An Overview of QBF Reasoning Techniques

Florian Lonsing

Knowledge-Based Systems Group, Vienna University of Technology, Austria
http://www.kr.tuwien.ac.at/staff/lonsing/

Dagstuhl Seminar 16381: SAT and Interactions
September 18-23 2016, Schloss Dagstuhl, Leibniz-Zentrum für Informatik

This work is supported by the Austrian Science Fund (FWF) under grant S11409-N23.

Introduction (1)

Quantified Boolean Formulas (QBF):

- Existential ($\exists$) / universal ($\forall$) quantification of propositional variables.
- Propositional CNF with linearly ordered quantifier prefix.
- QBF satisfiability: PSPACE-completeness.
- Potentially more succinct encodings than propositional logic.
- Applications to presumably harder problems, e.g. NEXPTIME.

Example

- CNF $\phi := (\bar{u} \vee x) \wedge (u \vee \bar{x})$.
- Quantifier prefix $\hat{Q} := \forall u \exists x$.
- QBF $\psi := \hat{Q}.\phi$ in *prenex conjunctive normal form (PCNF)*.
- $\psi = \forall u \exists x.(\bar{u} \vee x) \wedge (u \vee \bar{x})$.

Introduction (2)

Recursive Semantics:

- Assume that a QBF does not contain free variables.
- The QBF $\perp$ is unsatisfiable, the QBF $\top$ is satisfiable.
- The QBF $\neg(\psi)$ is satisfiable iff the QBF ψ is unsatisfiable.
- The QBF $\psi_1 \wedge \psi_2$ is satisfiable iff ψ_1 and ψ_2 are satisfiable.
- The QBF $\psi_1 \vee \psi_2$ is satisfiable iff ψ_1 or ψ_2 is satisfiable.
- The QBF $\forall x.(\psi)$ is satisfiable iff $\psi[\neg x]$ and $\psi[x]$ are satisfiable. The QBF $\psi[\neg x]$ ($\psi[x]$) results from ψ by replacing x in ψ by $\perp$ ($\top$).
- The QBF $\exists x.(\psi)$ is satisfiable iff $\psi[\neg x]$ or $\psi[x]$ is satisfiable.

Example

$\psi = \forall u \exists x.(\bar{u} \vee x) \wedge (u \vee \bar{x})$ satisfiable iff

- $\psi[\bar{u}] = \exists x.(\bar{x})$ satisfiable and
- $\psi[u] = \exists x.(x)$ satisfiable.

Introduction (3): Success Story of QBF Solving?

[MVB10] Hratch Mangassarian, Andreas G. Veneris, Marco Benedetti: Robust QBF Encodings for Sequential Circuits with Applications to Verification, Debug, and Test. IEEE Trans. Computers 59(7), 2010.

> *Admittedly, the theory and results of this paper emphasize the need for further research in QBF solvers [...] Since the first complete QBF solver was presented decades after the first complete engine to solve SAT, research in this field remains at its infancy.*

See e.g. [BM08] for references to further comparisons of SAT and QBF.

Introduction (4)

The Beginning of QBF Solving:

- 1998: backtracking DPLL for QBF [CGS98].
- 2002: clause learning for QBF (proofs) [GNT02, Let02, ZM02a].
- 2002: expansion (elimination) of variables [AB02].

$\Rightarrow$ compared to SAT (1960s), QBF still is a young field of research!

Introduction (5): Progress in QBF Research

Increased Interest in QBF:

- QBF proof systems: theoretical frameworks of solving techniques.
- CDCL (clause learning) and expansion: orthogonal solving approaches.
- QBF solving by counterexample guided abstraction refinement (CEGAR) [CGJ$^+$03, JM15b, JKMSC16, RT15].
- QBFEVAL'16: largest number of participants ever.
- 10 QBF-related papers at SAT 2016 conference (27%).

QBF Research Community:

- QBFEVAL'16: `http://www.qbflib.org/qbfeval16.php`
- QBF Workshop 2016: `http://fmv.jku.at/qbf16/`
- Beyond NP Workshop: `http://beyondnp.org/`

Introduction (6): Motivating QBF Applications

Synthesis and Realizability of Distributed Systems:

[GT14] A. Gascón, A. Tiwari: A Synthesized Algorithm for Interactive Consistency. NASA Formal Methods 2014.

[FT15] B. Finkbeiner, L. Tentrup: Detecting Unrealizability of Distributed Fault-tolerant Systems. Logical Methods in Computer Science 11(3) (2015).

Solving Dependency Quantified Boolean Formulas (NEXPTIME):

[FT14] B. Finkbeiner, L. Tentrup: Fast DQBF Refutation. SAT 2014.

Formal Verification and Synthesis:

[HSM$^+$14] T. Heyman, D. Smith, Y. Mahajan, L. Leong, H. Abu-Haimed: Dominant Controllability Check Using QBF-Solver and Netlist Optimizer. SAT 2014.

[CHR16] C. Cheng, Y. Hamza, H. Ruess: Structural Synthesis for GXW Specifications. CAV 2016.

Outline

1. The beginning of QBF solving: QDPLL and variable expansion.
2. Modern approaches: QCDCL and CEGAR-based expansion.
3. Open problems and future research directions.

Part 1:
The Beginning of QBF Solving

Expansion (1)

$$\psi_0 \rightsquigarrow \psi_1 \rightsquigarrow \psi_2 \rightsquigarrow \ldots \rightsquigarrow \psi_n = \bot / \top$$

- Successively eliminate variables from a given PCNF ψ_0.
- Elimination produces satisfiability-equivalent PCNFs $\psi_i \equiv_{sat} \psi_{i+1}$.
- Worst case exponential space procedure.
- Redundancy elimination on ψ_i (depending on formula representation).
- Stop if ψ_i reduces to truth constant $\top$ or $\bot$.
- Invoke a SAT solver if ψ_i contains only $\exists$-variables.

Expansion (2)

Example

$\psi = \exists x \forall u \exists y.\ (\bar{x} \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{u} \vee y) \wedge (u \vee \bar{y})$

- Eliminate y: $\psi = \exists x \forall u.\ \underbrace{[(\bar{x}) \wedge (\bar{u})]}_{y \text{ replaced by } \perp} \vee \underbrace{[(x) \wedge (u)]}_{y \text{ replaced by } \top}$
- Convert to PCNF: $\psi = \exists x \forall u.\ (\bar{x} \vee x) \wedge (\bar{x} \vee u) \wedge (x \vee \bar{u}) \wedge (u \vee \bar{u})$

Expansion of $\exists$-Variables: cf. [AB02, Bie04]

- Eliminate rightmost variables by Shannon expansion [Sha49].
- Replace $\hat{Q}\exists x.\phi$ by $\hat{Q}.(\phi[x/\perp] \vee \phi[x/\top])$.
- Based on CNF, NNF, and-inverter graphs [AB02, LB08, PS09].
- If ϕ in CNF:
  - Similar to DP algorithm (add all possible resolvents of x).
  - Delete literals of innermost universal variables ("universal reduction").

Expansion (3)

Definition ([BKF95])

Given a clause C, *universal reduction (UR)* on C produces the clause

$$UR(C) := C \setminus \{l \in C \mid q(l) = \forall \text{ and } \forall l' \in C \text{ with } q(l') = \exists : l' < l\},$$

where $<$ is the linear variable ordering given by the quantifier prefix.

- UR shortens clauses by deleting “trailing” universal literals.
- UR is central in QBF proof systems, cf. [BBC16].

