SAT Solving and its Extensions

Quantified Boolean Formulas

Uwe Egly, Katalin Fazekas

Slides are kindly provided by Florian Lonsing

Institute of Logic and Computation
TU Wien, Austria

Introduction (1)

Propositional Logic:

- Formula ϕ over propositional variables, Boolean domain $\mathcal{B} = \{\top, \bot\}$.
- Satisfiability problem (SAT): is ϕ satisfiable?
- NP-completeness of SAT.
- Modelling NP-complete problems in formal verification, AI, ...
- A SAT solver returns a model of ϕ or a proof that ϕ has no model.

Example

Propositional formulas in conjunctive normal form (CNF):

- $\Phi := (x \vee \bar{y}) \wedge (\bar{x} \vee y)$
- Φ is satisfiable: models $M := \{x, y\}$ or $M' := \{\bar{x}, \bar{y}\}$
- $\Phi' := (x) \wedge (\bar{x})$ is unsatisfiable (i.e., has no model).

Introduction (2)

Quantified Boolean Formulas (QBF):

- Propositional logic extended by existential ($\exists$) / universal ($\forall$) quantification of propositional variables.
- Checking QBF satisfiability: PSPACE-complete.
- Propositional satisfiability (SAT): NP-complete.
- QBF encodings: potentially more succinct than propositional logic.
- Applications to presumably harder problems, e.g. NEXPTIME

Example

- QBF $\psi := \hat{Q}.\phi$ in *prenex conjunctive normal form (PCNF)*.
- $\psi = \underbrace{\forall x \exists y.}_{\text{quantifier prefix } \hat{Q}} \underbrace{(x \vee \bar{y}) \wedge (\bar{x} \vee y)}_{\text{propositional CNF } \phi}.$

Introduction (3): Progress in QBF Research

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 (3): Progress in QBF Research

Maturity of QBF Technology:

- QBF *not* yet widely applied at large scale.
- Higher complexity (PSPACE) comes at a cost.

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].

QBF Research Community:

- QBFLIB: `http://www.qbflib.org/index.php`
- QBFEVAL'17: `http://www.qbflib.org/qbfeval17.php`

Introduction (4): Motivating QBF Applications

- **Synthesis and Realizability of Distributed Systems**
  - [GT14] Adria Gascón, Ashish Tiwari: A Synthesized Algorithm for Interactive Consistency. NASA Formal Methods 2014: 270-284.
  - [FT15] Bernd Finkbeiner, Leander Tentrup: Detecting Unrealizability of Distributed Fault-tolerant Systems. Logical Methods in Computer Science 11(3) (2015).
  - [FFRT17] Peter Faymonville, Bernd Finkbeiner, Markus N. Rabe, Leander Tentrup: Encodings of Bounded Synthesis. TACAS (1) 2017: 354-370.
- **Solving Dependency Quantified Boolean Formulas** (NEXPTIME)
  - [FT14] Bernd Finkbeiner, Leander Tentrup: Fast DQBF Refutation. SAT 2014: 243-251.
  - [WKB$^+$18] Ralf Wimmer, Andreas Karrenbauer, Ruben Becker, Christoph Scholl, Bernd Becker: From DQBF to QBF by Dependency Elimination. SAT 2017: 326-343.

Introduction (5): Motivating QBF Applications

- **Formal Verification and Synthesis**
  - [HSM+14] Tamir Heyman, Dan Smith, Yogesh Mahajan, Lance Leong, Husam Abu-Haimed: Dominant Controllability Check Using QBF-Solver and Netlist Optimizer. SAT 2014: 227-242.
  - [CHR16] Chih-Hong Cheng, Yassine Hamza, Harald Ruess: Structural Synthesis for GXW Specifications. CAV 2016.
- **Automated Planning**
  - [CFG13] Michael Cashmore, Maria Fox, Enrico Giunchiglia: Partially Grounded Planning as Quantified Boolean Formula. ICAPS 2013
  - [EKLP17] Uwe Egly, Martin Kronegger, Florian Lonsing, Andreas Pfandler: Conformant planning as a case study of incremental QBF solving. Ann. Math. Artif. Intell. 80(1): 21-45 (2017)
  - [GLM+18] Olivier Gasquet, Dominique Longin, Frederic Maris, Pierre Régnier, Maël Valais: Compact Tree Encodings for Planning as QBF. Inteligencia Artif. 21(62): 103-114 (2018)
  - [SvdP22] Irfansha Shaik, Jaco van de Pol: Classical Planning as QBF Without Grounding. (to appear at ICAPS 2022)

Outline

Preliminaries

- QBF syntax and semantics.

Proof Systems

- Results in QBF proof complexity.
- Understanding and analyzing techniques implemented in QBF solvers.

A Typical QBF Workflow

- How to encode problems as a QBF?
- How to solve a QBF?
- How to obtain a solution to a problem from a solved QBF?

Outlook and Future Work

- Open problems and possible research directions.

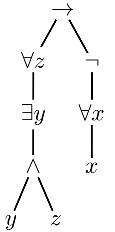
Syntax (1)

QBFs as Quantified Circuits:

- $\top$ and $\bot$ are QBFs.
- For propositional variables *Vars*, (x) where $x \in \mathit{Vars}$ is a QBF.
- If ψ is a QBF then $\neg(\psi)$ is a QBF.
- If ψ_1 and ψ_2 are QBFs then $(\psi_1 \circ \psi_2)$ is a QBF, $\circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$.
- If ψ is a QBF and $x \in \mathit{Vars}(\psi)$, then $\forall x.(\psi)$ and $\exists x.(\psi)$ are QBFs.

Example

$\psi := (\forall z.(\exists y.(y \wedge z))) \rightarrow \neg(\forall x.(x))$



Syntax (1)

QBFs in Prenex CNF: $\psi := \hat{Q}.\phi$

- Quantifier prefix $\hat{Q} = Q_1 B_1 \dots Q_n B_n$, $Q_i \in \{\forall, \exists\}$, $Q_i \neq Q_j$, $B_i \subseteq \mathit{Vars}$, $(B_i \cap B_j) = \emptyset$.
- Linear ordering of variables: $x_i < x_j$ iff $x_i \in B_i$, $x_j \in B_j$, and $i < j$.
- Quantifier-free CNF ϕ over propositional variables x_i.
- Assume: ϕ does not contain free variables, all x_i in $\hat{Q}$ appear in ϕ.

Example

- PCNF $\psi = \forall u \exists x.(\bar{u} \vee x) \wedge (u \vee \bar{x})$.
- Linear ordering: $u < x$.

Syntax (2)

Example (QDIMACS Format)

$\exists x_1, x_3, x_4 \forall y_5 \exists x_2.$
$(\bar{x}_1 \lor x_2) \land (x_3 \lor y_5 \lor \bar{x}_2) \land (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \land (\bar{x}_3 \lor \bar{x}_4)$

- Extension of DIMACS format used in SAT solving.
- Literals of variables encoded as signed integers.
- One quantifier block per line, terminated by zero.
- "a" labels $\forall$, "e" labels $\exists$.
- One clause per line, terminated by zero.

```
p cnf 5 4
e 1 3 4 0
a 5 0
e 2 0
-1 2 0
3 5 -2 0
4 -5 -2 0
-3 -4 0
```

QDIMACS format: `http://www.qbflib.org/qdimacs.html`

Semantics (1)

Recursive Definition:

- Assume that a QBF does not contain free variables.
- The QBF $\bot$ 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 $\bot$ ($\top$).
- The QBF $\exists x.(\psi)$ is satisfiable iff $\psi[\neg x]$ or $\psi[x]$ is satisfiable.

Definition

The QBFs ψ and ψ' are *satisfiability-equivalent* ($\psi \equiv_{sat} \psi'$) iff ψ is satisfiable whenever ψ' is satisfiable.

Semantics (1)

Example

Observe: recursive evaluation assigns variables in prefix ordering.

The PCNF $\psi = \forall x \exists y.(x \vee \bar{y}) \wedge (\bar{x} \vee y)$ is satisfiable if

(1) $\psi[x] = \exists y.(y)$ and

(2) $\psi[\bar{x}] = \exists y.(\bar{y})$ are satisfiable.

