# Scalable Certificate Extraction for QBF

Aina Niemetz, Mathias Preiner,
Florian Lonsing, Martina Seidl, and Armin Biere

Institute for Formal Models and Verification (FMV)
Johannes Kepler University, Linz, Austria
http://fmv.jku.at/

Alpine Verification Meeting (AVM),
May 21 - 22, 2012,
Passau, Germany

**Quantified Boolean Formulas (QBF)**

... extension of propositional logic (SAT) with quantifiers ($\forall$, $\exists$)

– satisfiability problem for QBF (QSAT) is PSPACE-complete

+ compact encodings for many real world problems
e.g., Formal Verification, Artificial Intelligence

**QBF Certificates**

- provide means to verify the correctness of a solver's result
- provide concrete solution as a base for
  e.g., counter-examples, error traces, strategies

**Skolem/Herbrand Function-based QBF Certificates**

- represent truth values of existential/universal variables
- provide strategies, counter-examples, error traces
- until recently: only Skolem functions derivable from Skolemization-based QBF solvers (e.g., sKizzo, Squolem)
  $\longrightarrow$ not as successful as search-based QBF solvers
  $\longrightarrow$ not maintained anymore
- novel approach presented at CAV'11 by Balabanov and Jiang [BJ11]
  $\longrightarrow$ extraction of Skolem/Herbrand functions from Q-resolution proofs

**Our Goal**

- verify correctness of a QBF solver's result
- extract concrete solutions instead of mere *sat*/*unsat* answers
  $\longrightarrow$ Skolem/Herbrand function-based certificates
- solver-independent framework for QBF certificate extraction

**Prenex Conjunctive Normal Form (PCNF)**

- $Q_1 X_1 \dots Q_n X_n.\phi$, where $\phi := \bigwedge C_i$ with clauses $C_i$ and $Q_i \in \{\exists, \forall\}$
- PCNF: Quantifier-free CNF $\phi$ over quantified Boolean variables
- $X_i$ ...set of quantified variables, linearly ordered: $Q_i X_i \leq Q_{i+1} X_{i+1}$
  $\longrightarrow$ variables in $X_i$ *precede* variables in $X_{i+1}$

**Prenex Disjunctive Normal Form (PDNF)**
... quantifier-free DNF over quantified Boolean variables (dual to PCNF)

**Semantics**

- $\forall x.\phi$ is satisfiable iff both $\phi[x/0]$ and $\phi[x/1]$ are satisfiable
- $\exists y.\phi$ is satisfiable iff either $\phi[y/0]$ or $\phi[y/1]$ is satisfiable

Theorem ([BKF95, GNT06])

*A QBF in PCNF (PDNF) is unsatisfiable (satisfiable) iff there exists a clause (cube) resolution sequence leading to the empty clause (cube).*

$\longrightarrow$ We refer to this sequence as **Q-resolution proof**.

### Definition (Universal Reduction)

Given a clause $C$, $UR(C) := C \setminus \{l_u \in L_\forall(C) \mid \nexists l_e \in L_\exists(C), l_u < l_e\}$, i.e., removing all universal literals that do not precede any existential literal in C.

### Example (UR)

Given PCNF $\exists x \forall y \exists z.(x \vee y \vee z) \wedge (\neg x \vee \neg y)$. Then, $UR((\neg x \vee \neg y)) = (\neg x)$.

### Definition (Q-Resolution)

Let $C_1$, $C_2$ be clauses with $v \in C_1$, $\neg v \in C_2$ and $q(v) = \exists$ [BKF95].

1. $C := (UR(C_1) \cup UR(C_2)) \setminus \{v, \neg v\}$.
2. If $\{x, \neg x\} \subseteq C$ (tautology), then no Q-resolvent exists.
3. Otherwise, Q-resolvent $C' := UR(C)$.

### Example (Q-Resolution)

Given PCNF $\exists x \forall y \exists z.(x \vee y \vee z) \wedge (\neg x \vee \neg y)$. Then, resolving $(x \vee y \vee z)$ and $(\neg x \vee \neg y)$ yields $(y \vee z)$.

**Skolemization/Skolem Functions** (PDNF)

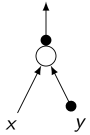
- technique for eliminating existential quantifiers
- $\exists$-variables are substituted by so-called *Skolem functions*
  $\longrightarrow$ truth value of $\exists$-variable is defined over all preceding $\forall$-variables
- resulting formula . . .
  - contains $\forall$-variables only
  - is satisfiable iff original formula is satisfiable

