Advances in QBF Reasoning

Florian Lonsing

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

SAT/SMT/AR Summer School
June 22-25 2016, Lisbon, Portugal

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

Introduction (1)

Propositional Logic (SAT):

- Modelling NP-complete problems in formal verification, AI, ...
- Success story of SAT solving.

Quantified Boolean Formulas (QBF):

- Existential and universal quantification of propositional variables.
- $Q_1x_1, \ldots, Q_nx_n.\ \phi$, where $Q_i \in \{\forall, \exists\}$ and ϕ a CNF.
- PSPACE-complete: potentially more succinct encodings than SAT.

Practice:

- Despite intractability, solvers often work well on structured problems.
- Applications to presumably harder problems, e.g. NEXPTIME.
- SAT/QBF solvers are tightly integrated in application workflows.

Introduction (2): QBF-Related Quotes from the Literature

[BCCZ99] Armin Biere, Alessandro Cimatti, Edmund M. Clarke, Yunshan Zhu: Symbolic Model Checking without BDDs. TACAS 1999: 193-207.

> *Unfortunately, we do not know of an efficient decision procedure for QBF.*

Introduction (2): QBF-Related Quotes from the Literature

[DHK05] Nachum Dershowitz, Ziyad Hanna, Jacob Katz: Bounded Model Checking with QBF. SAT 2005: 408-414.

> *We found that modern state-of-the-art general-purpose QBF solvers are still unable to handle the real-life instances of BMC problems in an efficient manner.*

Introduction (2): QBF-Related Quotes from the Literature

[Rin07] Jussi Rintanen: Asymptotically Optimal Encodings of Conformant Planning in QBF. AAAI 2007: 1045-1050.

> *We believe that the future successes of QBF in many applications is strongly dependent on the development of better algorithms for evaluating QBF.*

Introduction (2): QBF-Related Quotes from the Literature

[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): 981-994 (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.*

Introduction (3): Progress in QBF Research

The Beginning of QBF Solving:

- 1998: DPLL for QBF [CGS98].
- 2002: CDCL for QBF [GNT02, Let02, ZM02a].
- 2002: expansion of variables [AB02].

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

Increased Interest in QBF:

- QBF proof systems: theoretical frameworks of solving techniques.
- CDCL and expansion as orthogonal approaches to QBF solving.
- QBF solving by counterexample guided abstraction refinement (CEGAR) [CGJ+03, JM15b, JKMSC16, RT15].

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).

Introduction (4): Motivating QBF Applications

Solving dependency quantified boolean formulas (NEXPTIME):

[FT14] Bernd Finkbeiner, Leander Tentrup: Fast DQBF Refutation. SAT 2014: 243-251.

Introduction (4): 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. To appear in the proceedings of CAV 2016.

Outline

Preliminaries:

- QBF syntax and semantics.

QBF 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 simplify and solve a QBF?
- How to obtain the solution to a problem from a solved QBF?

Outlook and Future Work:

- Open problems and possible research directions.

Preliminaries

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.

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.

Syntax (2)

Example (QDIMACS Format)

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

- 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.

Semantics (1)

Game-Based View:

- Player $P_\exists$ ($P_\forall$) assigns existential (universal) variables.
- Goal: $P_\exists$ ($P_\forall$) wants to satisfy (falsify) the formula.
- Players pick variables from left to right wrt. quantifier ordering.
- QBF ψ is satisfiable (unsatisfiable) iff $P_\exists$ ($P_\forall$) has a winning strategy.
- Winning strategy: $P_\exists$ ($P_\forall$) can satisfy (falsify) the formula regardless of opponent's choice of assignments.
- Close relation between winning strategies and QBF certificates.

Example

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

- $P_\exists$ wins by setting x to the same value as u.

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

Proof Systems (1): QBF Resolution

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

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

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

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

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

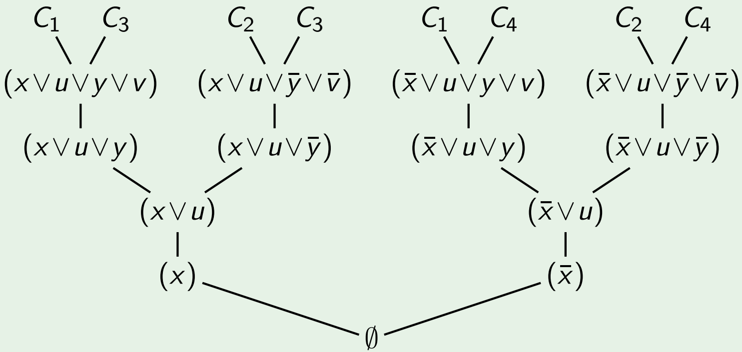
- Axiom *init*, universal reduction *red*, resolution *res*.
- PCNF ψ is unsatisfiable iff empty clause $\emptyset$ can be derived by QRES.

Proof Systems (2): QBF Resolution

Example

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

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



Proof Systems (3): QBF Resolution

Example (continued)

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

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

$C_1 \quad C_2$

$(v \vee \bar{v} \vee z)$

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: $p < x$.
- Exponentially stronger than traditional Q-resolution.

Proof Systems (3): QBF Resolution

Example (continued)

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

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

$$\frac{C_4 \qquad C_5}{(\bar{x} \vee \bar{z})}$$

QU-Resolution: [VG12]

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

Proof Systems (3): QBF Resolution

Example (continued)

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

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

$$\frac{C_4 \qquad C_5}{(\bar{x} \vee \bar{z})}$$

Further Variants: [BWJ14]

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

Proof Systems (4): 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 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).

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

- Idea: eliminate all universal variables, cf. Shannon expansion [Sha49].
- Finally, apply propositional resolution (no universal reduction).
- If x innermost: replace $\hat{Q}\forall x.\phi$ by $\hat{Q}.(\phi[x/\top] \wedge \phi[x/\top])$.
- Otherwise, duplicate existential variables inner to x [Bie04, BK07].
- Based on CNF, NNF, and-inverter graphs [AB02, LB08, PS09].

Proof Systems (5): 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.