(1) $\psi[x] = \exists y.(y)$ is satisfiable since $\psi[x, y] = \top$ is satisfiable.
(2) $\psi[\bar{x}] = \exists y.(\bar{y})$ is satisfiable since $\psi[\bar{x}, \bar{y}] = \top$ is satisfiable.

Semantics (1)

Example

Observe: recursive evaluation assigns variables in prefix ordering.

The PCNF $\psi = \exists y \forall x.(x \vee \bar{y}) \wedge (\bar{x} \vee y)$ is unsatisfiable because neither

(1) $\psi[y] = \forall x.(x)$ nor

(2) $\psi[\bar{y}] = \forall x.(\bar{x})$ is satisfiable.

(1) $\psi[y] = \forall x.(x)$ is unsatisfiable since $\psi[y, \bar{x}]$ is unsatisfiable.
(2) $\psi[\bar{y}] = \forall x.(\bar{x})$ is unsatisfiable since $\psi[\bar{y}, x]$ is unsatisfiable.

Semantics (2)

Definition (Skolem/Herbrand Function)

Let ψ be a PCNF, x (y) a universal (existential) variable.

- Let $D^{\psi}(v) := \{w \in \psi \mid q(v) \neq q(w) \text{ and } w < v\}$, $q(v) \in \{\forall, \exists\}$.
- Skolem function $f_y(x_1, \ldots, x_k)$ of y: $D^{\psi}(y) = \{x_1, \ldots, x_k\}$.
- Herbrand function $f_x(y_1, \ldots, y_k)$ of x: $D^{\psi}(x) = \{y_1, \ldots, y_k\}$.

Definition (Skolem Function Model)

A PCNF ψ with existential variables $y_1, \ldots, y_m$ is satisfiable iff $\psi[y_1/f_{y_1}(D^{\psi}(y_1)), \ldots, y_m/f_{y_m}(D^{\psi}(y_m))]$ is satisfiable.

Definition (Herbrand Function Countermodel)

A PCNF ψ with universal variables $x_1, \ldots, x_m$ is unsatisfiable iff $\psi[x_1/f_{x_1}(D^{\psi}(x_1)), \ldots, x_m/f_{x_m}(D^{\psi}(x_m))]$ is unsatisfiable.

Semantics (3)

Example (Skolem Function Model)

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

- Skolem function $f_x = \bot$ of x with $D^{\psi}(x) = \emptyset$.
- Skolem function $f_y(u) = \bar{u}$ of y with $D^{\psi}(y) = \{u\}$.
- $\psi[x/f_x, y/f_y(u)] = \forall u. (\bot \vee u \vee \bar{u}) \wedge (\bot \vee \bar{u} \vee u)$
- Satisfiable: $\psi[x/f_x, y/f_y(u)] = \top$

Example (Herbrand Function Countermodel)

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

- Herbrand function $f_u(x) = (x)$ of u with $D^{\psi}(u) = \{x\}$.
- $\psi[u/f_u(x)] = \exists x, y. (x \vee x \vee y) \wedge (x \vee x \vee \bar{y}) \wedge (\bar{x} \vee \bar{x} \vee y) \wedge (\bar{x} \vee \bar{x} \vee \bar{y})$
- Unsatisfiable: $\psi[u/f_u(x)] = \exists x, y. (x \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge (\bar{x} \vee \bar{y})$

QBF Proof Systems

QBF Proof Systems (1): Q-Resolution

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

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

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

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

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

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

QBF Proof Systems (2): Resolution

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

- Generation of tautologies must respect prefix ordering of pivots.
- 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.

QU-Resolution: [VG12]

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

Further Variants: [BWJ14]

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

QBF Proof Systems (3): Expansion and Instantiation

Example

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

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

Universal Expansion: cf. [AB02, Bie04, JKMSC16]

- Idea: Eliminate all universal variables by Shannon expansion.
- Finally, apply SAT solving.
- 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].
- Based on CNF, NNF, and-inverter graphs [AB02, LB08, PS09].

QBF Proof Systems (4): 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.

QBF Proof Systems (5): 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].

Encoding Problems as QBF

Encoding Problems (1): Bounded Model Checking

Example (Bounded Model Checking (BMC) [BCCZ99])

- System S, states of S as a state graph, invariant P.
- Goal: search for a counterexample to P of bounded length k.
- Counterexample: path to reachable state s_k where P violated.

SAT Encoding:

- Initial state predicate $I(s)$, transition relation $T(s, s')$.
- "Bad state" predicate $B(s)$: s is a state where P is violated.
- Error trace of length k: $I(s_0) \wedge T(s_0, s_1) \wedge \ldots \wedge T(s_{k-1}, s_k) \wedge B(s_k)$.

QBF Encoding: [BM08, JB07]

- $\exists s_0, \ldots, s_k \forall x, x'.$
  $I(s_0) \wedge B(s_k) \wedge \Big[\big[\bigvee_{i=0}^{k-1}((x = s_i) \wedge (x' = s_{i+1}))\big] \rightarrow T(x, x')\Big].$
- Only one copy of T in contrast to k copies in SAT encoding.

Encoding Problems (2)

QCIR: Quantified CIRcuit

- Format for QBFs in non-prenex non-CNF.
- Conversion tools, e.g., part of GhostQ solver [Gho16, KSGC10].

2 Format Specification

2.1 Syntax

The following BNF grammar specifies the structure of a formula represented in QCIR (Quantified CIRcuit).

qcir-file	::=	*format-id qblock-stmt output-stmt (gate-stmt nl)**
format-id	::=	`#QCIR-G14` [*integer*] *nl*
qblock-stmt	::=	[`free`(*var-list*)*nl*] *qblock-quant**
qblock-quant	::=	*quant*(*var-list*)*nl*
var-list	::=	(*var*,)* *var*
lit-list	::=	(*lit*,)* *lit* | ϵ
output-stmt	::=	`output`(*lit*)*nl*
gate-stmt	::=	*gvar* = *ngate_type*(*lit-list*)
	|	*gvar* = `xor`(*lit*, *lit*)
	|	*gvar* = `ite`(*lit*, *lit*, *lit*)
	|	*gvar* = *quant*(*var-list*; *lit*)
quant	::=	`exists` | `forall`
var	::=	(A string of ASCII letters, digits, and underscores)
gvar	::=	(A string of ASCII letters, digits, and underscores)
nl	::=	newline
lit	::=	*var* | -*var* | *gvar* | -*gvar*
ngate_type	::=	`and` | `or`

3.2 Formula in Non-Prenex Form

A formula in non-prenex form looks as follows:

```
#QCIR-G14

forall(z)

output(g3)

g1 = and(x1, x2, z)

g2 = exists(x1, x2; g1)

g3 = or(z, g2)
```

$$\forall z.\ \overbrace{z \vee \underbrace{\exists x_1.\exists x_2.\ \underbrace{(x_1 \wedge x_2 \wedge z)}_{g_1}}_{g_2}}^{g_3}$$

From [QCI14]: `http://qbf.satisfiability.org/gallery/qcir-gallery14.pdf`

Encoding Problems (3)

Definition (Prenexing, cf. [AB02, Egl94, EST+03, ETW02, GNT07])

$(Qx.\ \phi) \circ \psi \equiv Qx.\ (\phi \circ \psi)$, ψ a QBF, $Q \in \{\forall, \exists\}, \circ \in \{\wedge, \vee\}$, $x \notin Var(\psi)$.

Definition (CNF transformation, cf. [Tse68, NW01, PG86])

- Given a prenex QBF $\psi := \hat{Q}.\phi$, subformulas ψ_i of ψ.
- $\psi_i = (\psi_{i,l} \circ \psi_{i,r})$, $\circ \in \{\vee, \wedge, \rightarrow, \leftrightarrow, \otimes\}$.
- Add equivalences $t_i \leftrightarrow (\psi_{i,l} \circ \psi_{i,r})$, fresh variable t_i.
- Convert each $t_i \leftrightarrow (\psi_{i,l} \circ \psi_{i,r})$ to CNF depending on $\circ$.
- Resulting PCNF ψ': satisfiability-equivalent to ψ, size linear in $|\psi|$.
- Safe: quantify each t_i innermost [GMN09]: $\psi := \hat{Q}\exists t_i.\phi$.