Example (continued)

- Eliminate y: $\psi = \exists x \forall u. \underbrace{[(\bar{x}) \wedge (\bar{u})]}_{y \text{ replaced by } \perp} \vee \underbrace{[(x) \wedge (u)]}_{y \text{ replaced by } \top}$
- Convert to PCNF: $\psi = \exists x \forall u. (\bar{x} \vee x) \wedge (\bar{x} \vee u) \wedge (x \vee \bar{u}) \wedge (u \vee \bar{u})$
- Simplify and reduce u: $\psi = \exists x. (\bar{x}) \wedge (x)$

Expansion (4)

Example (continued)

$\psi = \exists x \forall u \exists y.\ (\bar{x} \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{u} \vee y) \wedge (u \vee \bar{y})$

- Expand u: copy CNF and replace y by fresh z in copy of CNF.
- $\psi = \exists x, y, z.\ \underbrace{(\bar{x} \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{y})}_{u \text{ replaced by } \bot} \wedge \underbrace{(\bar{x} \vee z) \wedge (x \vee \bar{z}) \wedge (z)}_{u \text{ replaced by } \top,\ y \text{ replaced by } z}$
- Obtain $(\bar{x})$ from $(\bar{x} \vee y)$ and $(\bar{y})$, (x) from $(x \vee \bar{z})$ and (z).

Expansion of $\forall$-Variables: cf. [AB02, Bie04]

- Eliminate all universal variables by Shannon expansion.
- Finally, apply propositional resolution (no universal reduction).
- If x innermost: replace $\hat{Q}\forall x.\phi$ by $\hat{Q}.(\phi[x/\bot] \wedge \phi[x/\top])$.
- Otherwise, duplicate existential variables inner to x [Bie04, BK07].

Backtracking Search (1)

- DPLL algorithm [DLL62] for QBF: QDPLL [CGS98, CSGG02].
- Chronological backtracking (QBF semantics), nonrecursive in practice.

```
bool qdpll (PCNF Qxψ, Assignment A)
   /* 1. Simplify under given assignment. */
       ψ' := simplify(Qxψ[A]);
   /* 2. Check base cases. */
       if (ψ' == ⊥)
         return false;
       if (ψ' == ⊤)
         return true;
   /* 3. Decision making, backtracking. */
       if (Q == ∃)
         return qdpll (ψ', A ∪ {¬x}) ||
                qdpll (ψ', A ∪ {x});
       if (Q == ∀)
         return qdpll (ψ', A ∪ {¬x}) &&
                qdpll (ψ', A ∪ {x});
```

Backtracking Search (2): Optimizations

- Goal: avoid making assignments by decisions.
  - Decisions open branches in search tree.
  - Decisions have to be made in prefix order.
- Universal reduction:
  - Detect unit and empty clauses earlier (implicitly in original QDPLL).
- Unit literal detection (UL):
  - A clause $C' = (l)$ with $C \in \psi$ and $q(l) = \exists$ is unit.
- Pure literal detection (PL):
  - A literal l is pure in ψ if $\bar{l}$ does not occur in ψ. Assign $var(l)$ wrt. $\forall/\exists$.

Example

$$\psi := \exists x \forall u \exists y.(y) \wedge (x \vee u \vee \bar{y}) \wedge (\bar{x} \vee \bar{u})$$

$$\psi := \exists x \forall u \exists y.(y) \wedge (x \vee u \vee \bar{y}) \wedge (\bar{x})$$

$$\psi[\{\bar{u}\}] := \exists x \exists y.(y) \wedge (x \vee \bar{y}) \wedge (\bar{x})$$

$$\psi[\{\bar{u}, \bar{x}, y\}] := \bot$$

Backtracking Search (3): Optimizations

- Goal: close branches in search tree early and backtrack.
- Use of SAT solving in QDPLL.
- Trivial falsity:
  - Obtain CNF ψ' from PCNF ψ by treating every variable as $\exists$.
  - If ψ' is unsatisfiable then also ψ is unsatisfiable.
- Trivial truth:
  - Obtain CNF ψ' from PCNF ψ by deleting all $\forall$-literals.
  - If ψ' is satisfiable then also ψ is satisfiable.

Example (continued)

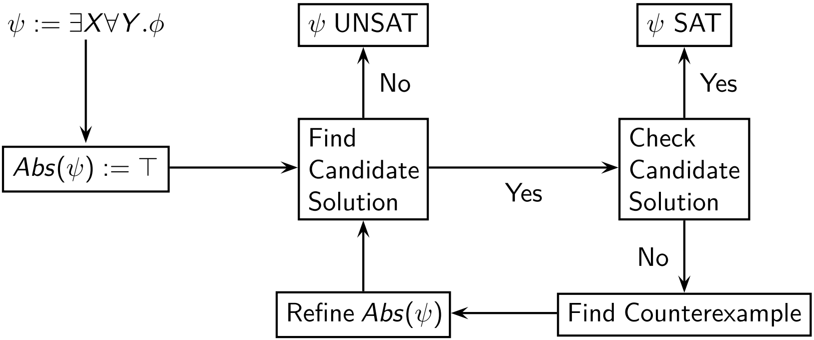
$$\psi := \exists x \forall u \exists y.(y) \wedge (x \vee u \vee \bar{y}) \wedge (\bar{x} \vee \bar{u})$$

Trivial falsity test:
$\psi' := \exists x \exists u \exists y.(y) \wedge (x \vee u \vee \bar{y}) \wedge (\bar{x} \vee \bar{u})$ is satisfiable.

Trivial truth test:
$\psi' := \exists x \exists y.(y) \wedge (x \vee \bar{y}) \wedge (\bar{x})$ is unsatisfiable.

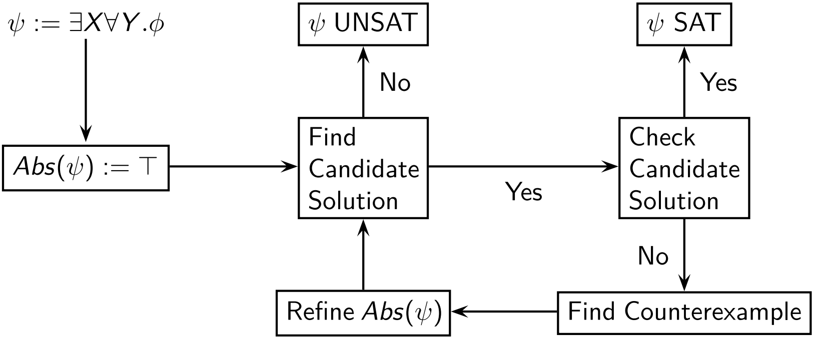
Part 2: Modern Approaches

Lazy Expansion by CEGAR



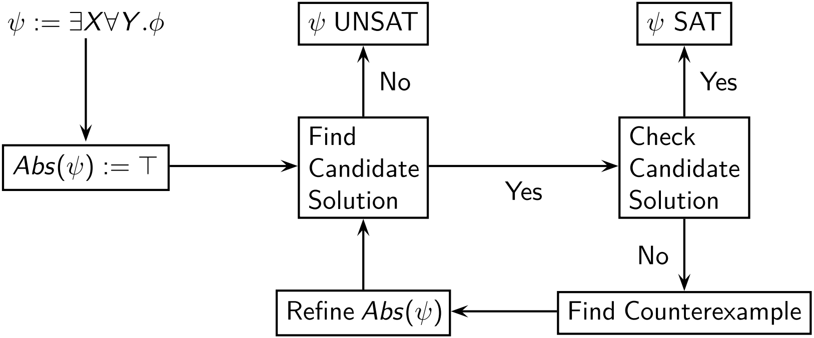
- Let $\psi := \exists X \forall Y.\ \phi$ be a one-alternation QBF, ϕ a non-CNF formula.
- ψ is satisfiable iff $\psi' := \exists X.(\bigwedge_{\mathbf{y} \in \mathcal{B}^{|Y|}} \phi[Y/\mathbf{y}])$ is satisfiable.
- Full expansion ψ' of $\forall Y$ by set $\mathcal{B}^{|Y|}$ of all possible assignments $\mathbf{y}$ of Y.
- Idea: consider a partial expansion of $\forall Y$ as an abstraction of ψ'.