Proof Systems (6): Expansion and Instantiation

Example (continued)

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

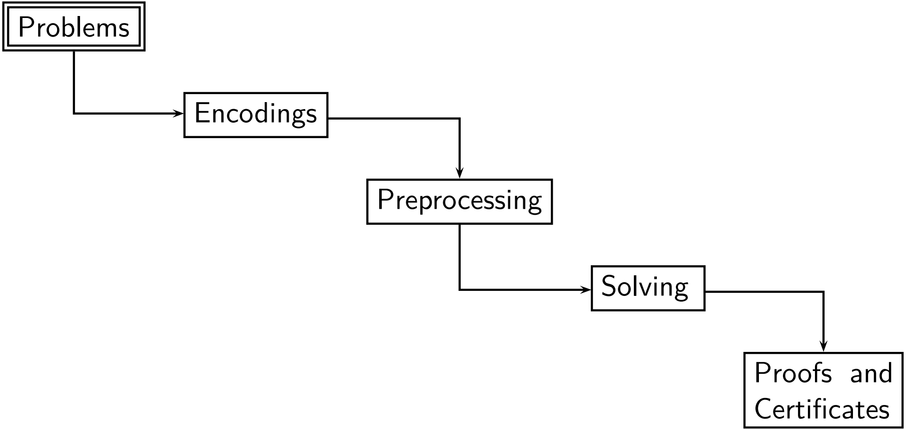
- Complete assignments: $A = \{\bar{u}\}$ and $A' = \{u\}$.
- Instantiate: $(\bar{x} \vee y^{\bar{u}}) \wedge (x \vee \bar{y}^{u}) \wedge (y^{u}) \wedge (\bar{y}^{\bar{u}})$
- Note: cannot resolve (y^{u}) and $(\bar{y}^{\bar{u}})$ due to mismatching annotations.
- Obtain (x) from $(x \vee \bar{y}^{u})$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

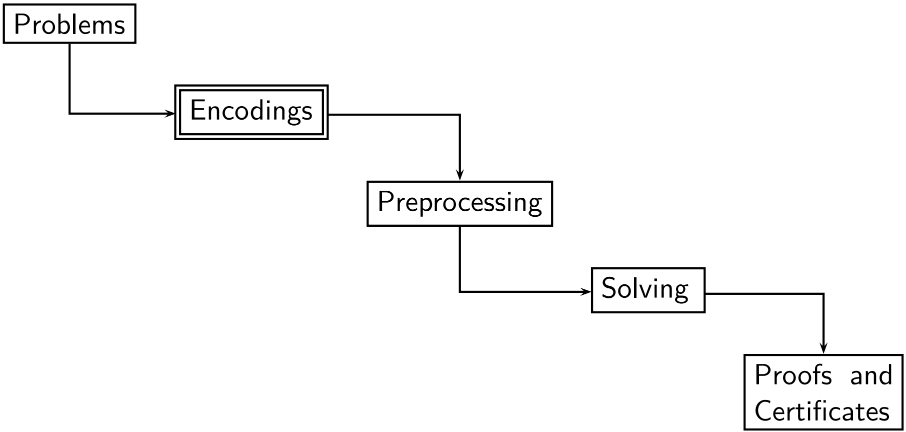
Typical QBF Workflow

Workflow Overview



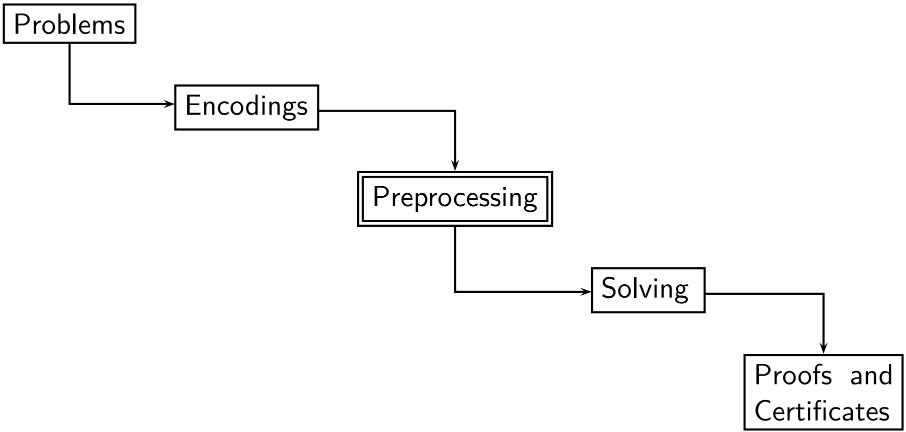
Which problems can be modelled as a QBF?

Workflow Overview



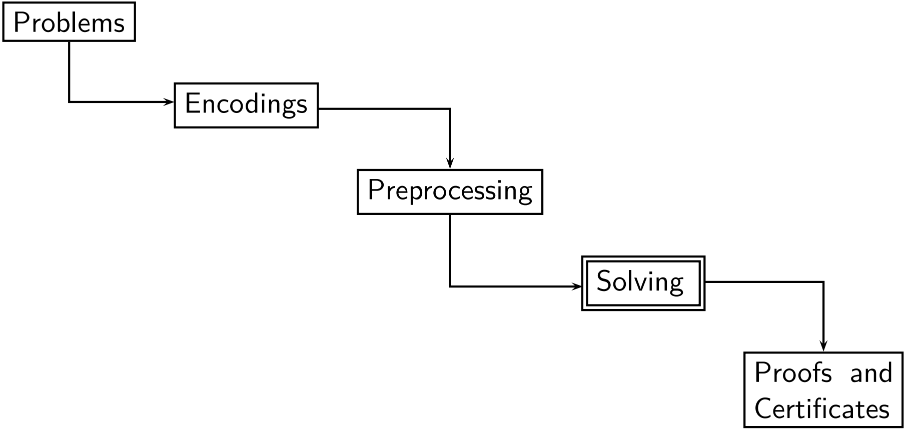
How to encode problems as a QBF?

