An Introduction to QBF Solving

Florian Lonsing

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

Second Indian SAT+SMT School
December 6-8 2017, Infosys Mysore Campus, Karnataka, India

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

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

- Origins: backtracking algorithms in 1960s [DP60, DLL62].
- Clause learning (CDCL): [SS96, SS99].
- Efficient data structures and heuristics: [MMZ\textsuperscript{+}01].
- SAT solver exploit structure of formulas.
- Despite intractability: many (industrial) applications.
Problem Solving using SAT:

- Problem encodings.
- Preprocessing (simplification).
- Solving.
- Result checking (proofs).
- Recent prominent example: Boolean Pythagorean Triples Problem [HKM16, HK17].

Many relevant search problems, from artificial intelligence to combinatorics, explore large search spaces to determine the presence or absence of a certain object. These problems are hard due to combinatorial explosion, and have traditionally been called infeasible. The brute-force method, which at least implicitly explores all possibilities, is a general approach to systematically search through such spaces. Brute force has long been regarded as suitable only for simple problems. This has changed in the last two decades, due to the progress in Satisfiability (SAT) solving, which by adding brute reason renders brute force into a powerful approach to deal with many problems easily and automatically. Search spaces with far more possibilities than the number of particles in the universe may be completely explored. SAT solving determines whether a formula in propositional logic has a solution, and its brute reasoning acts in a blind and uninformed way—as a feature, not a bug. We focus on applying SAT to mathematics, as a systematic development of the traditional method of proof by exhaustion. Can we trust the result of running complicated algorithms on many machines for a long time? The strongest solution is to provide a proof, which is also needed to show correctness of highly complex systems, which are everywhere, from finance to health care to aviation.
Quantified Boolean Formulas (QBF):

- Propositional logic extended by existential (∃) / universal (∀) quantification of propositional variables.
- Checking QBF satisfiability: PSPACE-complete.
- Propositional satisfiability (SAT): NP-complete.
- QBF encodings: potentially more succinct than propositional logic.

Example

- QBF $\psi := \hat{Q}\phi$ in *prenex conjunctive normal form (PCNF).*
- $\psi = \forall u \exists x. (\bar{u} \lor x) \land (u \lor \bar{x}).$
  - quantifier prefix $\hat{Q}$
  - propositional CNF $\phi$
Quantifier Alternations in PCNFs:

- A PCNF $Q_1B_1Q_2B_2 \ldots Q_nB_n. \phi$ has $n \geq 1$ quantifier blocks $Q_iB_i$.
- $Q_iB_i$: sets $B_i$ of variables, quantifiers $Q_i \in \{\forall, \exists\}$ with $Q_i \neq Q_{i+1}$.
- A PCNFs with $n$ quantifier blocks has $n - 1$ quantifier alternations.

Example

- PCNF $\psi = \exists x_1, x_2 \forall u_1, u_2 \exists x_3. \phi$.
- $\psi$ has two quantifier alternations.
- Quantifier blocks $\exists B_1, \forall B_2, \exists B_3$.
- $B_1 : \{x_1, x_2\}, B_2 : \{u_1, u_2\}, B_3 : \{x_3\}$.
**Polynomial Hierarchy (PH):** cf. [MS72, Sto76, Wra76]

- Framework to describe the complexity of problems beyond NP.
- Satisfiability problem of a given PCNF is located in PH.