Encoding Problems (4)

Definition (QBF Extension Rule, cf. [Tse68, JBS+07, BCJ16])

- Let $\psi := Q_1 x_1 \ldots Q_i x_i \ldots Q_j x_j \ldots Q_n x_n.\phi$ be a PCNF.
- Consider variables x_i, x_j with $x_i \leq x_j$ in ψ, fresh existential variable v.
- Add definition $v \leftrightarrow (\bar{x}_i \vee \bar{x}_j)$ in CNF: $(\bar{v} \vee \bar{x}_i \vee \bar{x}_j) \wedge (v \vee x_i) \wedge (v \vee x_j)$.
- Strong variant: quantify v after x_j, $Q_1 x_1 \ldots Q_i x_i \ldots Q_j x_j \exists v \ldots Q_n x_n$.
- Weak variant: quantify v innermost, $Q_1 x_1 \ldots Q_i x_i \ldots Q_j x_j \ldots Q_n x_n \exists v$.

Proposition (cf. [JBS+07, BCJ16])

Q-resolution with the strong extension rule is exponentially more powerful than with the weak extension rule with respect to lengths of refutations.

$\Rightarrow$ "bad" placement of Tseitin variables in encoding phase may have negative impact on solving in a later stage.

Encoding Problems (5): QParity

Definition (QParity Function [BCJ15])

$QParity_n := \exists x_1, \ldots, x_n \forall y.\ XOR(XOR(\ldots XOR(x_1, x_2), \ldots, x_n), y)$.

CNF ϕ of $QParity_n$ by Tseitin translation:

$$\begin{array}{c} (t_1 \leftrightarrow XOR(x_1, x_2)) \wedge \\ \bigwedge_{1<i<n} (t_i \leftrightarrow XOR(t_{i-1}, x_{i+1})) \wedge \\ (t_n \leftrightarrow XOR(t_{n-1}, y)) \wedge (t_n) \end{array}$$

Prefix by weak extension rule : $\hat{Q}_W := \exists x_1, \ldots, x_n \forall y \exists t_1, \ldots, t_n$

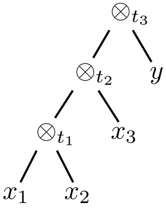
Prefix by strong extension rule: $\hat{Q}_S := \exists x_1, \ldots, x_n \exists t_1, \ldots, t_{n-1} \forall y \exists t_n$

Proposition ([BCJ15, BCJ16])

- *The PCNF $\hat{Q}_W.\phi$ has only exponential Q-resolution refutations.*
- *The PCNF $\hat{Q}_S.\phi$ has polynomial Q-resolution refutations.*

Encoding Problems (6): QParity

$$\hat{Q}_W.\phi := \exists x_1, x_2, x_3 \forall y \qquad .\; XOR_3(XOR_2(XOR_1(x_1, x_2), x_3), y)$$

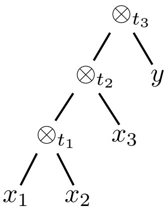


$$t_1 \leftrightarrow XOR(x_1, x_2)$$
$$t_2 \leftrightarrow XOR(t_1, x_3)$$
$$t_3 \leftrightarrow XOR(t_2, y)$$

$$\begin{array}{ll}
t_1: & (\bar{t}_1 \vee x_1 \vee x_2) \wedge \\
& (\bar{t}_1 \vee \bar{x}_1 \vee \bar{x}_2) \wedge \\
& (t_1 \vee \bar{x}_1 \vee x_2) \wedge \\
& (t_1 \vee x_1 \vee \bar{x}_2) \wedge \\
\hline
t_2: & (\bar{t}_2 \vee t_1 \vee x_3) \wedge \\
& (\bar{t}_2 \vee \bar{t}_1 \vee \bar{x}_3) \wedge \\
& (t_2 \vee \bar{t}_1 \vee x_3) \wedge \\
& (t_2 \vee t_1 \vee \bar{x}_3) \wedge \\
\hline
t_3: & (\bar{t}_3 \vee t_2 \vee y) \wedge \\
& (\bar{t}_3 \vee \bar{t}_2 \vee \bar{y}) \wedge \\
& (t_3 \vee \bar{t}_2 \vee y) \wedge \\
& (t_3 \vee t_2 \vee \bar{y}) \wedge \\
\hline
out: & (t_3)
\end{array}$$

Encoding Problems (6): QParity

$\hat{Q}_W.\phi := \exists x_1, x_2, x_3 \forall y \exists t_1, t_2, t_3.\ XOR_3(XOR_2(XOR_1(x_1, x_2), x_3), y)$

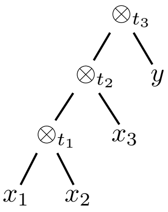


$t_1 \leftrightarrow XOR(x_1, x_2)$
$t_2 \leftrightarrow XOR(t_1, x_3)$
$t_3 \leftrightarrow XOR(t_2, y)$

t_1 :	$(\bar{t}_1 \vee x_1 \vee x_2) \wedge$
	$(\bar{t}_1 \vee \bar{x}_1 \vee \bar{x}_2) \wedge$
	$(t_1 \vee \bar{x}_1 \vee x_2) \wedge$
	$(t_1 \vee x_1 \vee \bar{x}_2) \wedge$
t_2 :	$(\bar{t}_2 \vee t_1 \vee x_3) \wedge$
	$(\bar{t}_2 \vee \bar{t}_1 \vee \bar{x}_3) \wedge$
	$(t_2 \vee \bar{t}_1 \vee x_3) \wedge$
	$(t_2 \vee t_1 \vee \bar{x}_3) \wedge$
t_3 :	$(\bar{t}_3 \vee t_2 \vee y) \wedge$
	$(\bar{t}_3 \vee \bar{t}_2 \vee \bar{y}) \wedge$
	$(t_3 \vee \bar{t}_2 \vee y) \wedge$
	$(t_3 \vee t_2 \vee \bar{y}) \wedge$
out :	(t_3)

Encoding Problems (6): QParity

$$\hat{Q}_S.\phi := \exists x_1, x_2, x_3 \qquad \forall y \qquad . \; XOR_3(XOR_2(XOR_1(x_1, x_2), x_3), y)$$

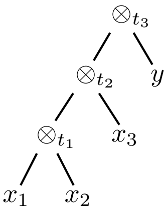


$$t_1 \leftrightarrow XOR(x_1, x_2)$$
$$t_2 \leftrightarrow XOR(t_1, x_3)$$
$$t_3 \leftrightarrow XOR(t_2, y)$$

$$\begin{array}{ll}
t_1: & (\bar{t}_1 \vee x_1 \vee x_2) \wedge \\
 & (\bar{t}_1 \vee \bar{x}_1 \vee \bar{x}_2) \wedge \\
 & (t_1 \vee \bar{x}_1 \vee x_2) \wedge \\
 & (t_1 \vee x_1 \vee \bar{x}_2) \wedge \\
\hline
t_2: & (\bar{t}_2 \vee t_1 \vee x_3) \wedge \\
 & (\bar{t}_2 \vee \bar{t}_1 \vee \bar{x}_3) \wedge \\
 & (t_2 \vee \bar{t}_1 \vee x_3) \wedge \\
 & (t_2 \vee t_1 \vee \bar{x}_3) \wedge \\
\hline
t_3: & (\bar{t}_3 \vee t_2 \vee y) \wedge \\
 & (\bar{t}_3 \vee \bar{t}_2 \vee \bar{y}) \wedge \\
 & (t_3 \vee \bar{t}_2 \vee y) \wedge \\
 & (t_3 \vee t_2 \vee \bar{y}) \wedge \\
\hline
out: & (t_3)
\end{array}$$

Encoding Problems (6): QParity

$\hat{Q}_S.\phi := \exists x_1, x_2, x_3, t_1, t_2 \forall y \exists t_3.\ XOR_3(XOR_2(XOR_1(x_1, x_2), x_3), y)$