Workflow Overview



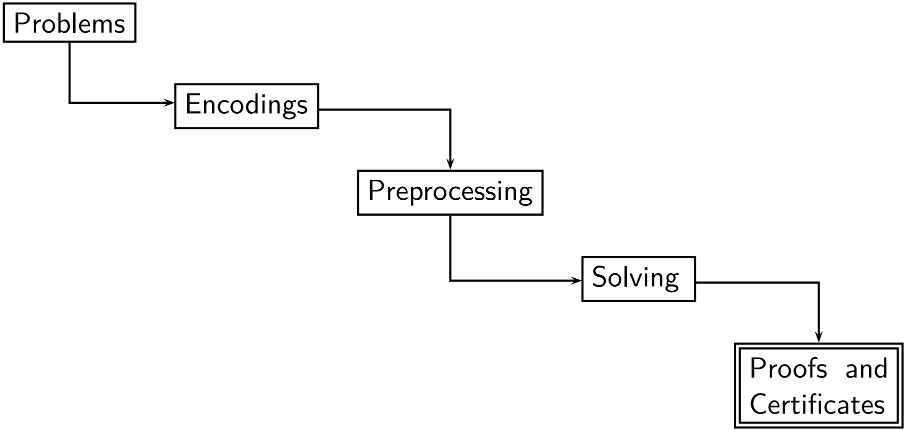
How to simplify QBF encodings?

Workflow Overview



How to solve a QBF?

Workflow Overview



How to obtain the solution to a problem from a solved QBF?

Problems (1)

Definition (Polynomial-Time Hierarchy, cf. [BB09, MS72])

For $k \geq 0$: $\Sigma_0^P := \Pi_0^P := P$, $\Sigma_{k+1}^P := NP^{\Sigma_k^P}$, $\Pi_{k+1}^P := co\Sigma_{k+1}^P$

- Σ_{k+1}^P: problems decidable in non-det. poly-time with Σ_k^P oracle.
- Π_{k+1}^P: class of problems whose complement is in Σ_{k+1}^P.
- $\Sigma_1^P = NP$, $\Pi_1^P = coNP$, every Σ_i^P, Π_i^P contained in PSPACE [Sto76].

Definition (Prefix Type [BB09])

A propositional formula ϕ has prefix type $\Sigma_0 = \Pi_0$. Given a QBF with prefix type Σ_n (Π_n), the QBF $\forall B.\phi$ ($\exists B.\phi$) has prefix type Π_{n+1} (Σ_{n+1}).

Proposition (cf. [BB09])

For $k \geq 1$, the satisfiability problem of a QBF ψ with prefix type Σ_k (Π_k) is Σ_k^P-complete (Π_k^P-complete).

Problems (2)

Class	*Prefix*	*Problems (e.g.)*
$\Sigma_1^P = NP$	$\exists B_1.\phi$	SAT, checking Herbrand function countermodels of QBFs [BJ12]
Σ_2^P	$\exists B_1 \forall B_2.\phi$	MUS membership testing [JS11b, Lib05], encodings of conformant planning [Rin07], ASP-related problems [FR05], abstract argumentation [CDG+15]
$\Pi_1^P = co\text{-}NP$	$\forall B_1.\phi$	Checking Skolem function models of QBFs [BJ12]
PSPACE	$Q_1 B_1 \ldots Q_n B_n.\phi$ (n depending on problem instance)	LTL model checking [SC85], NFA language inclusion, games [Sch78]

Problems (3): Using Universal Quantifiers

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

- System S, states of S as a state graph, invariant P.
- Goal: search for a counterexample of P of bounded length.

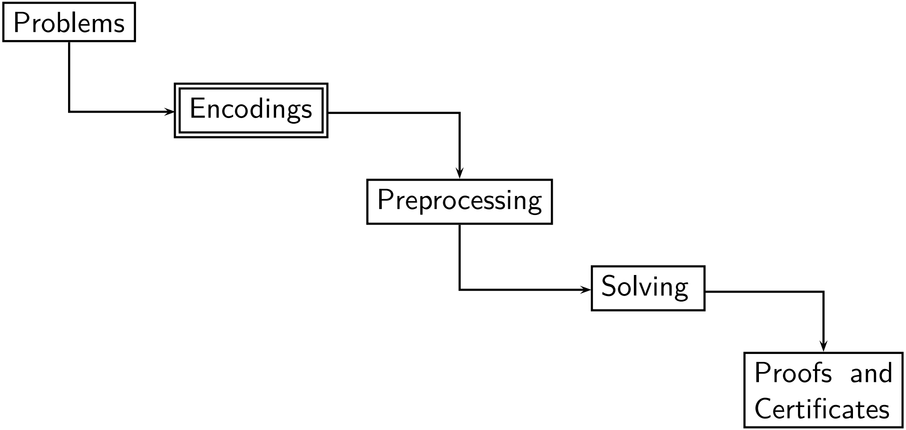
SAT Encoding:

- 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]

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

Workflow Overview



How can problems be encoded as a QBF?

Encodings (1)

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`

Encodings (2)

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]: $\psi := \hat{Q}\exists t_i.\phi$.

Encodings (3)

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.

Encodings (4): 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:

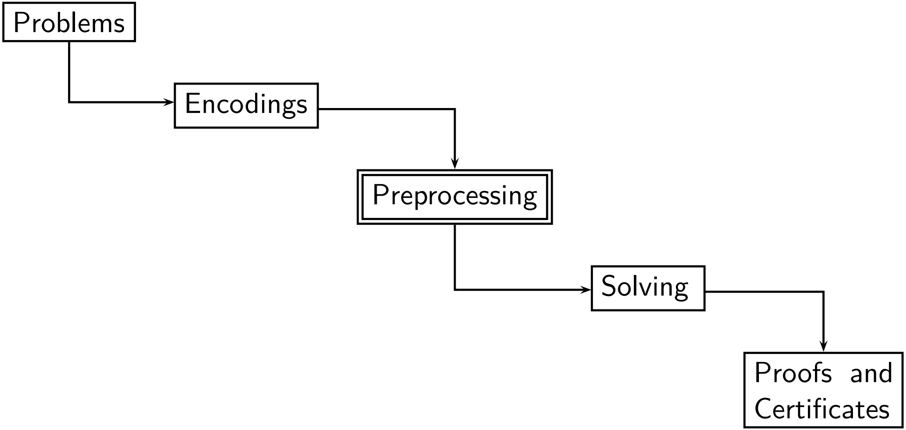
$$(t_1 \leftrightarrow XOR(x_1, x_2)) \land$$
$$\bigwedge_{1<i<n} (t_i \leftrightarrow XOR(t_{i-1}, x_{i+1})) \land$$
$$(t_n \leftrightarrow XOR(t_{n-1}, y)) \land (t_n)$$

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.*

Workflow Overview



How can QBF encodings be simplified?