Lazy Expansion by CEGAR



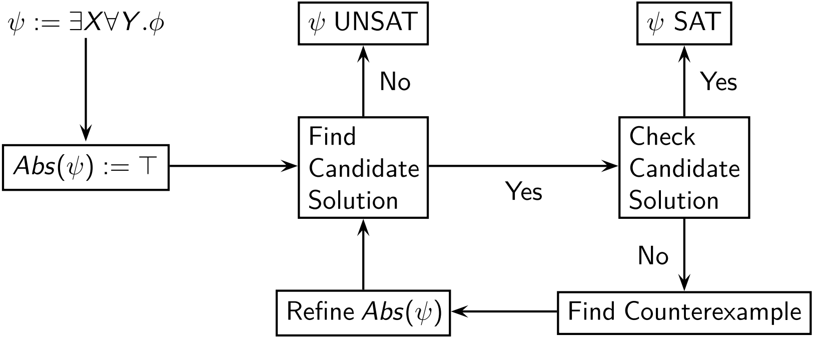
- Subset $U \subseteq \mathcal{B}^{|Y|}$ of set $\mathcal{B}^{|Y|}$ of all possible assignments $\mathbf{y}$ of Y.
- Partial expansion: given U, define $Abs(\psi) := \exists X . (\bigwedge_{\mathbf{y} \in U} \phi[Y/\mathbf{y}])$.
- Abstraction $Abs(\psi)$: if $Abs(\psi)$ unsatisfiable, then also ψ unsatisfiable.
- Initially, set $U := \emptyset$ and $Abs(\psi) := \top$.

Lazy Expansion by CEGAR



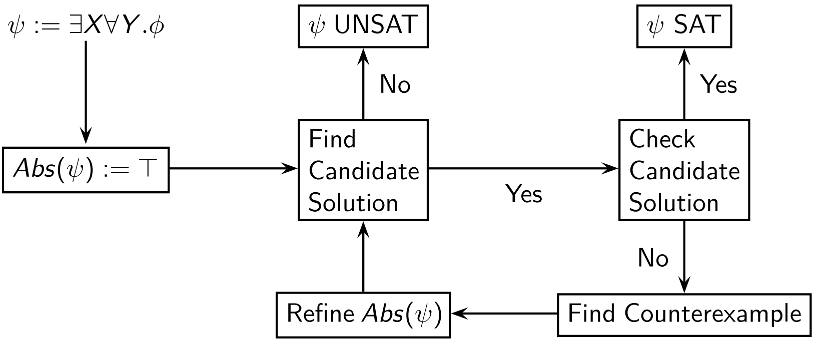
- Check satisfiability of $Abs(\psi)$ using a SAT solver.
- If $Abs(\psi)$ unsatisfiable: also ψ unsatisfiable, terminate.
- If $Abs(\psi)$ satisfiable: let $\mathbf{x} \in \mathcal{B}^{|X|}$ be a model of $Abs(\psi)$.
- $\mathbf{x} \in \mathcal{B}^{|X|}$: candidate solution of full exp. $\psi' := \exists X.(\bigwedge_{\mathbf{y} \in \mathcal{B}^{|Y|}} \phi[Y/\mathbf{y}])$.

Lazy Expansion by CEGAR



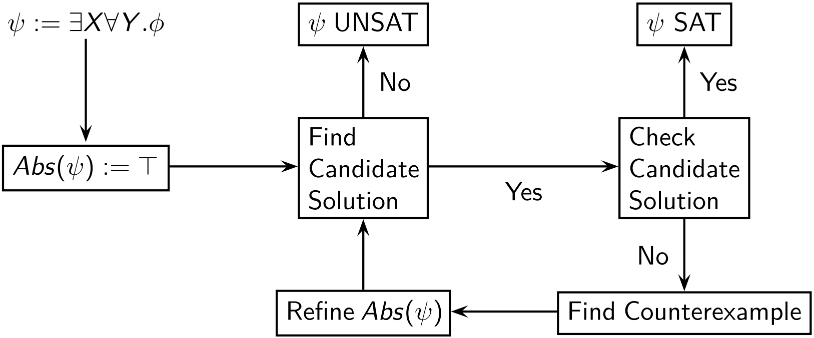
- If $\mathbf{x}$ is also a model of the full expansion ψ', then ψ is satisfiable.
- $\mathbf{x}$ is a model of full expansion ψ' iff $\forall Y.\phi[X/\mathbf{x}]$ is satisfiable.
- $\forall Y.\phi[X/\mathbf{x}]$ is satisfiable iff $\exists Y.\neg\phi[X/\mathbf{x}]$ is unsatisfiable.
- Check satisfiability of $\exists Y.\neg\phi[X/\mathbf{x}]$ using a SAT solver.

Lazy Expansion by CEGAR



- If $\exists Y.\neg\phi[X/\mathbf{x}]$ unsatisfiable: ψ is satisfiable, return $\mathbf{x}$ and terminate.
- If $\exists Y.\neg\phi[X/\mathbf{x}]$ satisfiable: let $\mathbf{y} \in \mathcal{B}^{|Y|}$ be a model of $\exists Y.\neg\phi[X/\mathbf{x}]$.
- Note: $\mathbf{y}$ is an assignment to $\forall$-variables in ψ.
- $\mathbf{y}$ is a counterexample to candidate solution $\mathbf{x}$ of full expansion ψ'.

Lazy Expansion by CEGAR



- Refine abstraction $Abs(\psi)$ by counterexample $\mathbf{y}$.
- Let $U := U \cup \{\mathbf{y}\}$ and $Abs(\psi) := \exists X.(\bigwedge_{\mathbf{y} \in U} \phi[Y/\mathbf{y}])$.
- Adding $\mathbf{y}$ to $Abs(\psi)$ prevents repetition of candidate solution $\mathbf{x}$.
- Used for 2QBF [RTM04, BJS$^+$16], RAReQS (recursive) [JKMSC16].