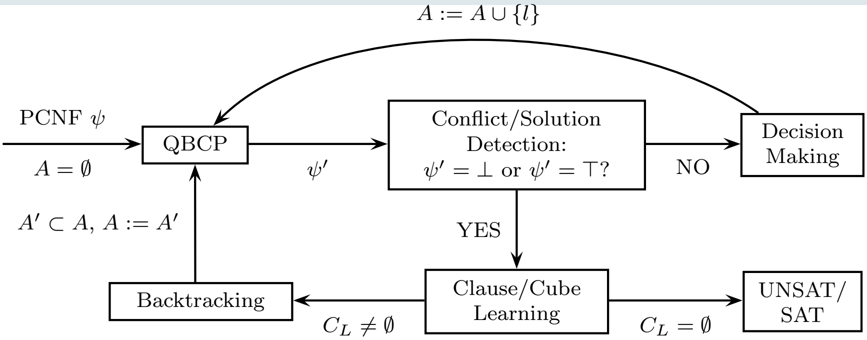


$$t_1 \leftrightarrow XOR(x_1, x_2)$$
$$t_2 \leftrightarrow XOR(t_1, x_3)$$
$$t_3 \leftrightarrow XOR(t_2, y)$$

t_1 :	$(\bar{t}_1 \vee x_1 \vee x_2) \wedge$
	$(\bar{t}_1 \vee \bar{x}_1 \vee \bar{x}_2) \wedge$
	$(t_1 \vee \bar{x}_1 \vee x_2) \wedge$
	$(t_1 \vee x_1 \vee \bar{x}_2) \wedge$
t_2 :	$(\bar{t}_2 \vee t_1 \vee x_3) \wedge$
	$(\bar{t}_2 \vee \bar{t}_1 \vee \bar{x}_3) \wedge$
	$(t_2 \vee \bar{t}_1 \vee x_3) \wedge$
	$(t_2 \vee t_1 \vee \bar{x}_3) \wedge$
t_3 :	$(\bar{t}_3 \vee t_2 \vee y) \wedge$
	$(\bar{t}_3 \vee \bar{t}_2 \vee \bar{y}) \wedge$
	$(t_3 \vee \bar{t}_2 \vee y) \wedge$
	$(t_3 \vee t_2 \vee \bar{y}) \wedge$
out :	(t_3)

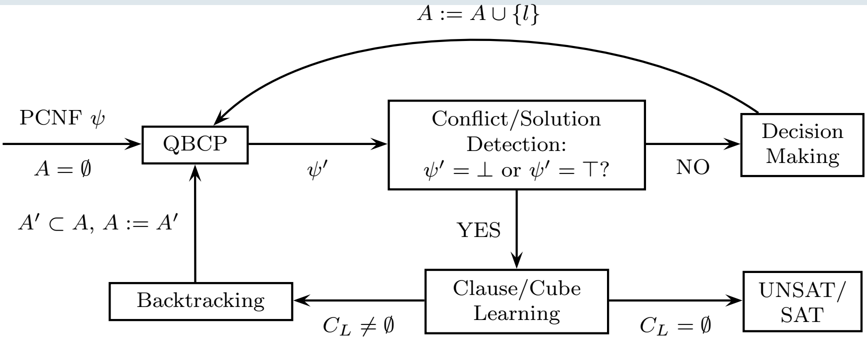
Solving Quantified Boolean Formulas

Solving QBF (1): QCDCL



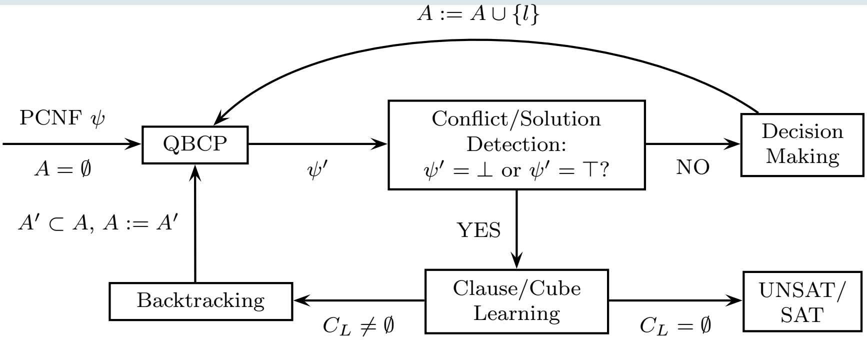
- Generate assignments A by decision making and (unit) propagation.
- Simplify ψ under A to obtain ψ'.
- Conflict: $\psi' = \bot$: ψ' contains a falsified clause.
- Solution: $\psi' = \top$: all clauses in ψ' satisfied (i.e., empty CNF).

Solving QBF (1): QCDCL



- Generate learned clause (cube) C_L by Q-resolution, added to ψ.
- Empty clause (cube) $C_L = \emptyset$: formula proved UNSAT (SAT).
- (LD)Q-resolution proofs of (un)satisfiability by QRES.

Solving QBF (1): QCDCL

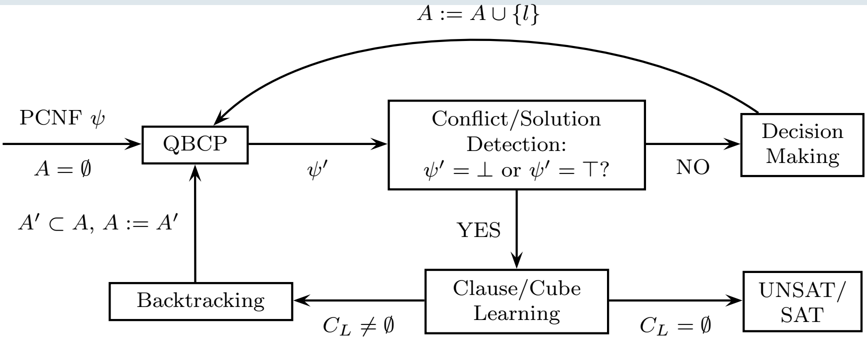


- Conflict detected: select clauses for Q-resolution.

Definition (Clause Axiom of QRES)

$$\frac{}{C}$$ Given a PCNF $\psi = \hat{Q}.\phi$, $C \in \phi$ is a clause.

Solving QBF (1): QCDCL



- Solution detected: select cubes for Q-resolution.

Definition (Cube Axiom of QRES)

$\overline{}$
C

Given a PCNF $\psi = \hat{Q}.\phi$ and an assignment A with $\psi[A] = \top$, $C = (\bigwedge_{l \in A})$ is a cube.

Pseudo Code of QCDCL

```
Result qcdcl (PCNF ψ)
  Result R = UNDEF;
  Assignment A = ∅;
  while (true)
    /* Simplify under A. */
    (R,A) = qbcp(ψ,A);
    if (R == UNDEF)
      /* Decision making. */
      A = assign_dec_var(ψ,A);
    else
      /* Backtracking. */
      /* R == UNSAT/SAT */
      B = analyze(R,A);
      if (B == INVALID)
        return R;
      else
        A = backtrack(B);
```

Boolean Constraint Propagation for QBF (1)

Definition (Unit Literal Detection [CGS98])

- Given a QBF ψ, a clause $C \in \psi$ is *unit* iff $C = (l)$ and $q(l) = \exists$.
- *Unit literal detection* $UL(C) := \{l\}$ collects the assignment $\{l\}$ from the unit clause $C = (l)$.
- Unit literal detection on a QBF ψ: $UL(\psi) := \bigcup_{C \in \psi} UL(C)$.

Definition (Pure Literal Detection [CGS98])

- A literal l is *pure* in a QBF ψ if there are clauses which contain l but no clauses which contain $\neg l$.
- *Pure literal detection (PL)* assigns $var(l)$ of an existential (universal) pure literal l so that clauses are satisfied (not satisfied, i.e., shortened)

Boolean Constraint Propagation for QBF (2)

Definition

Boolean Constraint Propagation for QBF (QBCP):

- Given a PCNF ψ and the empty assignment $A = \{\}$, i.e. $\psi[A] = \psi$.
  1. Apply universal reduction (UR) to $\psi[A]$.
  2. Apply unit literal detection (UL) to $\psi[A]$ to get new assignments.
  3. Apply pure literal detection (PL) to $\psi[A]$ to find new assignments.
- Add assignments found by UL and PL to A, repeat steps 1-3.
- Stop if A does not change anymore or if $\psi[A] = \top$ or $\psi[A] = \bot$.