**Proposition (cf. [BB09, MS72, Sto76, Wra76])**

- Let $\psi := Q_1B_1 \ldots Q_nB_n. \phi$ be a PCNF with $k \geq 0$ alternations.
- $Q_1 = \exists$: satisfiability problem of $\psi$ is $\Sigma^P_{k+1}$-complete.
- $Q_1 = \forall$: satisfiability problem of $\psi$ is $\Pi^P_{k+1}$-complete.
### Introduction (6): Encoding Problems as QBFs

<table>
<thead>
<tr>
<th>Class</th>
<th>Prefix Pattern</th>
<th>Problems (e.g.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Sigma^P_1 = NP$</td>
<td>$\exists B_1 . \phi$</td>
<td>Checking prop. logic satisfiability</td>
</tr>
<tr>
<td>$\Pi^P_1 = co-NP$</td>
<td>$\forall B_1 . \phi$</td>
<td>Checking prop. logic validity</td>
</tr>
<tr>
<td>$\Sigma^P_2$</td>
<td>$\exists B_1 \forall B_2 . \phi$</td>
<td>MUS membership testing [JS11, Lib05], encodings of conformant planning [Rin07], ASP-related problems [FR05], abstract argumentation [CDG+15]</td>
</tr>
<tr>
<td>$\Pi^P_2$</td>
<td>$\forall B_1 \exists B_2 . \phi$</td>
<td></td>
</tr>
<tr>
<td>\vdots</td>
<td>\vdots</td>
<td>\vdots</td>
</tr>
<tr>
<td>PSPACE</td>
<td>$Q_1 B_1 \ldots Q_n B_n . \phi$ ($n$ depending on problem instance)</td>
<td>LTL model checking [SC85], NFA language inclusion, games [Sch78]</td>
</tr>
</tbody>
</table>
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.

Initial

```
s_0 \rightarrow s_1 \rightarrow \ldots \rightarrow s_{k-1} \rightarrow s_k
```

Bad
Introduction (7): Compact QBF Encodings

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.

\[
I(s_0) \quad B(s_k)
\]

\[
\begin{array}{cccc}
S_0 & \xrightarrow{T(s_0, s_1)} & S_1 & \xrightarrow{T(\ldots)} \cdots & \xrightarrow{T(\ldots)} & S_{k-1} & \xrightarrow{T(s_{k-1}, s_k)} & S_k
\end{array}
\]

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 $k$: $I(s_0) \land T(s_0, s_1) \land \ldots \land T(s_{k-1}, s_k) \land B(s_k)$.
**Introduction (7): Compact QBF Encodings**

**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.

\[
I(s_0) \land B(s_k) \land \left[ \bigvee_{i=0}^{k-1} ((x = s_i) \land (x' = s_{i+1}) \right] \rightarrow T(x, x').
\]

**QBF Encoding: [BM08, JB07]**

- Only one copy of $T$ in contrast to $k$ copies in SAT encoding.
Introduction (8): Typical QBF Workflow

Problems → Encodings → Preprocessing → Solving → Proofs and Certificates
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].

⇒ compared to SAT (1960s), QBF still is a young field of 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
Synthesis and Realizability of Distributed Systems:


Solving Dependency Quantified Boolean Formulas (NEXPTIME):

Formal Verification and Synthesis:


Our Focus: Search-Based QBF Solving.
Outline of Tutorial

- Preliminaries:
  - Brief recapitulation: propositional logic.
  - QBF syntax and semantics.
- From backtracking search to modern search based QBF solving:
  - Basic backtracking approach.
  - Better assignment generation.
  - Backjumping.
  - Clause learning and Q-resolution.
  - Cube learning.
- QBF proofs and certificates.
- Preprocessing: blocked clause elimination (**hands-on session**).
- Expansion-based QBF solving.
- Experiments.
- Summary and conclusion.
Propositional Logic (1)

Definition (Basic Definitions)

- Boolean domain $\mathcal{B} = \{\top, \bot\}$: truth values “true” and “false”.
- Boolean variables $\text{Vars} = \{x, y, \ldots\}$ (arbitrarily many but finite).
- Assignment $A : \text{Vars} \to \mathcal{B}$
Definition (Propositional Formulas (PF))

- $\top$ and $\bot$ are PFs.
- For propositional variables $\mathit{Vars}$, $(x)$ where $x \in \mathit{Vars}$ is a PF.
- If $\psi$ is a PF then $\neg(\psi)$ is a PF.
- To save space in notation, we also write $\overline{x}$ instead of $\neg x$.
- If $\psi_1$ and $\psi_2$ are PFs then $(\psi_1 \circ \psi_2)$ is a PF, $\circ \in \{\land, \lor, \to, \leftrightarrow\}$.

