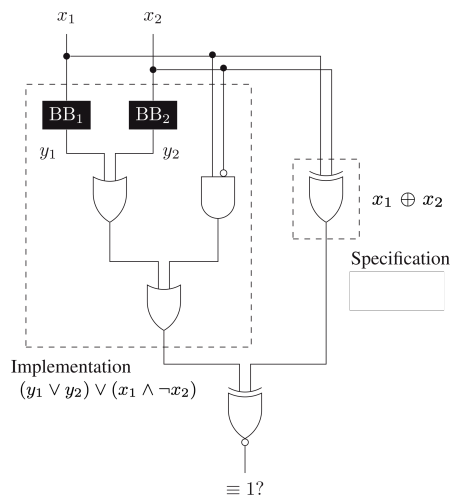
Faculty of Informatics, Masaryk University

Problem

- Decide satisfiability of a given DQBF
- DQBF = **dependency quantified Boolean formula**
 - Propositional logic formula extended with **quantifiers with explicit dependencies** between them
- NEXPTIME-complete problem
- Example:

$$\forall x_1 \forall x_2 \exists y_1(x_1) \exists y_2(x_2). (x_1 \wedge x_2) \Leftrightarrow (y_1 \Leftrightarrow y_2)$$

- y_1 depends only on x_1 (and y_2 only on x_2), meaning that the value of y_1 cannot change based on the value of x_2
- Formula is **unsatisfiable** as y_1 and y_2 cannot coordinate
- Can be used for solving
 - controller synthesis problem (CSP)**
 - partial equivalence checking (PEC)** – Can a combinational circuit with black boxes (BB) be equivalent to a given specification?

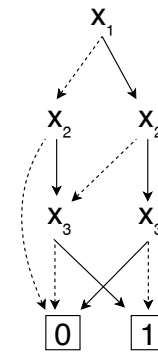


- Figure above [2] shows a PEC problem encoded by a DQBF

$$\forall x_1 \forall x_2 \exists y_1(x_1) \exists y_2(x_2). ((y_1 \vee y_2) \vee (x_1 \wedge \neg x_2)) \Leftrightarrow (x_1 \oplus x_2)$$

Method

- Quantifier elimination** is used as the basic solving technique
 - Quantifiers are iteratively eliminated until we end up with True or False
- Algorithm improved by **quantifier localisation**
 - Quantifiers are pushed inside the formula resulting in a faster elimination
- Binary decision diagrams (BDDs)** are used to represent propositional subformulas in DQBF
- The BDD on the right represents $(\neg x_1 \wedge x_2 \wedge x_3) \vee (x_1 \wedge \neg(x_2 \Leftrightarrow x_3))$



Results

Quantifier localisation improvements

- Correction of existing results
- Proved that it can be used in subformulas
- Proved that universal quantifier elimination can be done locally

Solver DQBDD

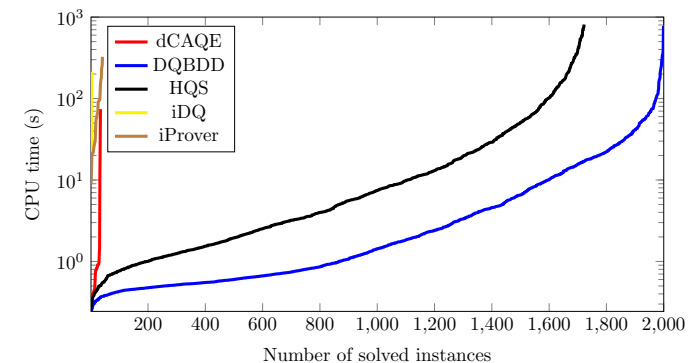
- New algorithm solving DQBF satisfiability
- Implemented in C++ using BDDs
- Winner** of the DQBF track of **QBFEval'20** competition [1]

Publications under preparation

- Joint journal paper with the HQS team (University of Freiburg)
- Publication about the new algorithm of DQBDD

Experiments

- Comparison of possible quantifier localisation and elimination strategies
- Comparison of DQBF solvers using different benchmarks
- Results:
 - DQBDD** is far **better** than other solvers for PEC
 - Figure below shows a cactus plot comparing runtimes of DQBF solvers for PEC instances



- DQBDD** is nearly as good as the best solvers for CSP

QBFEval'20 Competition

- Comparison of DQBF solvers on selected benchmarks
- Results
 - DQBDD – 257 solved in 5396 s
 - HQS – 195 solved in 2662 s
 - iProver – 170 solved in 17399 s

References

- QBF evaluation 2020. <http://www.qbflib.org/qbfeval20.php>.
- Karina Gitina et al. Equivalence checking of partial designs using dependency quantified boolean formulae. In **ICCD'13**.