Q-Resolution (1)

Definition (Q-Resolution Calculus QRES, c.f. [BKF95])

Let $\psi = \hat{Q}.\phi$ be a PCNF and C, C_1, C_2 clauses.

$$\frac{}{C} \quad \text{for all } x \in \hat{Q}\text{: } \{x, \bar{x}\} \not\subseteq C \text{ and } C \in \phi \qquad (\mathit{init})$$

$$\frac{C \cup \{l\}}{C} \quad \begin{array}{l} \text{for all } x \in \hat{Q}\text{: } \{x, \bar{x}\} \not\subseteq (C \cup \{l\}),\ q(l) = \forall, \text{ and} \\ l' < l \text{ for all } l' \in C \text{ with } q(l') = \exists \end{array} \qquad (\mathit{red})$$

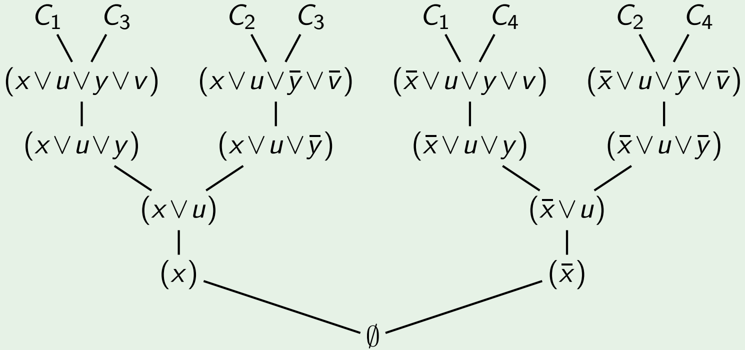
$$\frac{C_1 \cup \{p\} \qquad C_2 \cup \{\bar{p}\}}{C_1 \cup C_2} \quad \begin{array}{l} \text{for all } x \in \hat{Q}\text{: } \{x, \bar{x}\} \not\subseteq (C_1 \cup C_2), \\ \bar{p} \notin C_1,\ p \notin C_2, \text{ and } q(p) = \exists \end{array} \qquad (\mathit{res})$$

- Axiom *init*, universal reduction *red*, resolution *res*.
- PCNF ψ is unsatisfiable iff empty clause $\emptyset$ can be derived by QRES.

Q-Resolution (2)

Example

$\psi = \exists x \forall u \exists y \forall v \exists z.$
$\underbrace{(y \vee v \vee z)}_{C_1} \wedge \underbrace{(\bar{y} \vee \bar{v} \vee z)}_{C_2} \wedge \underbrace{(x \vee u \vee \bar{z})}_{C_3} \wedge \underbrace{(\bar{x} \vee u \vee \bar{z})}_{C_4} \wedge \underbrace{(\bar{x} \vee \bar{u} \vee \bar{z})}_{C_5}$



Q-Resolution (3)

Example (continued)

$\psi = \exists x \forall u \exists y \forall v \exists z.$

$\underbrace{(y \vee v \vee z)}_{C_1} \wedge \underbrace{(\bar{y} \vee \bar{v} \vee z)}_{C_2} \wedge \underbrace{(x \vee u \vee \bar{z})}_{C_3} \wedge \underbrace{(\bar{x} \vee u \vee \bar{z})}_{C_4} \wedge \underbrace{(\bar{x} \vee \bar{u} \vee \bar{z})}_{C_5}$

$$\begin{array}{ccc} C_1 & & C_2 \\ & \diagdown \;\; \diagup & \\ & (v \vee \bar{v} \vee z) & \end{array}$$

Long-Distance Q-Resolution: [ZM02a, BJ12]

- Like Q-resolution, but allow certain tautological resolvents.
- Tautological resolvent C with $\{x, \bar{x}\} \subseteq C$:
  - $q(x) = \forall$
  - Existential pivot p: $p < x$ in prefix ordering.
- Exponentially stronger than traditional Q-resolution.

Q-Resolution (3)

Example (continued)

$\psi = \exists x \forall u \exists y \forall v \exists z.$
$\underbrace{(y \vee v \vee z)}_{C_1} \wedge \underbrace{(\bar{y} \vee \bar{v} \vee z)}_{C_2} \wedge \underbrace{(x \vee u \vee \bar{z})}_{C_3} \wedge \underbrace{(\bar{x} \vee u \vee \bar{z})}_{C_4} \wedge \underbrace{(\bar{x} \vee \bar{u} \vee \bar{z})}_{C_5}$

$$\begin{array}{ccc} C_4 & & C_5 \\ & \diagdown \quad \diagup & \\ & (\bar{x} \vee \bar{z}) & \end{array}$$

QU-Resolution: [VG12]

- Like Q-resolution but additionally allow universal variables as pivots.
- Exponentially stronger than traditional Q-resolution.

Q-Resolution (3)

Example (continued)

$\psi = \exists x \forall u \exists y \forall v \exists z.$

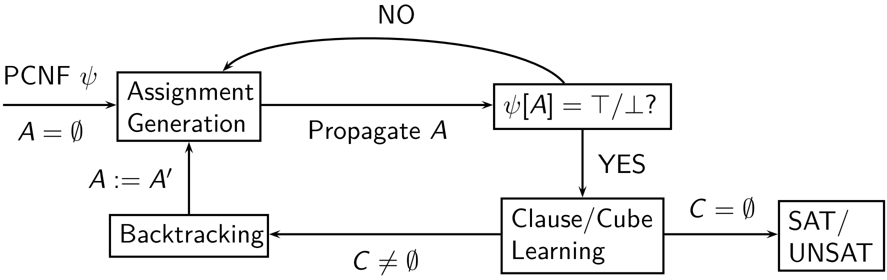
$\underbrace{(y \vee v \vee z)}_{C_1} \wedge \underbrace{(\bar{y} \vee \bar{v} \vee z)}_{C_2} \wedge \underbrace{(x \vee u \vee \bar{z})}_{C_3} \wedge \underbrace{(\bar{x} \vee u \vee \bar{z})}_{C_4} \wedge \underbrace{(\bar{x} \vee \bar{u} \vee \bar{z})}_{C_5}$

$$\begin{array}{ccc} C_4 & & C_5 \\ & \searrow \swarrow & \\ & (\bar{x} \vee \bar{z}) & \end{array}$$

Further Variants: [BWJ14]

- Combinations of QU- and long-distance Q-resolution.
- Existential and universal pivots, tautologies due to universal variables.

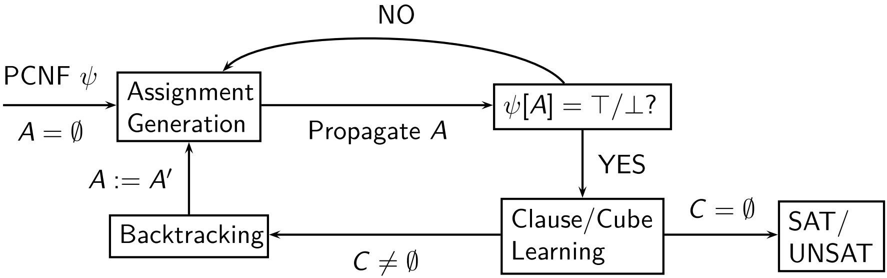
QCDCL (1)



High-Level Workflow:

- Assign *decision variables* starting at left end of prefix of $\psi[A]$.
- Propagation: simplify ψ under A and universal reduction.
- Conflict: $\psi[A] = \bot$: CNF ϕ contains a falsified clause.
- Solution: $\psi[A] = \top$: all clauses in CNF of ψ satisfied.

QCDCL (1)



High-Level Workflow:

- Clause (cube) learning based on Q-resolution.
- Asserting clause (cube) C: $C[A']$ unit for some $A' \subseteq A$.
- Empty clause (cube) $C = \emptyset$: formula proved UNSAT (SAT).
- QCDCL solvers, e.g., [LB10, GMN10, KSGC10, ZM02b]

QCDCL (2)

Example (Clause Learning)

- $\psi = \exists x_1, x_3, x_4 \forall y_5 \exists x_2.$
  $(\bar{x}_1 \vee x_2) \wedge (x_3 \vee y_5 \vee \bar{x}_2) \wedge (x_4 \vee \bar{y}_5 \vee \bar{x}_2) \wedge (\bar{x}_3 \vee \bar{x}_4)$
- Make decision $A = \{x_1\}$:
  $\psi[\{x_1\}] = \exists x_3, x_4 \forall y_5 \exists x_2.(x_2) \wedge (x_3 \vee y_5 \vee \bar{x}_2) \wedge (x_4 \vee \bar{y}_5 \vee \bar{x}_2) \wedge (\bar{x}_3 \vee \bar{x}_4)$
- By UL: $\psi[\{x_1, x_2\}] = \exists x_3, x_4 \forall y_5.(x_3 \vee y_5) \wedge (x_4 \vee \bar{y}_5) \wedge (\bar{x}_3 \vee \bar{x}_4)$.
- By UR: $\psi[\{x_1, x_2\}] = \exists x_3, x_4.(x_3) \wedge (x_4) \wedge (\bar{x}_3 \vee \bar{x}_4)$
- By UL: $\psi[\{x_1, x_2, x_3, x_4\}] = \perp$, clause $(\bar{x}_3 \vee \bar{x}_4)$ conflicting.