Example

$$\psi := (y \land z) \to \neg(x)$$
**Definition (Conjunctive Normal Form (CNF))**

- A literal \( l \) is a variable \( x \) or its negation \( \bar{x} \).
- A clause \( C = (l_1 \lor \ldots \lor l_m) \) is a disjunction over literals.
- A formula is in *CNF* if it consists of a conjunction of clauses.
Definition (CNF Semantics)

- Given a CNF $\phi$ and an assignment $A$ to the variables in $\phi$.
- $\phi[A]$: replace variables $x$ in $\phi$ by $\top$ ($\bot$) if $A(x) = \top$ ($A(x) = \bot$).
- We write $A := \{x\}$ if $A(x) = \top$ and $A := \{\overline{x}\}$ if $A(x) = \bot$.
- CNF $\phi$ is satisfiable iff there exists $A$ such that $\phi[A] = \top$. Otherwise, $\phi$ is unsatisfiable.

Example

- $\phi := (x \lor \overline{y}) \land (\overline{x} \lor y)$.
- Models $M$ and $M'$ of $\phi$:
  - $M := \{x, y\}$ where $M(x) = M(y) = \top$.
  - $M' := \{\overline{x}, \overline{y}\}$ where $M'(x) = M'(y) = \bot$.
QBFs as Quantified Circuits:

- \( \top \) and \( \bot \) are QBFs.
- For propositional variables \( \text{Vars} \), \( (x) \) where \( x \in \text{Vars} \) is a QBF.
- If \( \psi \) is a QBF then \( \neg(\psi) \) is a QBF.
- If \( \psi_1 \) and \( \psi_2 \) are QBFs then \( (\psi_1 \circ \psi_2) \) is a QBF, \( \circ \in \{\land, \lor, \rightarrow, \leftrightarrow\} \).
- If \( \psi \) is a QBF and \( x \in \text{Vars}(\psi) \), then \( \forall x.(\psi) \) and \( \exists x.(\psi) \) are QBFs.

Example

\[
\psi := (\forall z. (\exists y. (y \land z))) \rightarrow \neg(\forall x. (x))
\]
QBFs in Prenex CNF: $\psi := \hat{Q}.\phi$

- Quantifier prefix $\hat{Q} = Q_1 B_1 \ldots Q_n B_n$, $Q_i \in \{\forall, \exists\}$, $Q_i \neq Q_j$, $B_i \subseteq \text{Vars}$, $(B_i \cap B_j) = \emptyset$.
- 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 $\phi$ over propositional variables $x_i$.
- Assume: $\phi$ does not contain free variables, all $x_i$ in $\hat{Q}$ appear in $\phi$.

Example

- PCNF $\psi = \forall u \exists x. (\bar{u} \lor x) \land (u \lor \bar{x})$.
- Linear ordering: $u < x$. 
Example (QDIMACS Format)

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

- 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.

QDIMACS format: [http://www.qbflib.org/qdimacs.html](http://www.qbflib.org/qdimacs.html)
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 $\psi$ is unsatisfiable.
- The QBF $\psi_1 \land \psi_2$ is satisfiable iff $\psi_1$ and $\psi_2$ are satisfiable.
- The QBF $\psi_1 \lor \psi_2$ is satisfiable iff $\psi_1$ or $\psi_2$ 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 $\psi$ by replacing $x$ in $\psi$ by $\bot$ ($\top$).
- The QBF $\exists x.(\psi)$ is satisfiable iff $\psi[\neg x]$ or $\psi[x]$ is satisfiable.