**Herbrandization/Herbrand Functions** (PCNF)

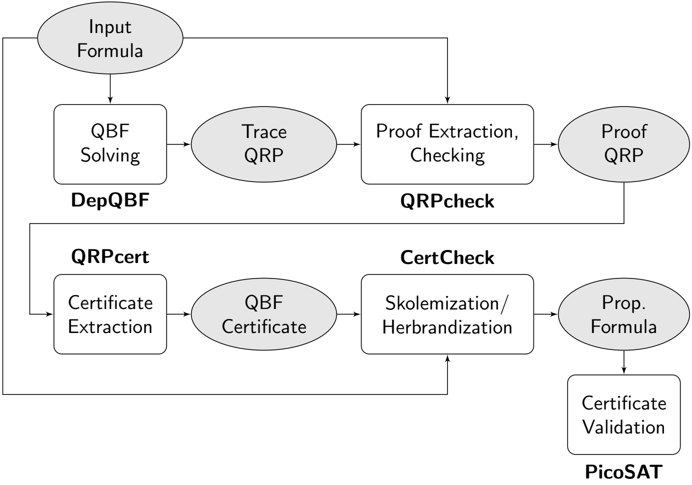
- technique for eliminating universal quantifiers (dual to Skolemization)

**And-Inverter Graphs (AIG)**

- directed acyclic graph (DAG)
- representation of circuits/Boolean formulas
- logical connectives: and ($\wedge$), negation ($\neg$)
- allow sharing of isomorphic subgraphs




$$x \rightarrow y \equiv \neg(x \wedge \neg y)$$


Input Formula
QBF Solving
DepQBF
Trace QRP
Proof Extraction, Checking
QRPcheck
Proof QRP
QRPcert
Certificate Extraction
QBF Certificate
CertCheck
Skolemization/ Herbrandization
Prop. Formula
Certificate Validation
PicoSAT

**DepQBF** [LB10]

- search-based state-of-the-art QBF solver
- for QBF in PCNF
- implements DLL algorithm for QBF (QDLL) [CGS98]
- placed 1st in main track of QBFEVAL'10

**Tracing in DepQBF**

- on top of QDLL with Learning
- records
    - input formula
    - each learnt constraint (clauses resp. cubes) and its antecedent(s)
    - derivation of the empty constraint
    - result (sat, unsat)
- in QRP format

**QRPcheck**
. . . tool for extracting and checking proofs in QRP format

- extract proof from trace on-the-fly, starting with the empty constraint
- check each proof step incrementally
- set of input constraints for deriving the empty constraint
  - **unsatisfiable**: subset of the input formula
    $\longrightarrow$ considered as given
  - **satisfiable**: set of learnt cubes generated by the solver
    $\longrightarrow$ checked individually
- provides possibility to extract QRP representation of proof

**QRPcert**

…tool for extracting Skolem/Herbrand function-based QBF certificates from Q-resolution proofs and traces in QRP format

- Skolem/Herbrand function extraction based on algorithm presented by Balabanov and Jiang [BJ11]
- Skolem/Herbrand functions are represented as AIGs
- employs structural sharing on AIGs
- set of extracted Skolem/Herbrand functions represents QBF certificate
  - QBF satisfiable: Skolem function-based QBF certificate
  - QBF unsatisfiable: Herbrand function-based QBF certificate

**CertCheck**
... tool for merging the input formula with the corresponding certificate AIG

1. translate input formula into an AIG
2. substitute $\exists/\forall$-variables with corresponding Skolem/Herbrand functions
   $\longrightarrow$ merge input formula AIG with certificate AIG
3. translate resulting (merged) AIG into prop. formula $\phi$ in CNF

**Certificate Validation**
... check prop. formula $\phi$ with a SAT solver

- QBF satisfiable: merged AIG contains $\forall$-variables only
  $\longrightarrow$ check if $\phi$ is tautological
- QBF unsatisfiable: merged AIG contains $\exists$-variables only
  $\longrightarrow$ check if $\phi$ is unsatisfiable

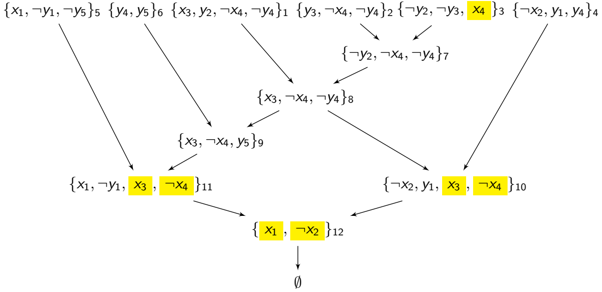
# Certificate Extraction Example

## Q-Resolution Proof DAG

**Input formula:**

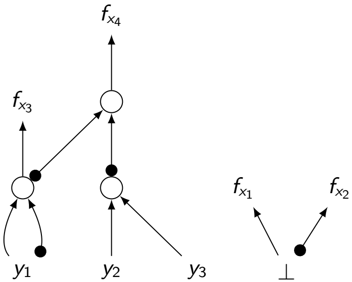
$\forall x_1 x_2 \exists y_1 \forall x_3 \exists y_2 y_3 \forall x_4 \exists y_4 y_5 . (x_1 \vee \neg y_1 \vee \neg y_5) \wedge (y_4 \vee y_5) \wedge (x_3 \vee y_2 \vee \neg x_4 \vee \neg y_4) \wedge (y_3 \vee \neg x_4 \vee \neg y_4) \wedge (\neg y_2 \vee \neg y_3 \vee x_4) \wedge (\neg x_2 \vee y_1 \vee y_4)$