Conflict graph G:

$$\begin{tikzcd} x_1 \arrow[r] & x_2 \arrow[r] \arrow[dr] & x_3 \arrow[r] & \emptyset \\ & & x_4 \arrow[ur] & \end{tikzcd}$$

Antecedent clauses:

$$\begin{array}{rl} x_2: & (\bar{x}_1 \vee x_2) \\ x_3: & (x_3 \vee y_5 \vee \bar{x}_2) \\ x_4: & (x_4 \vee \bar{y}_5 \vee \bar{x}_2) \\ \emptyset: & (\bar{x}_3 \vee \bar{x}_4) \end{array}$$

QCDCL (3)

Example (Clause Learning, continued)

Prefix: $\exists x_1, x_3, x_4 \forall y_5 \exists x_2$
Assignment $A = \{x_1, x_2, x_3, x_4\}$
Conflict graph G:

$$\begin{tikzcd}
x_1 \arrow[r] & x_2 \arrow[r] \arrow[dr] & x_3 \arrow[r] & \emptyset \\
 & & x_4 \arrow[ur] &
\end{tikzcd}$$

Antecedent clauses:

$$\begin{array}{ll}
x_2: & (\bar{x}_1 \vee x_2) \\
x_3: & (x_3 \vee y_5 \vee \bar{x}_2) \\
x_4: & (x_4 \vee \bar{y}_5 \vee \bar{x}_2) \\
\emptyset: & (\bar{x}_3 \vee \bar{x}_4)
\end{array}$$

- Start at $\emptyset$, select pivots in reverse assignment ordering.
- Resolve antecedents of x_4, x_3, x_2.
- Pivots obey order restriction of LDQ-resolution.
- Derivation of learned clause is regular, size linear in $|G|$.

$$\begin{tikzcd}[column sep=tiny]
(\bar{x}_3 \vee \bar{x}_4) \arrow[dr, dash] & & (x_4 \vee \bar{y}_5 \vee \bar{x}_2) \arrow[dl, dash] & \\
 & (\bar{x}_3 \vee \bar{y}_5 \vee \bar{x}_2) \arrow[dr, dash] & & (x_3 \vee y_5 \vee \bar{x}_2) \arrow[dl, dash] \\
(\bar{x}_1 \vee x_2) \arrow[dr, dash] & & (\bar{y}_5 \vee y_5 \vee \bar{x}_2) \arrow[dl, dash] & \\
 & (\bar{x}_1) & &
\end{tikzcd}$$

QCDCL (4): Satisfiable QBFs

Definition (Model Generation, cf. [GNT06, Let02, ZM02b])

$$\frac{}{C}$$

$C = (\bigwedge_{l \in A})$ is a cube where $\{x, \bar{x}\} \not\subseteq C$ and A is an assignment with $\psi[A] = \top$, i.e. every clause of PCNF ψ satisfied under A.

- Cube learning: *conjunctions*, *existential reduction*, *universal pivots*.
- PCNF ψ is satisfiable iff the empty cube can be derived from ψ.

Example

$\psi = \exists x \forall u \exists y.(\bar{x} \vee u \vee \bar{y}) \wedge (\bar{x} \vee \bar{u} \vee y) \wedge (x \vee u \vee y) \wedge (x \vee \bar{u} \vee \bar{y})$

$$\begin{array}{ccc}
(\bar{x} \wedge u \wedge \bar{y}) & & (\bar{x} \wedge \bar{u} \wedge y) \\
| & & | \\
(\bar{x} \wedge u) & & (\bar{x} \wedge \bar{u}) \\
& \diagdown \quad \diagup & \\
& (\bar{x}) & \\
& | & \\
& \emptyset &
\end{array}$$

- By model generation: derive cubes $(\bar{x} \wedge u \wedge \bar{y})$ and $(\bar{x} \wedge \bar{u} \wedge y)$.
- By existential reduction: reduce trailing $\bar{y}$ from $(\bar{x} \wedge u \wedge \bar{y})$, y from $(\bar{x} \wedge \bar{u} \wedge y)$.
- Resolve $(\bar{x} \wedge \bar{u})$ and $(\bar{x} \wedge u)$ on universal u.
- Reduce $(\bar{x})$ to derive $\emptyset$.

QCDCL (5): QRES with Generalized Axioms

Definition (Generalized Clause Axiom [LES16])

$$\frac{}{C}$$ Given a PCNF $\psi = \Pi.\phi$ and assignment A generated in QCDCL, $\psi[A]$ is unsatisfiable, and $C = (\bigvee_{l \in A} \bar{l})$ is a clause.

Definition (Generalized Cube Axiom [LES16])

$$\frac{}{C}$$ Given a PCNF $\psi = \Pi.\phi$ and assignment A generated in QCDCL, $\psi[A]$ is satisfiable, and $C = (\bigwedge_{l \in A} l)$ is a cube.

- Close branches in search tree earlier, derive clause/cube, backtrack.
- Generalizes trivial truth/falsity tests in QDPLL.
- Clauses and cubes derived by axioms used in learning as usual.
- Practice: interface to combining QRES with other proof systems.

Part 3:
Future Directions and Open Problems

Experiments (1)

Solver	*Solved*	*UNSAT*	*SAT*	*Time (s)*
DepQBF (SAT 2016)	457	255	202	689K
Quantor	439	228	211	710K
DepQBF 5.0 (LPAR 2015)	434	247	187	727K
DepQBF 4.01	380	219	161	822K
Nenofex	362	193	169	853K
RAReQS	341	211	130	891K
DepQBF 4.01 w/o learning	222	121	102	1101K

- 825 QBFEVAL'16 prenex CNF instances, no preprocessing.
- Limits: 1800 seconds, 7 GB memory.
- Expansion: Nenofex (NNF), Quantor (PCNF), RAReQS (CEGAR).
- QCDCL: public DepQBF X.YZ, SAT 2016 version not yet released.
- Diversity: RAReQS solves 42 instances not solved by DepQBF (SAT 2016), and vice versa 158 instances.

Experiments (2)

Solver	*Solved*	*UNSAT*	*SAT*	*Time (s)*
RAReQS	631	329	302	385K
DepQBF (SAT 2016)	590	299	291	440K
DepQBF 4.01	589	294	295	449K
DepQBF 5.0 (LPAR 2015)	587	300	287	448K
Quantor	494	253	241	608K
Nenofex	487	244	243	623K
DepQBF 4.01 w/o learning	436	222	214	710K

- 825 QBFEVAL'16 prenex CNF instances, with preprocessing.
- Preprocessing by Bloqqer: 344 instances solved (41%), 481 remaining.
- Diversity: RAReQS solves 71 instances not solved by DepQBF (SAT 2016), and vice versa 30 instances.