Definition

The QBFs $\psi$ and $\psi'$ are *satisfiability-equivalent* ($\psi \equiv_{sat} \psi'$) iff $\psi$ is satisfiable whenever $\psi'$ is satisfiable.
Observe: recursive evaluation assigns variables in prefix ordering.

The PCNF $\psi = \forall x \exists y. (x \lor \bar{y}) \land (\bar{x} \lor y)$ 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 $\psi = \exists y \forall x. (x \lor \bar{y}) \land (\bar{x} \lor y)$ 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)

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 $\psi$ 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

$\psi = \forall x \exists y. (x \lor \bar{y}) \land (\bar{x} \lor y)$.

- $P_\exists$ wins by setting $y$ to the same value as $x$. 
Backtracking Search

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

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

The PCNF \( \psi = \forall x \exists y. (x \lor \bar{y}) \land (\bar{x} \lor y) \) is satisfiable:

- Assign \( x \): \( \psi[x] = \exists y. (y) \)
  - Assign \( \bar{y} \): \( \psi[x, \bar{y}] = \bot \) unsatisfiable.
  - Backtrack, assign \( y \): \( \psi[x, y] = \top \) satisfiable.

- One subcase of \( \forall x \) completed.

- Assign \( \bar{x} \): \( \psi[\bar{x}] = \exists y. (\bar{y}) \)
  - Assign \( y \): \( \psi[\bar{x}, y] = \bot \) unsatisfiable.
  - Backtrack, assign \( \bar{y} \): \( \psi[\bar{x}, \bar{y}] = \top \) satisfiable.
Assignment $A$ extended tentatively (decision making, splitting).

Termination: no open subcases left, depending on quantifier type.

Backtracking: flipping of assignments depending on subcase.

⇒ refine workflow step by step.
The Need for Better Assignment Generation

Example

$$\psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4 . \left( \neg y_5 \lor x_4 \right) \land \left( y_5 \lor \neg x_4 \right) \land \left( x_1 \lor y_2 \lor \neg x_4 \right) \land \left( \neg x_1 \lor x_3 \lor \neg x_4 \right) \land \left( \neg y_2 \lor \neg x_3 \right).$$

- Worst case: $2^5$ branches to be explored by backtracking search.
- However: with better assignment generation, exploring a single branch is sufficient!
- **Goal:** make assignments that do not have to be flipped.
Definition (Unit Literal Detection [CGS98])

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

Example

$\psi := \forall y \exists x_1, x_2. (x_2) \land (\neg y \lor \neg x_2) \land (\neg y \lor x_1)$.

Clause $(x_2)$ is unit: $UL(\psi) = \{x_2\}$. 
Definition (Pure Literal Detection [CGS98])
- A literal \( l \) is pure in a QBF \( \psi \) if there are clauses which contain \( l \) but no clauses which contain \( \neg l \).
- Pure literal detection \( PL(\psi) := \bigcup \{ l' \} \) collects the assignment \( \{ l' \} \) such that \( l \) is pure and \( l' := l \) if \( q(l) = \exists \) and \( l' := \neg l \) if \( q(l) = \forall \).
- The variable of an existential (universal) pure literal is assigned so that clauses are satisfied (not satisfied) by that assignment.

Example (continued)
\[ \psi := \forall y \exists x_1, x_2. (x_2) \land (\neg y \lor \neg x_2) \land (\neg y \lor x_1). \]
The universal literal \( \neg y \) is pure: \( PL(\psi) = \{ y \} \).
\[ \psi[y] := \exists x_1, x_2. (x_2) \land (\neg x_2) \land (x_1). \]
Definition (Universal Reduction [BKF95])

Given a clause $C$, universal reduction (UR) of $C$ produces the clause

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

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

- UR deletes locally “trailing” universal literals, i.e., shortens clauses.