Preprocessing (1)

Preprocessing as Incomplete Solving:

- Apply Q-resolution and expansion in restricted and bounded fashion.
- E.g. Bloqqer [BLS11, HJL$^+$15] and sQueezeBF[GMN10b].
- Failed literal detection [LB11, VGWL12]: find necessary assignments.

Reconstructing Structure:

- Recover non-CNF structure from Tseitin encodings [GB13, KSGC10].
- Move definition variables in prefix outwards, e.g. QParity function.

Effect on Solver Performance: [LSVG16]

- Iterative and incremental preprocessing may be powerful.
- Preprocessing may blur formula structure and thus be harmful.

Preprocessing (2)

Category/ Solvers	*Number Solved* Best Foot	Worst Foot
NO Bloqqer (solvers perform better without Bloqqer)		
bGhostQ-CEGAR	142	93
GhostQ-CEGAR	142	93
GhostQ	122	84
sDual_Ooq	118	99
sDual_Ooq	105	89
WANT Bloqqer (solvers perform better with Bloqqer)		
RAReQS	132	79
DepQBF-lazy-qpup	128	88
DepQBF	125	86
Hiqqer3	117	113
Qoq	93	65
QuBE	91	90
Nenofex	68	50

- QBF Gallery 2013 [LSVG16]: QBFLIB set (276 formulas).
- Solver performance with and without preprocessing by Bloqqer.
- Preprocessing may be harmful to the performance of some solvers.

Preprocessing (3): Prefix Ordering Matters

Definition (Blocking Literal, Blocked Clause [Kul99, BLS11, HJL+15])

Let $\psi = \hat{Q}.\phi$ be a PCNF and $C \in \phi$ a clause.

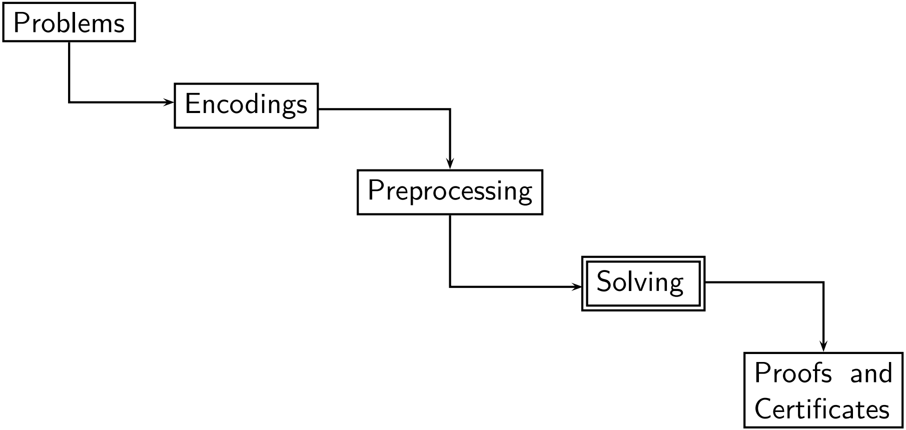
- *blocking literal* l: $l \in C$ with $q(l) = \exists$ such that for all $C' \in \phi$ with $\bar{l} \in C'$, there exists l' with $l' \leq l$ such that $\{l', \bar{l'}\} \subseteq (C \cup (C' \setminus \{\bar{l}\}))$.
- A clause C is *blocked* if it contains a blocking literal.
- Removing blocked clauses preserves satisfiability.

Example

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

- No clause in ψ is blocked.
- Informally, inspect all resolvents on potential blocking literals.
- Prefix ordering has to be taken into account in QBF preprocessing.

Solving (1)



How can a QBF be solved?