$\Rightarrow$ expansion and QCDCL have orthogonal strengths.

Experiments (3)

481 Instances not Solved by Preprocessing			
	No Prep.	*With Prep.*	*Diff.*
$\exists$ min	38	10	-73%
$\exists$ max	726K	572K	-21%
$\exists$ avg	16K	7K	-56%
$\exists$ med	4K	1K	-75%
$\forall$ min	1	0	-100%
$\forall$ max	30K	30K	-0%
$\forall$ avg	846	808	-4%
$\forall$ med	66	53	-19%
Qblocks min	2	1	-50%
Qblocks max	1K	179	-82%
Qblocks avg	15.7	6.8	-56%
Qblocks med	3	3	-0%

- Min., max., average and median quantifier blocks and $\forall/\exists$-variables.
- Preprocessing makes instances "more propositional" (67 instances become propositional).

Experiments (4)

Compare RAReQS and DepQBF (SAT 2016).

- Consider the 481 original (*not preprocessed*) instances:
  - RAReQS solved 177: avg qblocks 13.67.
  - DepQBF solved 206: avg qblocks 18.01.
  - RAReQS failed on 304: avg qblocks 16.88.
  - DepQBF failed on 275: avg qblocks 13.97.

- Consider the 481 *preprocessed* instances:
  - RAReQS solved 287: avg qblocks 5.96.
  - DepQBF solved 246: avg qblocks 7.36.
  - RAReQS failed on 194: avg qblocks 8.15.
  - DepQBF failed on 235: avg qblocks 6.30.

$\Rightarrow$ expansion (QCDCL) tends to solve instances with few (many) qblocks.
$\Rightarrow$ expansion (QCDCL) tends to fail on instances with many (few) qblocks.

Experiments (5)

- Consider the 481 original (*not preprocessed*) instances:
  - 311 instances with ≤ 3 qblocks:
    - ⋆ RAReQS solves 121 (38%).
    - ⋆ DepQBF solves 112 (36%).
  - 170 instances with ≥ 4 qblocks:
    - ⋆ RAReQS solves 56 (32%).
    - ⋆ DepQBF solves 94 (55%).
- Consider the 481 *preprocessed* instances:
  - 335 instances with ≤ 3 qblocks:
    - ⋆ RAReQS solves 211 (62%).
    - ⋆ DepQBF solves 155 (46%).
  - 146 instances with ≥ 4 qblocks:
    - ⋆ RAReQS solves 76 (52%).
    - ⋆ DepQBF solves 91 (62%).

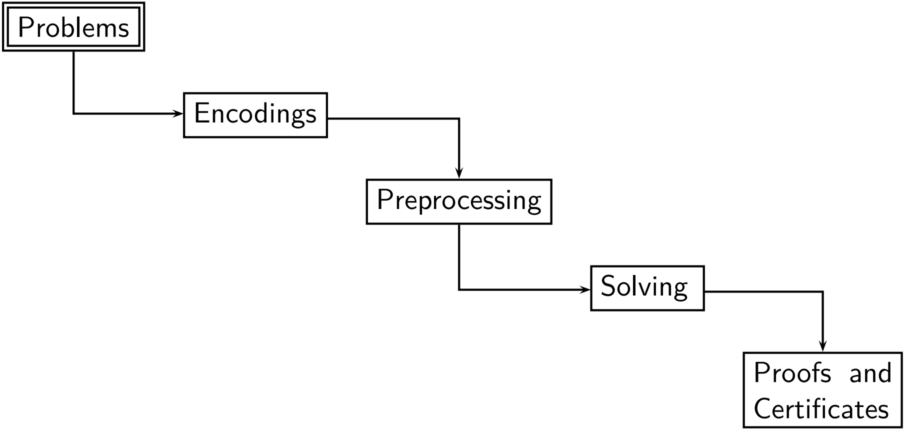
$\Rightarrow$ expansion outperforms QCDCL on instances with few qblocks.

$\Rightarrow$ QCDCL outperforms expansion on instances with many qblocks.

Open Problems: Proof Systems in Theory and Practice

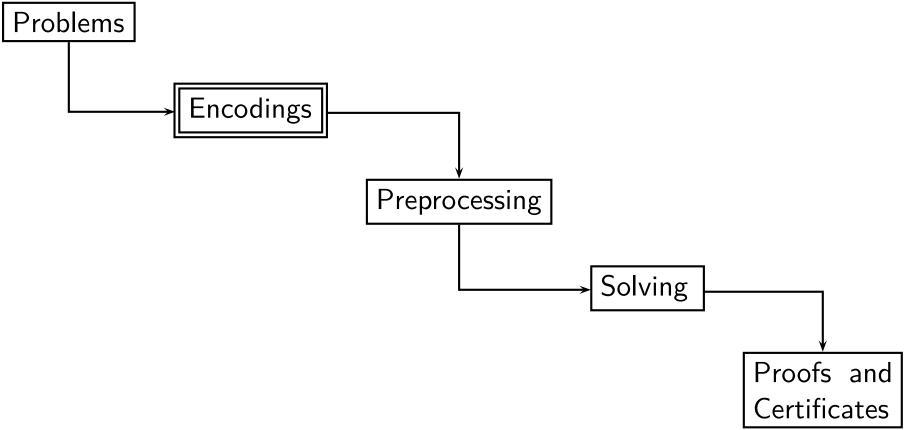
- How to apply proof systems stronger than expansion or QRES in solvers (e.g. variants of instantiation)?
- How to effectively combine expansion and QRES in a single solver to fully benefit from their individual strengths?
- What about proof systems for *satisfiable* QBFs and related theory? E.g. cube learning.
- How to better understand the empirical hardness of instances? What is the role of alternations? Cf. [Rin07].
- How to harness the full power of Q-resolution in QCDCL [Jan16]?

The Need for an Integrated QBF Workflow



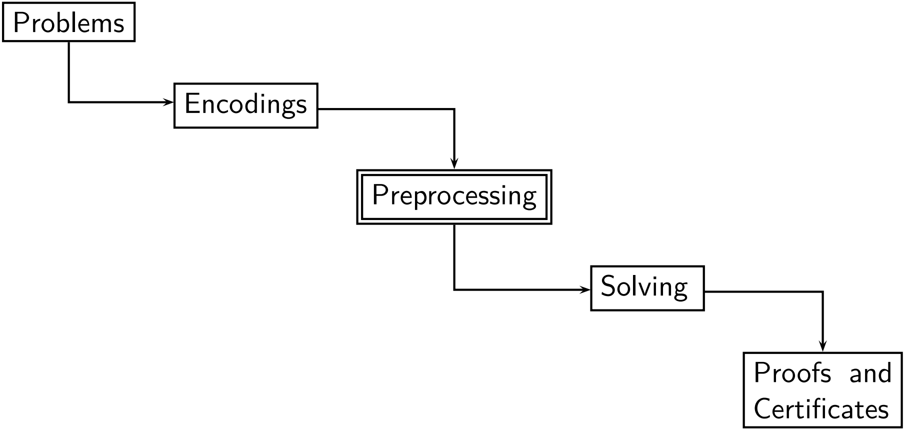
Choose an instance P of a problem to be solved.