Properties of QBCP:

- QBCP takes a PCNF ψ and an assignment A and produces an extended assignment A' and a PCNF $\psi' = \psi[A']$ by UL, PL, and UR.
- Soundness: $\psi \equiv_{sat} \psi'$ (satisfiability-equivalence).
- No prefix ordering restriction: QBCP potentially assigns any variables.

QBCP and Implication Graphs

Definition (Implication Graph (IG))

- Let ψ be the original QBF.
- Vertices: literals (assignments) in A made as decisions or by UL. Special vertex $\emptyset$ denoting a clause $C \in \psi$ such that $C[A] = \bot$ by UR.
- For assignments $\{l\}$ by UL from a unit clause $C[A]$: the clause $ante(l) := C$ with $C \in \psi$ is the *antecedent clause* of assignment $\{l\}$.
- Define $ante(\emptyset) = C$, for a clause $C \in \psi$ such that $C[A] = \bot$.
- Edges: $(x, y) \in E$ if y assigned by UL and literal $\neg x \in ante(y)$.

- Antecedent clauses in the original PCNF ψ are recorded.
- Implication graphs are constructed on the fly during QBCP.
- *Conflict*: assignment A such that QBCP on $\psi[A]$ produces empty clause $\emptyset$.
- *Conflict graph*: implication graph containing empty clause $\emptyset$.

Clause Learning in QCDCL

Example (Clause Learning)

$\psi = \exists x_1, x_3, x_4 \forall y_5 \exists x_2.(\bar{x}_1 \vee x_2) \wedge (x_3 \vee y_5 \vee \bar{x}_2) \wedge (x_4 \vee \bar{y}_5 \vee \bar{x}_2) \wedge (\bar{x}_3 \vee \bar{x}_4)$

Clause Learning in QCDCL

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 $A = \{x_1\}$:
  $\psi[\{x_1\}] = \exists x_3, x_4 \forall y_5 \exists x_2 . (x_2) \wedge (x_3 \vee y_5 \vee \bar{x}_2) \wedge (x_4 \vee \bar{y}_5 \vee \bar{x}_2) \wedge (\bar{x}_3 \vee \bar{x}_4)$

Clause Learning in QCDCL

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: $\psi[\{x_1, x_2\}] = \exists x_3, x_4 \forall y_5.(x_3 \vee y_5) \wedge (x_4 \vee \bar{y}_5) \wedge (\bar{x}_3 \vee \bar{x}_4)$.

Clause Learning in QCDCL

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: $\psi[\{x_1, x_2\}] = \exists x_3, x_4.(x_3) \wedge (x_4) \wedge (\bar{x}_3 \vee \bar{x}_4)$

Clause Learning in QCDCL

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.

Clause Learning in QCDCL

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.

Implication graph G:

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

Antecedent clauses:

$ante(x_2)$: $(\bar{x}_1 \vee x_2)$
$ante(x_3)$: $(x_3 \vee y_5 \vee \bar{x}_2)$
$ante(x_4)$: $(x_4 \vee \bar{y}_5 \vee \bar{x}_2)$
$ante(\emptyset)$: $(\bar{x}_3 \vee \bar{x}_4)$

Clause Learning in QCDCL (cont.)

Example (Clause Learning, continued)

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

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

Antecedent clauses:

$$\begin{aligned}
ante(x_2) &: \quad (\bar{x}_1 \lor x_2) \\
ante(x_3) &: \quad (x_3 \lor y_5 \lor \bar{x}_2) \\
ante(x_4) &: \quad (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \\
ante(\emptyset) &: \quad (\bar{x}_3 \lor \bar{x}_4)
\end{aligned}$$

- Start at $\emptyset$, select pivots in reverse assignment ordering:
  resolve antecedents of x_4, x_3.
- Q-resolution [BKF95] disallows tautologies like $(\bar{y}_5 \lor y_5 \lor \bar{x}_2)$!
- Pivot selection more complex than in CDCL for SAT solving.

$$\begin{tikzcd}[column sep=small]
(\bar{x}_3 \lor \bar{x}_4) \arrow[dr, no head] & & (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \arrow[dl, no head] & \\
& (\bar{x}_3 \lor \bar{y}_5 \lor \bar{x}_2) \arrow[dr, no head] & & (x_3 \lor y_5 \lor \bar{x}_2) \arrow[dl, no head] \\
& & \cancel{(\bar{y}_5 \lor y_5 \lor \bar{x}_2)} &
\end{tikzcd}$$

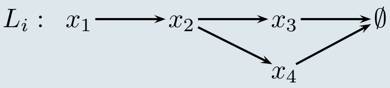
Clause Learning in QCDCL (cont.)

Example (Clause Learning, continued)

Prefix: $\exists x_1, x_3, x_4 \forall y_5 \exists x_2$

Assignment $A = \{x_1, x_2, x_3, x_4\}$

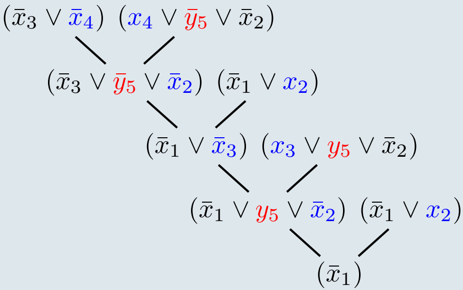
Implication graph G:



- To avoid tautologies, resolve on UR-blocking existentials.
- Select pivots: x_4, x_2, x_3, x_2.
- Potentially resolve on variables more than once to derive learned clause $C_L := (\neg x_1)$.

Antecedent clauses:

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



Clause Learning in QCDCL (cont.)

QCDCL by Traditional Q-Resolution [BKF95]:

- Avoid tautologies by appropriate pivot selection [GNT06].
- Problem: derivation of a learned clause may be exponential [VG12].
- Annotate nodes in conflict graph with intermediate resolvents, resulting in *tree-like* (instead of linear) Q-resolution derivations of learned clauses [LEG13].

QCDCL by Long Distance (LD) Q-Resolution [ZM02a, BJ12]:

- Key property: allow tautological resolvents of a certain kind.
- First implementation in QCDCL solver quaffle: `https://www.princeton.edu/~chaff/quaffle.html`.
- LDQ-resolution calculus is exponentially stronger than QRES.
- Practice: always select pivots in strict reverse assignment ordering.
  - Every resolution step is a valid LDQ-resolution step [ZM02a, ELW13].

Clause Learning in QCDCL (cont.)

Example (Clause Learning, continued)

Prefix: $\exists x_1, x_3, x_4 \forall y_5 \exists x_2$

Assignment $A = \{x_1, x_2, x_3, x_4\}$

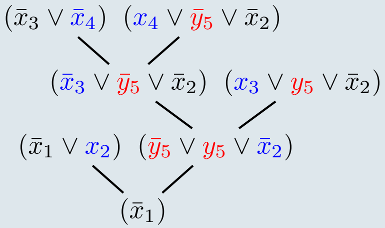
Implication graph G:

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

Antecedent clauses:

$$
\begin{aligned}
ante(x_2) &: \quad (\bar{x}_1 \lor x_2) \\
ante(x_3) &: \quad (x_3 \lor y_5 \lor \bar{x}_2) \\
ante(x_4) &: \quad (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \\
ante(\emptyset) &: \quad (\bar{x}_3 \lor \bar{x}_4)
\end{aligned}
$$

- Start at $\emptyset$, *always* select pivots in reverse assignment ordering: Resolve antecedents of x_4, x_3, x_2.
- Pivots obey order restriction of LDQ-resolution: $x_3 < y_5$
- To derive $C_L := (\neg x_1)$, resolve at most once on a variable.