Solving (2): 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);
```

- High-level flow similar to CDCL for SAT.
- Generate assignments *A* by decision making and QBF-specific BCP.
- Decisions in prefix ordering.
- Interpret formula ψ under *A* and universal reduction.
- *A* is conflicting: clause learning.
- *A* is a CNF model: cube learning.
- Asserting clauses and cubes for backjumping.
- QCDCL solvers, e.g., [LB10a, GMN10a, KSGC10, ZM02b]

Solving (3): QCDCL

Definition (Unit Literal Detection [CGS98])

- Given a QBF ψ, a clause $C \in \psi$ is *unit* if $C = (l)$ and $q(l) = \exists$.
- *Unit literal detection (UL)* assigns $var(l)$ to satisfy the unit clause $C = (l)$.
- (If $q(l) = \forall$ then C is effectively empty by universal reduction.)

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 $\overline{l}$.
- *Pure literal detection (PL)* assigns $var(l)$ of an existential (universal) pure literal l so that clauses are satisfied (not satisfied, i.e. shortened).

Solving (4): QCDCL

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 UL to $\psi[A]$, record *antecedent clauses* $C \in \psi$ like in CDCL.
  3. Apply PL to $\psi[A]$.
- 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:

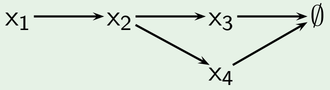
- Result: extended assignment A' and simplified PCNF $\psi' = \psi[A']$ by UL, PL, and UR such that $\psi \equiv_{sat} \psi'$.
- QBCP can assign variables out of prefix ordering.
- Construct implication graph like in BCP for SAT.

Solving (5): 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.

Conflict graph G:



Antecedent clauses:

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

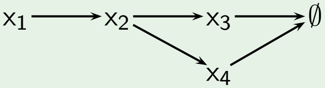
Solving (6): QCDCL

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:



Antecedent clauses:

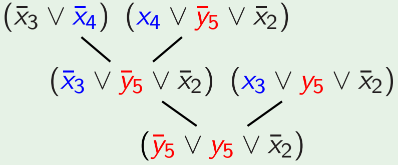
x_2 : $(\bar{x}_1 \vee x_2)$

x_3 : $(x_3 \vee y_5 \vee \bar{x}_2)$

x_4 : $(x_4 \vee \bar{y}_5 \vee \bar{x}_2)$

$\emptyset$: $(\bar{x}_3 \vee \bar{x}_4)$

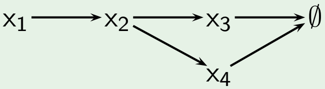
- Idea: start at $\emptyset$, select pivots in reverse assignment ordering.
- Resolve antecedents of x_4, x_3.
- Q-resolution [BKF95] disallows tautologies like $(\bar{y}_5 \vee y_5 \vee \bar{x}_2)$!
- Pivot selection more complex than in CDCL for SAT.



Solving (7): QCDCL

Example (Clause Learning, continued)

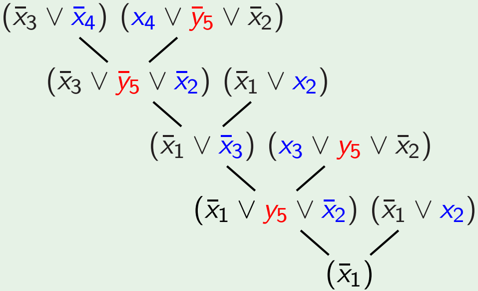
Prefix: $\exists x_1, x_3, x_4 \forall y_5 \exists x_2$
Assignment $A = \{x_1, x_2, x_3, x_4\}$
Conflict graph G:



Antecedent clauses:

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

- Avoid tautologies: resolve on UR-blocking existentials.
- Select pivots: x_4, x_2, x_3, x_2.
- Q-resolution derivation of a learned clause $(\bar{x}_1)$ is not regular, i.e. resolve on variables more than once.



Solving (8): QCDCL

Clause Learning by Traditional Q-Resolution [BKF95]:

- Avoid tautologies by appropriate pivot selection [GNT06].
- 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].

Clause Learning by Long Distance Q-Resolution [ZM02a, BJ12]:

- First implementation in quaffle: `https://www.princeton.edu/~chaff/quaffle.html`.
- Select pivots in strict reverse assignment ordering.
- Every resolution step is a valid LDQ-resolution step [ZM02a, ELW13].

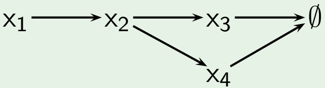
Solving (9): QCDCL

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:



Antecedent clauses:

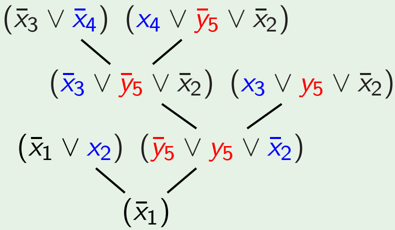
$x_2 : \quad (\bar{x}_1 \vee x_2)$

$x_3 : \quad (x_3 \vee y_5 \vee \bar{x}_2)$

$x_4 : \quad (x_4 \vee \bar{y}_5 \vee \bar{x}_2)$

$\emptyset : \quad (\bar{x}_3 \vee \bar{x}_4)$

- 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.
- Derivation of learned clause is regular, size linear in $|G|$.



Solving (10): QCDCL for Satisfiable QBFs

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

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

$$\frac{}{C}$$

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

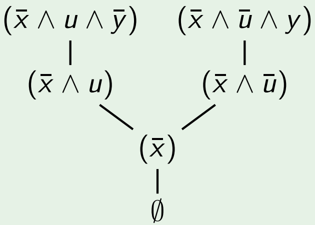
Cube Learning Dual to Clause Learning:

- 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 is an axiom of QRES.
- Q-resolution and *existential reduction* on cubes.
- Learn asserting cubes similar to asserting clauses.
- PCNF ψ is satisfiable iff the empty cube can be derived from ψ.

Solving (11): QCDCL for Satisfiable QBFs

Example

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



- 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 (12): QCDCL for Satisfiable QBFs

QCDCL and Cube Learning in Practice:

- PCNF $\psi := \hat{Q}.\,\phi$ with quantifier prefix $\hat{Q}$ and CNF ϕ.
- Original clauses ϕ, learned clauses θ and cubes γ.
- Properties: $\hat{Q}.\,\phi \equiv_{sat} \hat{Q}.\,(\phi \wedge \theta)$ and $\hat{Q}.\,\phi \equiv_{sat} \hat{Q}.\,(\phi \vee \gamma)$.

Problem: [RBM97, Let02]

- Easy formula with exponential DNF (and exponential cube proofs): $\psi = \forall u_1 \exists x_1 \ldots \forall u_n \exists x_n.\, \bigwedge_{i=1}^{n} [(u_i \vee \bar{x}_i) \wedge (\bar{u}_i \vee x_i)]$

Generalized Axioms: [LBB$^+$15, LES16]