The Need for an Integrated QBF Workflow



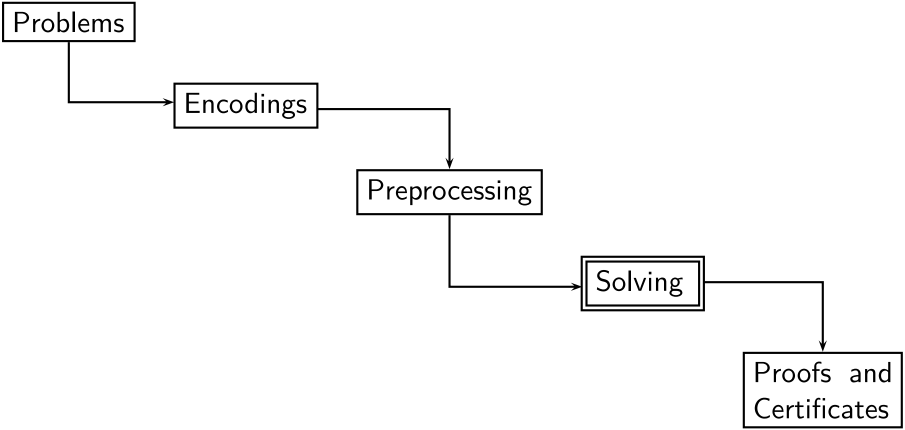
Encode P as (an incremental sequence of) QBFs.

The Need for an Integrated QBF Workflow



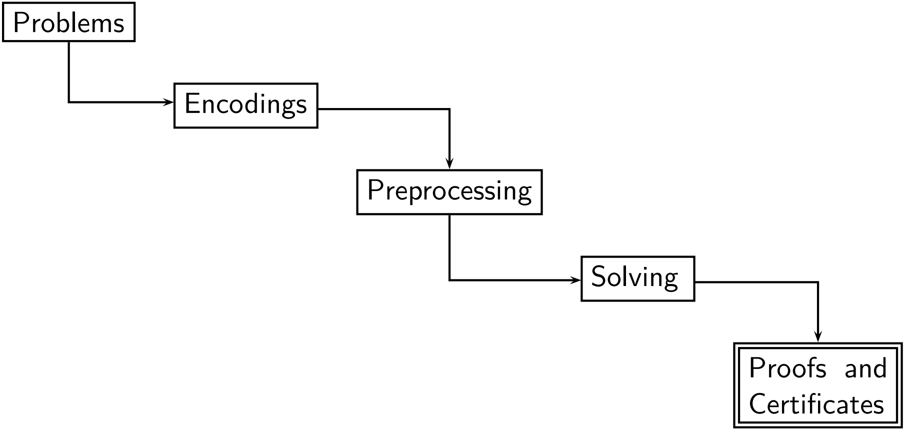
Simplify the QBF encoding (optional).

The Need for an Integrated QBF Workflow



Solve the QBF encoding (incrementally).

The Need for an Integrated QBF Workflow



Obtain a solution to P from a (counter-)model of the QBF.

Open Problems: Application Workflows

- How to equip QBF *workflows* with proof generation and/or extraction of Skolem/Herbrand functions?
- How to make the entire workflow incremental?
- How to parallelize the entire workflow?

Conclusion

- QBF is still an emerging field with plenty of applications.
- Assuming that NP $\neq$ PSPACE, QBF is more difficult than SAT...
- ...but allows for exponentially more succinct encodings than SAT.
- Computational hardness motivates exploring alternative approaches: e.g. CEGAR-based expansion, computing Skolem functions [RS16].
- QBF tools are not (yet) a push-button technology.
- Expert and/or domain knowledge may be necessary for tuning.
- Please document and publish your tools and benchmarks!

Appendix

[Appendix] Expansion and Instantiation

Definition ($\forall$Exp+RES [JM13, BCJ14, JM15a])

- Axiom: $\dfrac{}{C}$ for all $x \in \hat{Q}$: $\{x, \bar{x}\} \not\subseteq C$ and $C \in \phi$
- Instantiation: $\dfrac{C}{\{l^{A_l} \mid l \in C, q(l) = \exists\}}$

  Complete assignment A to universal variables s.t. literals in C falsified, $A_l \subseteq A$ restricted to universal variables u with $u < l$.
- Resolution: $\dfrac{C_1 \cup \{p^A\} \qquad C_2 \cup \{\bar{p}^A\}}{C_1 \cup C_2}$ for all $x \in \hat{Q}$: $\{x, \bar{x}\} \not\subseteq (C_1 \cup C_2)$

- First, instantiate (i.e. replace) all universal variables by constants.
- Existential literals in a clause are annotated by partial assignments.
- Finally, resolve on existential literals with matching annotations.
- Instantiation and annotation mimics universal expansion.

[Appendix] Expansion and Instantiation

Example (continued)

$\psi = \exists x \forall u \exists y.\ (\bar{x} \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{u} \vee y) \wedge (u \vee \bar{y})$

- Complete assignments: $A = \{\bar{u}\}$ and $A' = \{u\}$.
- Instantiate: $(\bar{x} \vee y^{\bar{u}}) \wedge (x \vee \bar{y}^{u}) \wedge (y^{u}) \wedge (\bar{y}^{\bar{u}})$
- Note: cannot resolve (y^{u}) and $(\bar{y}^{\bar{u}})$ due to mismatching annotations.
- Obtain (x) from $(x \vee \bar{y}^{u})$ and (y^{u}), $(\bar{x})$ from $(\bar{x} \vee y^{\bar{u}})$ and $(\bar{y}^{\bar{u}})$.

Different Power of QBF Proof Systems:

- Q-resolution and expansion/instantiation are incomparable [BCJ15].
- Interpreting QBFs as first-order logic formulas [SLB12, Egl16].

References

References I

Please note: since the duration of this talk is limited, the list of references below is incomplete and does not reflect the history and state of the art in QBF research in full accuracy.

[AB02] Abdelwaheb Ayari and David A. Basin.
QUBOS: Deciding Quantified Boolean Logic Using Propositional Satisfiability Solvers.
In *FMCAD*, volume 2517 of *LNCS*, pages 187–201. Springer, 2002.

[BBC16] Olaf Beyersdorff, Ilario Bonacina, and Leroy Chew.
Lower Bounds: From Circuits to QBF Proof Systems.
In *ITCS*, pages 249–260. ACM, 2016.

[BCJ14] Olaf Beyersdorff, Leroy Chew, and Mikolas Janota.
On unification of QBF resolution-based calculi.
In *MFCS*, volume 8635 of *LNCS*, pages 81–93. Springer, 2014.

[BCJ15] Olaf Beyersdorff, Leroy Chew, and Mikolás Janota.
Proof Complexity of Resolution-based QBF Calculi.
In *STACS*, volume 30 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 76–89. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.

References II

[Bie04] Armin Biere.
Resolve and Expand.
In *SAT*, volume 3542 of *LNCS*, pages 59–70. Springer, 2004.

[BJ12] Valeriy Balabanov and Jie-Hong R. Jiang.
Unified QBF certification and its applications.
Formal Methods in System Design, 41(1):45–65, 2012.

[BJS+16] Valeriy Balabanov, Jie-Hong Roland Jiang, Christoph Scholl, Alan Mishchenko, and Robert K. Brayton.
2QBF: Challenges and Solutions.
In *SAT*, volume 9710 of *LNCS*, pages 453–469. Springer, 2016.

[BK07] Uwe Bubeck and Hans Kleine Büning.
Bounded Universal Expansion for Preprocessing QBF.
In *SAT*, volume 4501 of *LNCS*, pages 244–257. Springer, 2007.

[BKF95] Hans Kleine Büning, Marek Karpinski, and Andreas Flögel.
Resolution for Quantified Boolean Formulas.
Inf. Comput., 117(1):12–18, 1995.