QCDCL for Satisfiable QBFs

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

Let $\psi = \hat{Q}.\phi$ be a PCNF.

$$\frac{}{\text{C}} \quad \begin{array}{l} C = (\bigwedge_{l \in A}) \text{ is a cube where } \{x, \bar{x}\} \not\subseteq C \text{ and } A \text{ is an assignment} \\ \text{with } \psi[A] = \top \text{, i.e. every clause of } \psi \text{ satisfied.} \end{array} \quad (\textit{cu-init})$$

Cube Learning as a Proof System:

- Cube C by model generation: $v \in C$ ($\bar{v} \in C$) if v assigned to $\top$ ($\bot$).
- C (also called *cover set*): implicant of CNF ϕ, i.e. $C \Rightarrow \phi$.
- Model generation: a new axiom added to QRES.
- QRES for cubes: Q-resolution and *existential reduction* on cubes.
- PCNF ψ is satisfiable iff the empty cube can be derived from ψ.

QCDCL for Satisfiable QBFs

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

Let $\psi = \hat{Q}.\phi$ be a PCNF.

$$\frac{}{C}\quad \begin{array}{l} C = (\bigwedge_{l \in A}) \text{ is a cube where } \{x, \bar{x}\} \not\subseteq C \text{ and } A \text{ is an assignment} \\ \text{with } \psi[A] = \top \text{, i.e. every clause of } \psi \text{ satisfied.} \end{array} \quad (\textit{cu-init})$$

Example

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

$(\bar{x} \wedge u \wedge \bar{y}) \qquad (\bar{x} \wedge \bar{u} \wedge y)$

- By model generation: derive cubes $(\bar{x} \wedge u \wedge \bar{y})$ and $(\bar{x} \wedge \bar{u} \wedge y)$.

QCDCL for Satisfiable QBFs

Definition (Existential Reduction, cf. [GNT06, Let02, ZM02b])

Let C be a cube.

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

Example

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

$$\begin{array}{c} (\bar{x} \wedge u \wedge \bar{y}) \\ | \\ (\bar{x} \wedge u) \end{array} \qquad \begin{array}{c} (\bar{x} \wedge \bar{u} \wedge y) \\ | \\ (\bar{x} \wedge \bar{u}) \end{array}$$

- 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 $(\bar{x} \wedge \bar{u} \wedge y)$.

QCDCL for Satisfiable QBFs

Definition (Cube Resolution, cf. [GNT06, Let02, ZM02b])

Let C_1, C_2 be cubes.

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

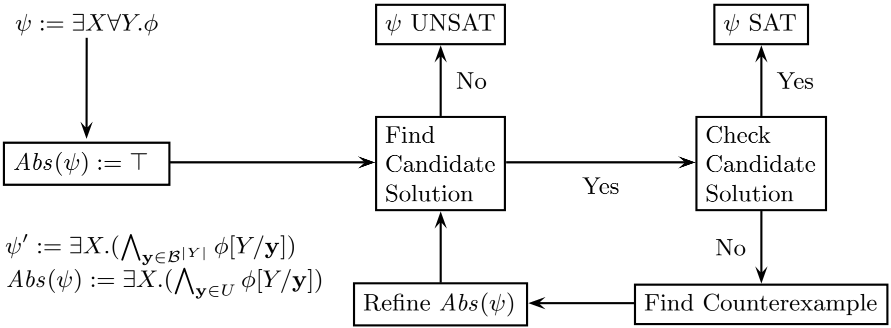
Example

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

$$\begin{tikzcd}
(\bar{x} \wedge u \wedge \bar{y}) \arrow[d, dash] & & (\bar{x} \wedge \bar{u} \wedge y) \arrow[d, dash] \\
(\bar{x} \wedge u) \arrow[dr, dash] & & (\bar{x} \wedge \bar{u}) \arrow[dl, dash] \\
& (\bar{x}) \arrow[d, dash] & \\
& \emptyset &
\end{tikzcd}$$

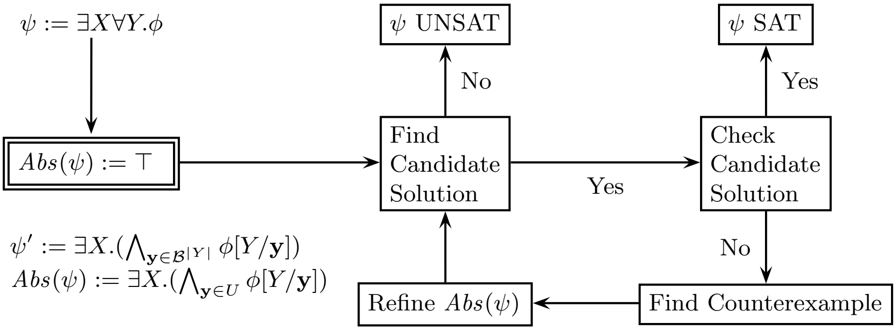
- By model generation: derive cubes $(\bar{x} \wedge u \wedge \bar{y})$ and $(\bar{x} \wedge \bar{u} \wedge y)$.
- By existential reduction: reduce trailing $\bar{y}$ from $(\bar{x} \wedge u \wedge \bar{y})$, y from $(\bar{x} \wedge \bar{u} \wedge y)$.
- Resolve $(\bar{x} \wedge \bar{u})$ and $(\bar{x}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

Solving QBF (2): Lazy Expansion by CEGAR



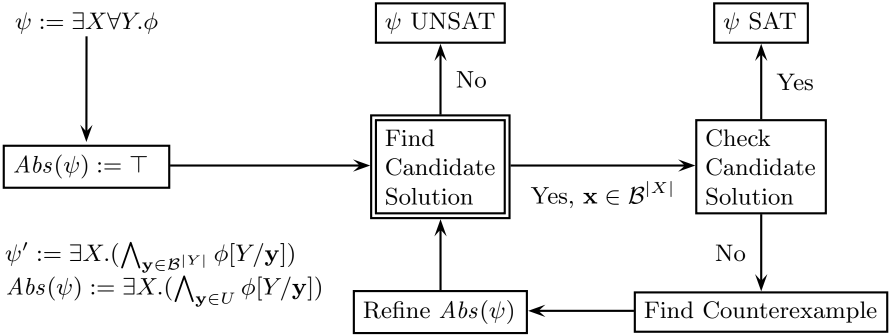
- Let $\psi := \exists X \forall Y.\ \phi$ be a one-alternation QBF, ϕ a non-CNF formula.
- ψ is satisfiable iff $\psi' := \exists X.(\bigwedge_{\mathbf{y} \in \mathcal{B}^{|Y|}} \phi[Y/\mathbf{y}])$ is satisfiable.
- Full expansion ψ' of $\forall Y$ by set $\mathcal{B}^{|Y|}$ of all possible assignments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

Solving QBF (2): Lazy Expansion by CEGAR



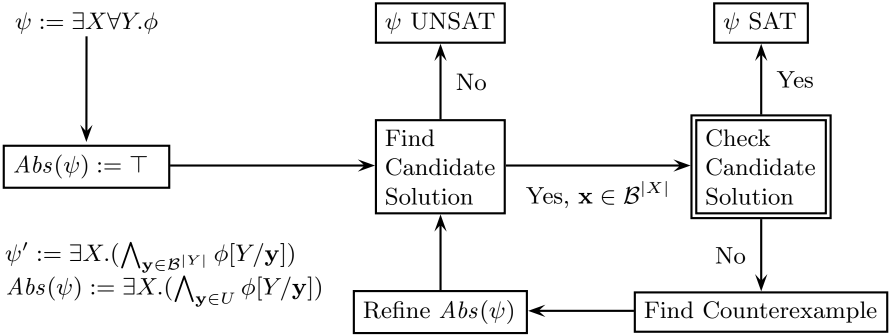
- Subset $U \subseteq \mathcal{B}^{|Y|}$ of set $\mathcal{B}^{|Y|}$ of all possible assignments $\mathbf{y}$ of Y.
- Partial expansion: given U, define $Abs(\psi) := \exists X. (\bigwedge_{\mathbf{y} \in U} \phi[Y/\mathbf{y}])$.
- Abstraction $Abs(\psi)$: if $Abs(\psi)$ unsatisfiable, then also ψ unsatisfiable.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