- Generalize model generation (axiom) to derive shorter cubes C from assignments A in QCDCL where $\psi[A]$ is *satisfiable*.
- In general, $C \not\Rightarrow \phi$.

Solving (13): Lazy Expansion by CEGAR

Example ([CGJ+03, JS11a, JKMC12, JKMSC16])

Let $\psi := \exists X \forall Y.\, \phi$ be a one-alternation QBF, ϕ a non-CNF formula.

- ψ is satisfiable iff $\psi' := \bigwedge_{\mathbf{y} \in \mathcal{B}^{|Y|}} \phi[Y/\mathbf{y}]$ is satisfiable.
- ψ': full expansion of $\forall Y$ over all possible assignments **y** of Y.
- Let $U \subseteq \mathcal{B}^{|Y|}$ and $Abs(\psi) := \bigwedge_{\mathbf{y} \in U} \phi[Y/\mathbf{y}]$ be a partial expansion.
- If abstraction $Abs(\psi)$ is unsatisfiable, then ψ is unsatisfiable.
- Otherwise, consider a model (candidate solution) $\mathbf{x} \in \mathcal{B}^{|X|}$ of $Abs(\psi)$.
- If **x** is also a model of the full expansion ψ', then ψ is satisfiable.
  - **x** is a model of 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.
  - Let **y** be a model of $\exists Y.\neg\phi[X/\mathbf{x}]$, if one exists (counterexample to **x**).
- Otherwise, refine $Abs(\psi)$ by $U := U \cup \{\mathbf{y}\}$.

Used in 2QBF solving [RTM04, BJS+16], RAReQS solver (recursive).

Solving (14): The Use of 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.

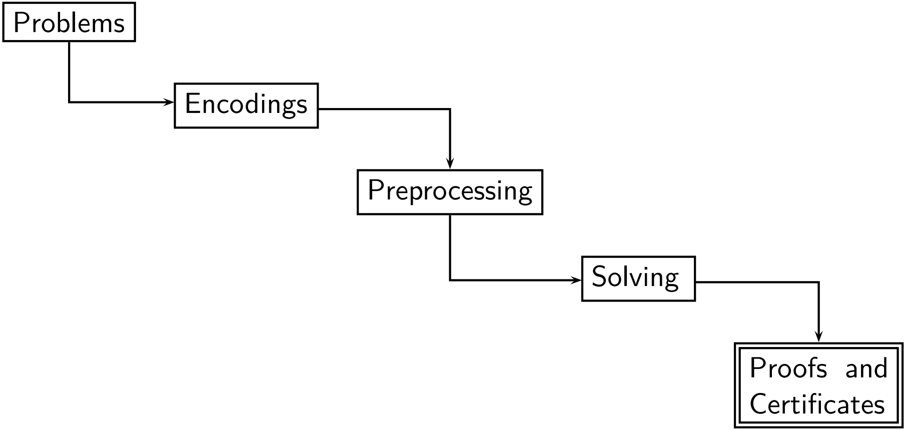
Coupling QCDCL with SAT Solving:

- 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 nestings [BJT16, JTT16].
- Invoke two SAT solvers $S_\forall$ and $S_\exists$ with respect to quantifier blocks, prefix processed from left to right [THJ15].

Workflow Overview



How to obtain the solution to a problem from a solved QBF?

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].

Proofs and Certificates (1)

Definition (Extracting Herbrand functions [BJ11, BJ12])

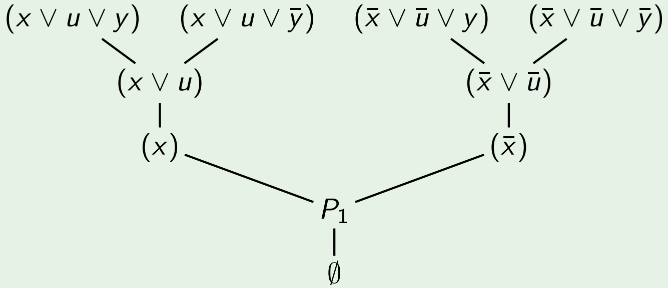
Let P be a proof (Q-resolution DAG) of the empty clause $\emptyset$.

- 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: $f_u(x) := f_u(x) \vee \neg(\bar{x}) = (x)$.
- 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): Special Case

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 (1)

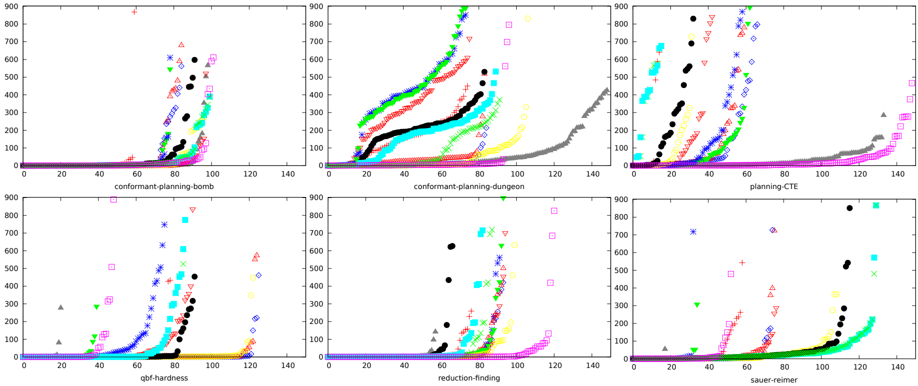
QBF in Practice:

- 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, LB10b, VG11, PSS16] 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 (2)



- QBF Gallery 2013 application benchmarks [LSVG16].
- 6 sets, 150 formulas each, 900 sec timeout, 7 GB memory limit.
- Diverse solver performance depending on implemented approaches.

Outlook and Future Work (3)

Take Home Messages:

- 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.
- The computational hardness of QBF motivates exploring alternative approaches (e.g. CEGAR, expansion) in addition to QCDCL.
- Number of quantifier alternations vs. observed hardness.
- Document and publish your tools and benchmarks!
- Upcoming QBFEVAL: `http://www.qbflib.org/qbfeval16.php`

Appendix

[Appendix] Syntax

Definition (QBFs as First-Order Logic Formulas [SLB12])

Mapping $\llbracket \cdot \rrbracket : QBF \rightarrow FOL$ with respect to unary FOL predicate p:

$$\llbracket \exists x.\phi \rrbracket = \exists x.\llbracket \phi \rrbracket \qquad \llbracket \forall x.\phi \rrbracket = \forall x.\llbracket \phi \rrbracket$$

$$\llbracket \phi \vee \psi \rrbracket = \llbracket \phi \rrbracket \vee \llbracket \psi \rrbracket \qquad \llbracket \phi \wedge \psi \rrbracket = \llbracket \phi \rrbracket \wedge \llbracket \psi \rrbracket$$

$$\llbracket x \rrbracket = p(x) \qquad \llbracket \neg \psi \rrbracket = \neg \llbracket \psi \rrbracket$$

$$\llbracket \top \rrbracket = p(\mathit{true}) \qquad \llbracket \bot \rrbracket = p(\mathit{false})$$

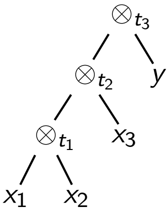
It holds that $p(\mathit{true})$ ($p(\mathit{false})$) is true (false) in every FOL interpretation.

Proposition ([SLB12])

The QBF ψ is satisfiable iff $\llbracket \psi \rrbracket \wedge p(\mathit{true}) \wedge \neg p(\mathit{false})$ is satisfiable.

[Appendix] Encodings: QParity

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

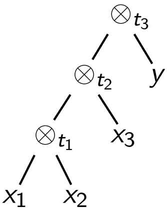


$t_1 \leftrightarrow XOR(x_1, x_2)$
$t_2 \leftrightarrow XOR(t_1, x_3)$
$t_3 \leftrightarrow XOR(t_2, y)$

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)

[Appendix] Encodings: QParity

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

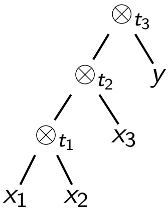


$$t_1 \leftrightarrow XOR(x_1, x_2)$$
$$t_2 \leftrightarrow XOR(t_1, x_3)$$
$$t_3 \leftrightarrow XOR(t_2, y)$$

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)

[Appendix] Encodings: QParity

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

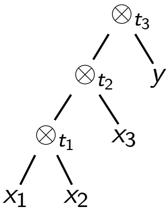


$$t_1 \leftrightarrow XOR(x_1, x_2)$$
$$t_2 \leftrightarrow XOR(t_1, x_3)$$
$$t_3 \leftrightarrow XOR(t_2, y)$$

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

[Appendix] Encodings: QParity

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



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

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

[Appendix] Solving: The Use of SAT Technology

Example (Clause Selection and Clausal Abstraction [JM15b, RT15])