**Q-Resolution Proof DAG:**




**Extracted Herbrand Functions:**

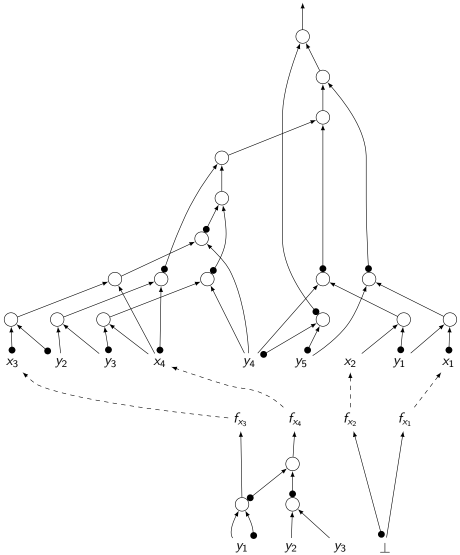
$f_{x_4} = \{UR(3), \neg UR(11), \neg UR(10)\} = UR(3) \wedge (\neg UR(11) \vee \neg UR(10)) = (\neg y_2 \vee \neg y_3) \wedge ((\neg x_1 \wedge y_1) \vee (x_2 \wedge \neg y_1))$
$f_{x_3} = \{UR(11), UR(10)\} = UR(11) \wedge UR(10) = (x_1 \vee \neg y_1) \wedge (\neg x_2 \vee y_1)$
$f_{x_2} = \{\neg UR(12)\} = \neg \emptyset = \top$
$f_{x_1} = \{UR(12)\} = \emptyset = \bot$




$$
\begin{aligned}
f_{x_1} &= \perp \\
f_{x_2} &= \top \\
f_{x_3} &= \neg y_1 \wedge y_1 \\
f_{x_4} &= (\neg y_2 \vee \neg y_3) \wedge (y_1 \vee \neg y_1)
\end{aligned}
$$

# Certificate Extraction Example

## Merging Input Formula and Certificate AIG



# Experimental Results

QBFEVAL'10 set (568 formulas), limits: 7 GB memory, 1800 seconds time

**Proof Extraction and Checking**

- 362 instances solved by DepQBF, 348 checked by QRPcheck
- difference: 14 instances due to memory out
- required 35% of solving time

**Certificate Extraction**

- out of 348 proofs, 337 certificates extracted
- difference: 11 certificates due to memory out
- avg. number of AND-gates: 20M (sat.), 170k (unsat.)
- avg. % of AIG compression: 65% (sat.), 23% (unsat.)
- required 41% of solving time

**Skolemization/Herbrandization**

- avg. number of clauses: 59M (sat.), 409k (unsat.)
- required 32% of solving time

**Certificate Validation**

- out of 337 prop. formulas, 275 were checked successfully
- difference: 45 (17) certificates not validated due to memory (time) out
- required 88% of solving time

# Conclusion

**Summary**

- framework for complete certification of QBF
- solver-independent tools for . . .
  - extracting/checking Q-resolution proofs
  - extracting/validating QBF Skolem/Herbrand function-based certificates
- Skolem/Herbrand function-based QBF certificates as a base for, e.g., counter-examples in model checking, strategies in AI
- certificates for over 93% of solved instances extracted
  $\longrightarrow$ 100% when lifting memory limit

**Open Problems/Challenges**

- trace file size (several GB on avg.)
- certificate validation bottleneck in certification workflow
  $\longrightarrow$ employ incremental SAT checking
  $\longrightarrow$ improve AIG-to-CNF translation
- support more AIG simplification techniques
- support for advanced dependency schemes as employed in DepQBF

Valeriy Balabanov and Jie-Hong R. Jiang.
Resolution Proofs and Skolem Functions in QBF Evaluation and Applications.
In *Proc. of the 23rd International Conference on Computer Aided Verification (CAV 2011)*, volume 6806 of *Lecture Notes in Computer Science*, pages 149–164. Springer, 2011.

Hans Kleine Büning, Marek Karpinski, and Andreas Flögel.
Resolution for Quantified Boolean Formulas.
*Information and Computation*, 117(1):12–18, 1995.

M. Cadoli, A. Giovanardi, and M. Schaerf.
An Algorithm to Evaluate Quantified Boolean Formulae.
In *AAAI/IAAI*, pages 262–267, 1998.

Enrico Giunchiglia, Massimo Narizzano, and Armando Tacchella.
Clause/Term Resolution and Learning in the Evaluation of Quantified Boolean Formulas.
*Journal of Artificial Intelligence Research (JAIR)*, 26:371–416, 2006.

F. Lonsing and A. Biere.
DepQBF: A Dependency-Aware QBF Solver.
*JSAT*, 7(2-3):71–76, 2010.