[BM08] Marco Benedetti and Hratch Mangassarian.
QBF-Based Formal Verification: Experience and Perspectives.
JSAT, 5(1-4):133–191, 2008.

References III

[BWJ14] Valeriy Balabanov, Magdalena Widl, and Jie-Hong R. Jiang.
QBF Resolution Systems and Their Proof Complexities.
In *SAT*, volume 8561 of *LNCS*, pages 154–169. Springer, 2014.

[CGJ+03] Edmund M. Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith.
Counterexample-guided abstraction refinement for symbolic model checking.
J. ACM, 50(5):752–794, 2003.

[CGS98] Marco Cadoli, Andrea Giovanardi, and Marco Schaerf.
An Algorithm to Evaluate Quantified Boolean Formulae.
In *AAAI*, pages 262–267. AAAI Press / The MIT Press, 1998.

[CHR16] Chih-Hong Cheng, Yassine Hamza, and Harald Ruess.
Structural Synthesis for GXW Specifications.
In *CAV*, volume 9779 of *LNCS*, pages 95–117. Springer, 2016.

[CSGG02] Marco Cadoli, Marco Schaerf, Andrea Giovanardi, and Massimo Giovanardi.
An Algorithm to Evaluate Quantified Boolean Formulae and Its Experimental Evaluation.
JAIR, 28(2):101–142, 2002.

[DLL62] Martin Davis, George Logemann, and Donald W. Loveland.
A Machine Program for Theorem-Proving.
Commun. ACM, 5(7):394–397, 1962.

References IV

[Egl16] Uwe Egly.
On Stronger Calculi for QBFs.
In *SAT*, volume 9710 of *LNCS*, pages 419–434. Springer, 2016.

[FT14] Bernd Finkbeiner and Leander Tentrup.
Fast DQBF Refutation.
In *SAT*, volume 8561 of *LNCS*, pages 243–251. Springer, 2014.

[FT15] Bernd Finkbeiner and Leander Tentrup.
Detecting Unrealizability of Distributed Fault-tolerant Systems.
Logical Methods in Computer Science, 11(3), 2015.

[GMN10] Enrico Giunchiglia, Paolo Marin, and Massimo Narizzano.
QuBE7.0.
JSAT, 7(2-3):83–88, 2010.

[GNT02] Enrico Giunchiglia, Massimo Narizzano, and Armando Tacchella.
Learning for Quantified Boolean Logic Satisfiability.
In *AAAI*, pages 649–654. AAAI Press / The MIT Press, 2002.

[GNT06] Enrico Giunchiglia, Massimo Narizzano, and Armando Tacchella.
Clause/Term Resolution and Learning in the Evaluation of Quantified Boolean Formulas.
JAIR, 26:371–416, 2006.

References V

[GT14] Adria Gascón and Ashish Tiwari.
A Synthesized Algorithm for Interactive Consistency.
In *NASA Formal Methods*, volume 8430 of *LNCS*, pages 270–284. Springer, 2014.

[HSM+14] Tamir Heyman, Dan Smith, Yogesh Mahajan, Lance Leong, and Husam Abu-Haimed.
Dominant Controllability Check Using QBF-Solver and Netlist Optimizer.
In *SAT*, volume 8561 of *LNCS*, pages 227–242. Springer, 2014.

[Jan16] Mikolás Janota.
On Q-Resolution and CDCL QBF Solving.
In *SAT*, volume 9710 of *LNCS*, pages 402–418. Springer, 2016.

[JKMSC16] Mikoláš Janota, William Klieber, Joao Marques-Silva, and Edmund Clarke.
Solving QBF with counterexample guided refinement.
Artificial Intelligence, 234:1–25, 2016.

[JM13] Mikolás Janota and João Marques-Silva.
On Propositional QBF Expansions and Q-Resolution.
In *SAT*, volume 7962 of *LNCS*, pages 67–82. Springer, 2013.

[JM15a] Mikolás Janota and Joao Marques-Silva.
Expansion-based QBF solving versus Q-resolution.
Theor. Comput. Sci., 577:25–42, 2015.

References VI

[JM15b] Mikolás Janota and Joao Marques-Silva.
Solving QBF by Clause Selection.
In *IJCAI*, pages 325–331. AAAI Press, 2015.

[KSGC10] William Klieber, Samir Sapra, Sicun Gao, and Edmund M. Clarke.
A Non-prenex, Non-clausal QBF Solver with Game-State Learning.
In *SAT*, volume 6175 of *LNCS*, pages 128–142. Springer, 2010.

[LB08] Florian Lonsing and Armin Biere.
Nenofex: Expanding NNF for QBF Solving.
In *SAT*, volume 4996 of *LNCS*, pages 196–210. Springer, 2008.

[LB10] Florian Lonsing and Armin Biere.
DepQBF: A Dependency-Aware QBF Solver.
JSAT, 7(2-3):71–76, 2010.

[LES16] Florian Lonsing, Uwe Egly, and Martina Seidl.
Q-Resolution with Generalized Axioms.
In *SAT*, volume 9710 of *LNCS*, pages 435–452. Springer, 2016.

[Let02] Reinhold Letz.
Lemma and Model Caching in Decision Procedures for Quantified Boolean Formulas.
In *TABLEAUX*, volume 2381 of *LNCS*, pages 160–175. Springer, 2002.

References VII

[MVB10] Hratch Mangassarian, Andreas G. Veneris, and Marco Benedetti.
Robust QBF Encodings for Sequential Circuits with Applications to Verification, Debug, and Test.
IEEE Trans. Computers, 59(7):981–994, 2010.

[PS09] Florian Pigorsch and Christoph Scholl.
Exploiting structure in an AIG based QBF solver.
In *DATE*, pages 1596–1601. IEEE, 2009.

[Rin07] Jussi Rintanen.
Asymptotically Optimal Encodings of Conformant Planning in QBF.
In *AAAI*, pages 1045–1050. AAAI Press, 2007.

[RS16] Markus N. Rabe and Sanjit A. Seshia.
Incremental Determinization.
In *SAT*, volume 9710 of *LNCS*, pages 375–392. Springer, 2016.

[RT15] Markus N. Rabe and Leander Tentrup.
CAQE: A Certifying QBF Solver.
In *FMCAD*, pages 136–143. IEEE, 2015.

[RTM04] Darsh P. Ranjan, Daijue Tang, and Sharad Malik.
A Comparative Study of 2QBF Algorithms.
In *SAT*, 2004.

References VIII

[Sha49] Claude Elwood Shannon.
The Synthesis of Two-Terminal Switching Circuits.
Bell System Technical Journal, 28(1):59–98, 1949.

[SLB12] Martina Seidl, Florian Lonsing, and Armin Biere.
qbf2epr: A Tool for Generating EPR Formulas from QBF.
In *PAAR Workshop*, volume 21 of *EPiC Series*, pages 139–148. EasyChair, 2012.

[VG12] Allen Van Gelder.
Contributions to the Theory of Practical Quantified Boolean Formula Solving.
In *CP*, volume 7514 of *LNCS*, pages 647–663. Springer, 2012.

[ZM02a] Lintao Zhang and Sharad Malik.
Conflict Driven Learning in a Quantified Boolean Satisfiability Solver.
In *ICCAD*, pages 442–449. ACM / IEEE Computer Society, 2002.

[ZM02b] Lintao Zhang and Sharad Malik.
Towards a Symmetric Treatment of Satisfaction and Conflicts in Quantified Boolean Formula Evaluation.
In *CP*, volume 2470 of *LNCS*, pages 200–215. Springer, 2002.