Let $\psi := \forall X \exists Y.\ \phi$ be a one-alternation QBF, ϕ a CNF.

- ψ unsatisfiable iff, for some $\mathbf{x} \in \mathcal{B}^{|X|}$, $\exists Y.\ \phi[X/\mathbf{x}]$ unsatisfiable.
- Think of $\mathbf{x} \in \mathcal{B}^{|X|}$ as a selection $\phi_S^{\mathbf{x}} \subseteq \phi$ of clauses.
- Clause $C \in \phi_S^{\mathbf{x}}$ iff C not satisfied by $\mathbf{x}$, i.e. $C[X/\mathbf{x}] \neq \top$.
- If $\exists Y.\ \phi_S^{\mathbf{x}}[X/\mathbf{x}]$ unsatisfiable then $\exists Y.\ \phi[X/\mathbf{x}]$ and ψ unsatisfiable.
- Otherwise, consider model $\mathbf{y} \in \mathcal{B}^{|Y|}$ of $\exists Y.\ \phi_S^{\mathbf{x}}[X/\mathbf{x}]$.
- Find new $\mathbf{x}' \in \mathcal{B}^{|X|}$ such that there exists $C \in \phi_S^{\mathbf{x}'}$ with $C[Y/\mathbf{y}] \neq \top$.
- If no such $\mathbf{x}'$ exists then ψ is satisfiable.
- CEGAR: find candidate solutions $\mathbf{x}$ and counterexamples $\mathbf{y}$ by SAT solving, refinement step blocks unsuccessful selections $\phi_S^{\mathbf{x}}$.

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.

[BB09] Hans Kleine Büning and Uwe Bubeck.
Theory of Quantified Boolean Formulas.
In *Handbook of Satisfiability*, volume 185 of *FAIA*, pages 735–760. IOS Press, 2009.

[BCCZ99] 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.

[BCJ14] Olaf Beyersdorff, Leroy Chew, and Mikolas Janota.
On unification of QBF resolution-based calculi.
In *Proc. of the 39th Int. Symbosium on Mathematical Foundations of Computer Science (MFCS)*, volume 8635 of *LNCS*, pages 81–93. Springer, 2014.

References II

[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.

[BCJ16] 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.

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

[BJ11] 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.

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

References III

[BJJW15] 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.

[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.

[BJT16] Bart Bogaerts, Tomi Janhunen, and Shahab Tasharrofi.
Solving QBF Instances with Nested SAT Solvers.
In *Beyond NP Workshop 2016 at AAAI-16*, 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.

References IV

[BLS11] Armin Biere, Florian Lonsing, and Martina Seidl.
Blocked Clause Elimination for QBF.
In *CADE*, volume 6803 of *LNCS*, pages 101–115. Springer, 2011.

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

[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.

[CDG+15] Günther Charwat, Wolfgang Dvorák, Sarah Alice Gaggl, Johannes Peter Wallner, and Stefan Woltran.
Methods for solving reasoning problems in abstract argumentation - A survey.
Artif. Intell., 220:28–63, 2015.

[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.

References V

[CHR16] Chih-Hong Cheng, Yassine Hamza, and Harald Ruess.
Structural Synthesis for GXW Specifications.
CoRR, abs/1605.01153, 2016.
To appear in the proceedings of CAV 2016, LNCS, Springer.

[DHK05] Nachum Dershowitz, Ziyad Hanna, and Jacob Katz.
Bounded Model Checking with QBF.
In *SAT*, volume 3569 of *LNCS*, pages 408–414. Springer, 2005.

[Egl94] Uwe Egly.
On the Value of Antiprenexing.
In *LPAR*, volume 822 of *LNCS*, pages 69–83. Springer, 1994.

[Egl16] Uwe Egly.
On Stronger Calculi for QBFs.
CoRR, abs/1604.06483, 2016.
To appear in the proceedings of SAT 2016, LNCS, Springer.

[ELW13] 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.

References VI

[EST+03] 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.

[ETW02] 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.

[FR05] Wolfgang Faber and Francesco Ricca.
Solving Hard ASP Programs Efficiently.
In *LPNMR*, volume 3662 of *LNCS*, pages 240–252. Springer, 2005.

[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.

[GB13] Alexandra Goultiaeva and Fahiem Bacchus.
Recovering and Utilizing Partial Duality in QBF.
In *SAT*, volume 7962 of *LNCS*, pages 83–99. Springer, 2013.

References VII

[GGB11] 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.

[Gho16] GhostQ: A QBF Solver, 2010–2016.
`http://www.cs.cmu.edu/~wklieber/ghostq/`.

[GMN09] 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.

[GMN10a] Enrico Giunchiglia, Paolo Marin, and Massimo Narizzano.
QuBE7.0.
JSAT, 7(2-3):83–88, 2010.

[GMN10b] Enrico Giunchiglia, Paolo Marin, and Massimo Narizzano.
sQueezeBF: An Effective Preprocessor for QBFs Based on Equivalence Reasoning.
In *SAT*, volume 6175 of *LNCS*, pages 85–98. Springer, 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.

References VIII

[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.

[GNT07] E. Giunchiglia, M. Narizzano, and A. Tacchella.
Quantifier Structure in Search-Based Procedures for QBFs.
TCAD, 26(3):497–507, 2007.

[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.

[HJL+15] Marijn Heule, Matti Järvisalo, Florian Lonsing, Martina Seidl, and Armin Biere.
Clause Elimination for SAT and QSAT.
JAIR, 53:127–168, 2015.

[HSB14a] 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.

[HSB14b] Marijn Heule, Martina Seidl, and Armin Biere.
Efficient extraction of Skolem functions from QRAT proofs.