Example (continued)

$$\psi := \forall y \exists x_1, x_2. (x_2) \land (\neg y \lor \neg x_2) \land (\neg y \lor x_1).$$

By UL: $\psi[x_2] := \forall y \exists x_1. (\neg y) \land (\neg y \lor x_1).$

In $\psi[x_2]$: $UR((\neg y)) = \emptyset.$
Boolean Constraint Propagation for QBF (QBCP):

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

QBCP in Practice:

- Combine decision making and QBCP.
- Successively apply QBCP after assigning some $x$ as decision.
- Backtracking: no need to flip assignments made in QBCP.
QBCP Example

Example

- $\psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \\
  (\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3)$.
- No simplifications of $\psi$ by QBCP possible, make decision: $A = \{y_5\}$.
- $\psi[y_5] = \exists x_1 \forall y_2 \exists x_3, x_4. (x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3)$.
- By UL: $\psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1 \lor y_2) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3)$.
- By UR: $\psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3)$.
- By PL: $\psi[y_5, x_4, y_2] = \exists x_1 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg x_3)$.
- By UL: $\psi[y_5, x_4, y_2, x_1] = \exists x_3. (x_3) \land (\neg x_3)$.
- By UL: $\psi[y_5, x_4, y_2, x_1, x_3] = \bot$.
- By QBCP, we have shown: $\psi[y_5] \equiv \text{sat} \ \psi[y_5, x_4, y_2, x_1, x_3] \equiv \text{sat} \ \bot$.
- Since $y_5$ is universal: $\psi[y_5] \equiv \text{sat} \ \bot \equiv \text{sat} \ \psi$. 
\( \psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \\
(\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \)

- No simplifications of \( \psi \) by QBCP possible, make decision: \( A = \{ y_5 \} \).
- \( \psi[y_5] = \exists x_1 \forall y_2 \exists x_3, x_4. (x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \)

  - By UL: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1 \lor y_2) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3). \)
  - By UR: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3). \)
  - By PL: \( \psi[y_5, x_4, y_2] = \exists x_1 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg x_3). \)
  - By UL: \( \psi[y_5, x_4, y_2, x_1] = \exists x_3. (x_3) \land (\neg x_3). \)
  - By UL: \( \psi[y_5, x_4, y_2, x_1, x_3] = \bot. \)
  - By QBCP, we have shown: \( \psi[y_5] \equiv_{sat} \psi[y_5, x_4, y_2, x_1, x_3] \equiv_{sat} \bot. \)
  - Since \( y_5 \) is universal: \( \psi[y_5] \equiv_{sat} \bot \equiv_{sat} \psi. \)
Example

\[ \psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \]
\[ (\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \]

- No simplifications of \( \psi \) by QBCP possible, make decision: \( A = \{ y_5 \} \).
- \( \psi[y_5] = \exists x_1 \forall y_2 \exists x_3, x_4. (x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \)
- By UL: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1 \lor y_2) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3). \)
- By UR: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3). \)
- By PL: \( \psi[y_5, x_4, y_2] = \exists x_1 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg x_3). \)
- By UL: \( \psi[y_5, x_4, y_2, x_1] = \exists x_3. (x_3) \land (\neg x_3). \)
- By UL: \( \psi[y_5, x_4, y_2, x_1, x_3] = \bot. \)
- By QBCP, we have shown: \( \psi[y_5] \equiv_{sat} \psi[y_5, x_4, y_2, x_1, x_3] \equiv_{sat} \bot. \)
- Since \( y_5 \) is universal: \( \psi[y_5] \equiv_{sat} \bot \equiv_{sat} \psi. \)
QBCP Example

Example

- $\psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4.$
  $\neg y_5 \lor x_4 \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3).$

- No simplifications of $\psi$ by QBCP possible, make decision: $A = \{y_5\}$.