Solving QBF (2): Lazy Expansion by CEGAR



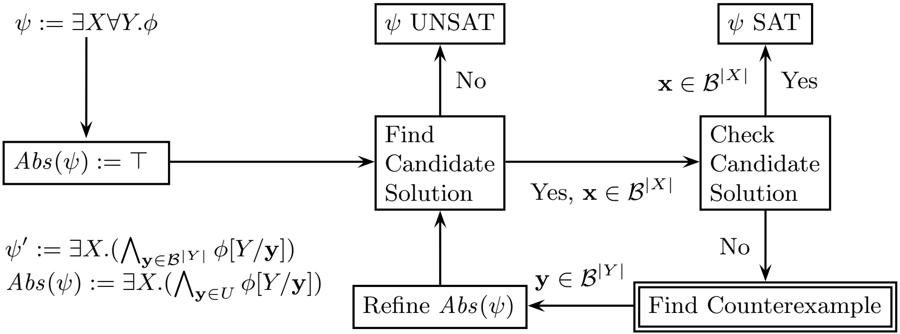
- Check satisfiability of $Abs(\psi)$ using a SAT solver.
- If $Abs(\psi)$ unsatisfiable: also ψ unsatisfiable, terminate.
- If $Abs(\psi)$ satisfiable: let $\mathbf{x} \in \mathcal{B}^{|X|}$ be a model of $Abs(\psi)$.
- $\mathbf{x} \in \mathcal{B}^{|X|}$: candidate solution of full exp. $\psi' := \exists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

Solving QBF (2): Lazy Expansion by CEGAR



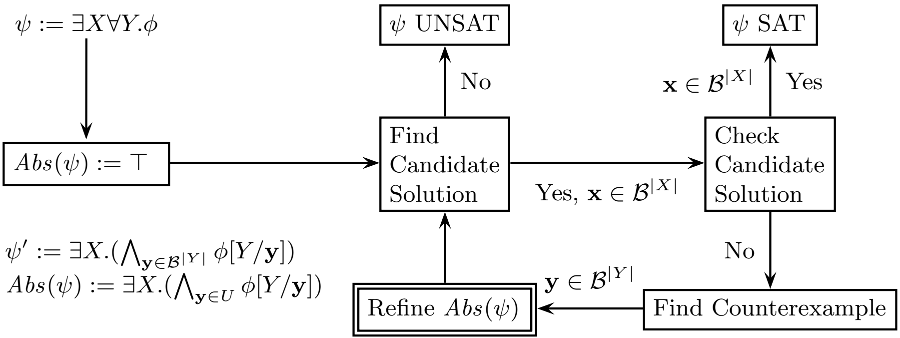
- If $\mathbf{x}$ is also a model of the full expansion ψ', then ψ is satisfiable.
- $\mathbf{x}$ is a model of full expansion ψ' iff $\forall Y.\phi[X/\mathbf{x}]$ is satisfiable.
- $\forall Y.\phi[X/\mathbf{x}]$ is satisfiable iff $\exists Y.\neg\phi[X/\mathbf{x}]$ is unsatisfiable.
- Check satisfiability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

Solving QBF (2): 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'.

Solving QBF (2): 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 $\mathbf{x}$.
- E.g. for 2QBF [RTM04, BJS$^+$16], RAReQS (recursive) [JKMSC16].

Solving QBF (3): Use SAT Technology

Proposition

Given a PCNF $\psi := \hat{Q}.\phi$. If a clause C can be derived from ϕ by a SAT solver, then C can be derived from ψ by QU-resolution.

Coupling QCDCL with SAT Solving:

- Clauses learned from ϕ by CDCL are shared with QCDCL [SB05]
- Models of ϕ found by SAT solver guide search process in QCDCL
- SAT-based generalizations of Q-resolution axioms in QCDCL [LES16]

Nested and Levelized SAT Solving:

- Solve $\exists B_1.\phi_1 \wedge (\forall B_2.\phi_2)$ by solving $\exists B_1.\phi_1 \wedge (\exists B_2.\neg\phi_2)$ with nested SAT solvers, applicable to arbitrary nesting [BJT16, JTT16]
- Invoke two SAT solvers $S_\forall$ and $S_\exists$ with respect to quantifier blocks, prefix processed from left to right [THJ15]

Proofs and Certificates

Proofs and Certificates (1)

Q-Resolution Proofs:

- QCDCL solvers produce derivations P of the empty clause/cube.
- Proof P can be filtered out of derivations of all learned clauses/cubes.

Extracting Skolem/Herbrand Functions from Proofs:

- By inspection of P, run time linear in $|P|$ ($|P|$ can be exponential).
- Extraction from long-distance Q-resolution proofs [BJJW15].
- Approaches to compute winning strategies from P [GGB11, ELW13].

Definition (Extracting Herbrand functions [BJ11, BJ12])

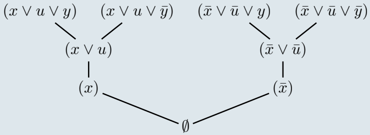
Let P be a proof (Q-resolution DAG) of the empty clause $\emptyset$.

- Visit clauses in P in topological ordering.
- Inspect universal reduction steps $C' = UR(C)$.
- Update Herbrand functions of variables u reduced from C by C'.

Proofs and Certificates (2)

Example (Extracting Herbrand Functions [BJ11, BJ12])

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



- Literal u reduced from $(x \vee u)$, update: $f_u(x) := (x)$.
- Literal $\bar{u}$ reduced from $(\bar{x} \vee \bar{u})$, update: $f_u(x) := f_u(x) \wedge \neg(\bar{x}) = (x)$.
- Unsatisfiable: $\psi[u/f_u(x)] = \exists x, y.(x \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge (\bar{x} \vee \bar{y})$

Proofs and Certificates (3)

Example

Let $\psi := \exists X \forall Y.\ \phi$ and $\psi' := \forall Y \exists X.\ \phi$ be one-alternation QBFs.

- If ψ satisfiable: all Skolem functions are constant.
- If ψ' unsatisfiable: all Herbrand functions are constant.
- No need to produce derivations of the empty clause/cube.
- QBF solvers can directly output values of Skolem/Herbrand functions.
- Useful for modelling and solving problems in Σ_2^P and Π_2^P.
- QDIMACS output format specification.

Outlook and Future Work

Outlook and Future Work

QBF in Practice:

- QBF tools are not (yet) a push-button technology.
- Pitfalls: Tseitin encodings, premature preprocessing.
- Goal: integrated workflow without the need for manual intervention.

Challenges:

- Extracting proofs and certificates in workflows including preprocessing [HSB14a, HSB14b] and incremental solving [MMLB12, LE14].
- Integrating *dependency schemes* [SS09, LB10, VG11, PSS16, PSS17] in workflows to relax the linear quantifier ordering.
- Implementations of QCDCL do not harness the full power of Q-resolution [Jan16].
- Combining strengths of orthogonal solving approaches.

Outlook and Future Work

- QBF is still an emerging field with plenty of applications.
- Assuming that NP $\neq$ PSPACE, QBF is more difficult than SAT...
- ...which is reflected in the complexity of solver implementations...
- ...but allows for exponentially more succinct encodings than SAT.
- Recent theoretical progress: QBF proof systems.
- Computational hardness motivates exploring alternative approaches: e.g. CEGAR-based expansion, computing Skolem functions [RS16].
- Expert and/or domain knowledge may be necessary for tuning.

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.

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.

Armin Biere, Alessandro Cimatti, Edmund M. Clarke, and Yunshan Zhu.
Symbolic Model Checking without BDDs.
In *TACAS*, volume 1579 of *LNCS*, pages 193–207. Springer, 1999.

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.

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

Olaf Beyersdorff, Leroy Chew, and Mikolas Janota.
Extension Variables in QBF Resolution.
Electronic Colloquium on Computational Complexity (ECCC), 23:5, 2016.
Beyond NP Workshop 2016 at AAAI-16.

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

Valeriy Balabanov and Jie-Hong R. Jiang.
Resolution Proofs and Skolem Functions in QBF Evaluation and Applications.
In *CAV*, volume 6806 of *LNCS*, pages 149–164. Springer, 2011.