In *FMCAD*, pages 107–114. IEEE, 2014.

References IX

[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.

[JB07] Toni Jussila and Armin Biere.
Compressing BMC Encodings with QBF.
Electr. Notes Theor. Comput. Sci., 174(3):45–56, 2007.

[JBS+07] 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.

[JKMC12] Mikolás Janota, William Klieber, João Marques-Silva, and Edmund M. Clarke.
Solving QBF with counterexample guided refinement.
In *SAT*, volume 7317 of *LNCS*, pages 114–128. Springer, 2012.

References X

[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.

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

[JS11a] Mikolás Janota and João P. Marques Silva.
Abstraction-Based Algorithm for 2QBF.
In *SAT*, volume 6695 of *LNCS*, pages 230–244. Springer, 2011.

[JS11b] Mikolás Janota and João P. Marques Silva.
On Deciding MUS Membership with QBF.
In *CP*, volume 6876 of *LNCS*, pages 414–428. Springer, 2011.

References XI

[JTT16] 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.

[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.

[Kul99] Oliver Kullmann.
On a Generalization of Extended Resolution.
Discrete Applied Mathematics, 96-97:149–176, 1999.

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

[LB10a] Florian Lonsing and Armin Biere.
DepQBF: A Dependency-Aware QBF Solver.
JSAT, 7(2-3):71–76, 2010.

[LB10b] 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 XII

[LB11] Florian Lonsing and Armin Biere.
Failed Literal Detection for QBF.
In *SAT*, volume 6695 of *LNCS*, pages 259–272. Springer, 2011.

[LBB+15] Florian Lonsing, Fahiem Bacchus, Armin Biere, Uwe Egly, and Martina Seidl.
Enhancing Search-Based QBF Solving by Dynamic Blocked Clause Elimination.
In *LPAR*, volume 9450 of *LNCS*, pages 418–433. Springer, 2015.

[LE14] Florian Lonsing and Uwe Egly.
Incremental QBF Solving.
In *CP*, volume 8656 of *LNCS*, pages 514–530. Springer, 2014.

[LEG13] 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.

[LES16] Florian Lonsing, Uwe Egly, and Martina Seidl.
Q-Resolution with Generalized Axioms.
CoRR, abs/1604.05994, 2016.
To appear in the proceedings of SAT 2016, LNCS, Springer.

References XIII

[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.

[Lib05] Paolo Liberatore.
Redundancy in logic I: CNF propositional formulae.
Artif. Intell., 163(2):203–232, 2005.

[LSVG16] Florian Lonsing, Martina Seidl, and Allen Van Gelder.
The QBF Gallery: Behind the scenes.
Artif. Intell., 237:92–114, 2016.

[MMLB12] 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.

[MS72] Albert R. Meyer and Larry J. Stockmeyer.
The Equivalence Problem for Regular Expressions with Squaring Requires Exponential Space.
In *13th Annual Symposium on Switching and Automata Theory*, pages 125–129. IEEE Computer Society, 1972.

References XIV

[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.

[NW01] Andreas Nonnengart and Christoph Weidenbach.
Computing Small Clause Normal Forms.
In *Handbook of Automated Reasoning*, pages 335–367. Elsevier and MIT Press, 2001.

[PG86] David A. Plaisted and Steven Greenbaum.
A Structure-Preserving Clause Form Translation.
J. Symb. Comput., 2(3):293–304, 1986.

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

[PSS16] 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.

References XV

[QCI14] QCIR-G14: A Non-Prenex Non-CNF Format for Quantified Boolean Formulas, 2014.
http://qbf.satisfiability.org/gallery/qcir-gallery14.pdf.

[RBM97] Anavai Ramesh, George Becker, and Neil V. Murray.
CNF and DNF Considered Harmful for Computing Prime Implicants/Implicates.
JAIR, 18(3):337–356, 1997.

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

[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.

[SB05] Horst Samulowitz and Fahiem Bacchus.
Using SAT in QBF.
In *CP*, volume 3709 of *LNCS*, pages 578–592. Springer, 2005.

References XVI

[SC85] A. Prasad Sistla and Edmund M. Clarke.
The Complexity of Propositional Linear Temporal Logics.
J. ACM, 32(3):733–749, 1985.

[Sch78] Thomas J Schaefer.
On the Complexity of Some Two-Person Perfect-Information Games.
Journal of Computer and System Sciences, 16(2):185–225, 1978.

[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.

[SS09] Marko Samer and Stefan Szeider.
Backdoor Sets of Quantified Boolean Formulas.
JAR, 42(1):77–97, 2009.

[Sto76] Larry J. Stockmeyer.
The Polynomial-Time Hierarchy.
Theor. Comput. Sci., 3(1):1–22, 1976.

References XVII

[THJ15] 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.

[Tse68] G. S. Tseitin.
On the Complexity of Derivation in Propositional Calculus.
Studies in Constructive Mathematics and Mathematical Logic, 1968.

[VG11] Allen Van Gelder.
Variable Independence and Resolution Paths for Quantified Boolean Formulas.
In *CP*, volume 6876 of *LNCS*, pages 789–803. Springer, 2011.

[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.

[VGWL12] Allen Van Gelder, Samuel B. Wood, and Florian Lonsing.
Extended Failed-Literal Preprocessing for Quantified Boolean Formulas.
In *SAT*, volume 7317, pages 86–99. 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.

References XVIII

[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.