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

Valeriy Balabanov, Jie-Hong Roland Jiang, Mikolas Janota, and Magdalena Widl.
Efficient Extraction of QBF (Counter)models from Long-Distance Resolution Proofs.
In *AAAI*, pages 3694–3701. AAAI Press, 2015.

References III

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.

Bart Bogaerts, Tomi Janhunen, and Shahab Tasharrofi.
Solving QBF Instances with Nested SAT Solvers.
In *Beyond NP Workshop 2016 at AAAI-16*, 2016.

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

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

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

References IV

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.

Michael Cashmore, Maria Fox, and Enrico Giunchiglia.
Partially grounded planning as quantified boolean formula.
In Daniel Borrajo, Subbarao Kambhampati, Angelo Oddi, and Simone Fratini, editors, *Proceedings of the Twenty-Third International Conference on Automated Planning and Scheduling, ICAPS 2013, Rome, Italy, June 10-14, 2013*. AAAI, 2013.

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.

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.

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

References V

Uwe Egly.
On the Value of Antiprenexing.
In *LPAR*, volume 822 of *LNCS*, pages 69–83. Springer, 1994.

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

Uwe Egly, Martin Kronegger, Florian Lonsing, and Andreas Pfandler.
Conformant planning as a case study of incremental QBF solving.
Ann. Math. Artif. Intell., 80(1):21–45, 2017.

Uwe Egly, Florian Lonsing, and Magdalena Widl.
Long-Distance Resolution: Proof Generation and Strategy Extraction in Search-Based QBF Solving.
In *LPAR*, volume 8312 of *LNCS*, pages 291–308. Springer, 2013.

Uwe Egly, Martina Seidl, Hans Tompits, Stefan Woltran, and Michael Zolda.
Comparing Different Prenexing Strategies for Quantified Boolean Formulas.
In *SAT*, volume 2919 of *LNCS*, pages 214–228. Springer, 2003.

References VI

Uwe Egly, Hans Tompits, and Stefan Woltran.
On Quantifier Shifting for Quantified Boolean Formulas.
In *In Proceedings of the SAT-02 Workshop on Theory and Applications of Quantified Boolean Formulas (QBF-02*, pages 48–61, 2002.

Peter Faymonville, Bernd Finkbeiner, Markus N. Rabe, and Leander Tentrup.
Encodings of Bounded Synthesis.
In *TACAS*, volume 10205 of *LNCS*, pages 354–370. Springer, 2017.

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

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

Alexandra Goultiaeva, Allen Van Gelder, and Fahiem Bacchus.
A Uniform Approach for Generating Proofs and Strategies for Both True and False QBF Formulas.
In *IJCAI*, pages 546–553. IJCAI/AAAI, 2011.

References VII

GhostQ: A QBF Solver, 2010–2016.
http://www.cs.cmu.edu/~wklieber/ghostq/.

Olivier Gasquet, Dominique Longin, Frederic Maris, Pierre Régnier, and Maël Valais.
Compact tree encodings for planning as QBF.
Inteligencia Artif., 21(62):103–114, 2018.

Enrico Giunchiglia, Paolo Marin, and Massimo Narizzano.
Reasoning with Quantified Boolean Formulas.
In *Handbook of Satisfiability*, volume 185 of *FAIA*, pages 761–780. IOS Press, 2009.

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

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

E. Giunchiglia, M. Narizzano, and A. Tacchella.
Quantifier Structure in Search-Based Procedures for QBFs.
TCAD, 26(3):497–507, 2007.

References VIII

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

Marijn Heule, Martina Seidl, and Armin Biere.
A Unified Proof System for QBF Preprocessing.
In *IJCAR*, volume 8562 of *LNCS*, pages 91–106. Springer, 2014.

Marijn Heule, Martina Seidl, and Armin Biere.
Efficient extraction of Skolem functions from QRAT proofs.
In *FMCAD*, pages 107–114. IEEE, 2014.

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.

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

References IX

Toni Jussila and Armin Biere.
Compressing BMC Encodings with QBF.
Electr. Notes Theor. Comput. Sci., 174(3):45–56, 2007.

Toni Jussila, Armin Biere, Carsten Sinz, Daniel Kröning, and Christoph M. Wintersteiger.
A First Step Towards a Unified Proof Checker for QBF.
In *SAT*, volume 4501 of *LNCS*, pages 201–214. Springer, 2007.

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

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

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

References X

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

Tomi Janhunen, Shahab Tasharrofi, and Eugenia Ternovska.
SAT-to-SAT: Declarative Extension of SAT Solvers with New Propagators.
In *AAAI*, pages 978–984. AAAI Press, 2016.

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.

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

Florian Lonsing and Armin Biere.
Integrating Dependency Schemes in Search-Based QBF Solvers.
In *SAT*, volume 6175 of *LNCS*, pages 158–171. Springer, 2010.

References XI

Florian Lonsing and Uwe Egly.
Incremental QBF Solving.
In *CP*, volume 8656 of *LNCS*, pages 514–530. Springer, 2014.

Florian Lonsing, Uwe Egly, and Allen Van Gelder.
Efficient clause learning for quantified boolean formulas via QBF pseudo unit propagation.
In *SAT*, volume 7962 of *LNCS*, pages 100–115. Springer, 2013.

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

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

Paolo Marin, Christian Miller, Matthew D. T. Lewis, and Bernd Becker.
Verification of partial designs using incremental QBF solving.
In *DATE*, pages 623–628. IEEE, 2012.

References XII

Andreas Nonnengart and Christoph Weidenbach.
Computing Small Clause Normal Forms.
In *Handbook of Automated Reasoning*, pages 335–367. Elsevier and MIT Press, 2001.

David A. Plaisted and Steven Greenbaum.
A Structure-Preserving Clause Form Translation.
J. Symb. Comput., 2(3):293–304, 1986.

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

Tomás Peitl, Friedrich Slivovsky, and Stefan Szeider.
Long Distance Q-Resolution with Dependency Schemes.
In *SAT*, volume 9710 of *LNCS*, pages 500–518. Springer, 2016.

Tomás Peitl, Friedrich Slivovsky, and Stefan Szeider.
Dependency Learning for QBF.
In *SAT*, volume 10491 of *LNCS*, pages 298–313. Springer, 2017.

QCIR-G14: A Non-Prenex Non-CNF Format for Quantified Boolean Formulas, 2014.
`http://qbf.satisfiability.org/gallery/qcir-gallery14.pdf`.

References XIII

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

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

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

Horst Samulowitz and Fahiem Bacchus.
Using SAT in QBF.
In *CP*, volume 3709 of *LNCS*, pages 578–592. Springer, 2005.

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.

References XIV

Marko Samer and Stefan Szeider.
Backdoor Sets of Quantified Boolean Formulas.
JAR, 42(1):77–97, 2009.

Irfansha Shaik and Jaco van de Pol.
Classical planning as QBF without grounding.
In Akshat Kumar, Sylvie Thiébaux, Pradeep Varakantham, and William Yeoh, editors, *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022, Singapore (virtual), June 13-24, 2022*, pages 329–337. AAAI Press, 2022.

Kuan-Hua Tu, Tzu-Chien Hsu, and Jie-Hong R. Jiang.
QELL: QBF Reasoning with Extended Clause Learning and Levelized SAT Solving.
In *SAT*, volume 9340 of *LNCS*, pages 343–359. Springer, 2015.

G. S. Tseitin.
On the Complexity of Derivation in Propositional Calculus.
Studies in Constructive Mathematics and Mathematical Logic, 1968.

Allen Van Gelder.
Variable Independence and Resolution Paths for Quantified Boolean Formulas.
In *CP*, volume 6876 of *LNCS*, pages 789–803. Springer, 2011.

References XV

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

Ralf Wimmer, Andreas Karrenbauer, Ruben Becker, Christoph Scholl, and Bernd Becker.
From DQBF to QBF by dependency elimination.
In *Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen, MBMV 2018, Tübingen, Germany, February 8-9, 2018*. Universität Tübingen, 2018.

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

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.