- $\psi[y_5] = \exists x_1 \forall y_2 \exists x_3, x_4. (x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3).$

- By UL: $\psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1 \lor y_2) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3).$

- By UR: $\psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3).$

- By PL: $\psi[y_5, x_4, y_2] = \exists x_1 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg x_3).$

- By UL: $\psi[y_5, x_4, y_2, x_1] = \exists x_3. (x_3) \land (\neg x_3).$

- By UL: $\psi[y_5, x_4, y_2, x_1, x_3] = \bot.$

- By QBCP, we have shown: $\psi[y_5] \equiv sat \ \psi[y_5, x_4, y_2, x_1, x_3] \equiv sat \bot.$

- Since $y_5$ is universal: $\psi[y_5] \equiv sat \bot \equiv sat \ \psi.$
Example

\[ \psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \\
\left( \neg y_5 \lor x_4 \right) \land \left( y_5 \lor \neg x_4 \right) \land \left( x_1 \lor y_2 \lor \neg x_4 \right) \land \left( \neg x_1 \lor x_3 \lor \neg x_4 \right) \land \left( \neg y_2 \lor \neg x_3 \right). \]

- No simplifications of \( \psi \) by QBCP possible, make decision: \( A = \{y_5\} \).
- \( \psi[y_5] = \exists x_1 \forall y_2 \exists x_3, x_4. (x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \)
- By UL: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1 \lor y_2) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3). \)
- By UR: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3). \)
- By PL: \( \psi[y_5, x_4, y_2] = \exists x_1 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg x_3). \)
- By UL: \( \psi[y_5, x_4, y_2, x_1] = \exists x_3. (x_3) \land (\neg x_3). \)
- By UL: \( \psi[y_5, x_4, y_2, x_1, x_3] = \bot. \)
- By QBCP, we have shown: \( \psi[y_5] \equiv_{sat} \psi[y_5, x_4, y_2, x_1, x_3] \equiv_{sat} \bot. \)
- Since \( y_5 \) is universal: \( \psi[y_5] \equiv_{sat} \bot \equiv_{sat} \psi. \)
Example

- \( \psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \\ (-y_5 \lor x_4) \land (y_5 \lor -x_4) \land (x_1 \lor y_2 \lor -x_4) \land (-x_1 \lor x_3 \lor -x_4) \land (-y_2 \lor -x_3). \)

- No simplifications of \( \psi \) by QBCP possible, make decision: \( A = \{ y_5 \} \).

- \( \psi[y_5] = \exists x_1 \forall y_2 \exists x_3, x_4. (x_4) \land (x_1 \lor y_2 \lor -x_4) \land (-x_1 \lor x_3 \lor -x_4) \land (-y_2 \lor -x_3). \)

- By UL: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1 \lor y_2) \land (-x_1 \lor x_3) \land (-y_2 \lor -x_3). \)

- By UR: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1) \land (-x_1 \lor x_3) \land (-y_2 \lor -x_3). \)

- By PL: \( \psi[y_5, x_4, y_2] = \exists x_1 \exists x_3. (x_1 \lor -x_1 \lor x_3) \land (-x_2 \lor x_3). \)

- By UL: \( \psi[y_5, x_4, y_2, x_1] = \exists x_3. (x_3) \land (-x_3). \)

- By UL: \( \psi[y_5, x_4, y_2, x_1, x_3] = \bot. \)

- By QBCP, we have shown: \( \psi[y_5] \equiv_{sat} \psi[y_5, x_4, y_2, x_1, x_3] \equiv_{sat} \bot. \)

- Since \( y_5 \) is universal: \( \psi[y_5] \equiv_{sat} \bot \equiv_{sat} \psi. \)
QBCP Example

Example

- $\psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \\
  (\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3)$.

- No simplifications of $\psi$ by QBCP possible, make decision: $A = \{y_5\}$.

- $\psi[y_5] = \\
  \exists x_1 \forall y_2 \exists x_3, x_4. (x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3)$.

- By UL: $\psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1 \lor y_2) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3)$.

- By UR: $\psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3)$.

- By PL: $\psi[y_5, x_4, y_2] = \exists x_1 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg x_3)$.

- By UL: $\psi[y_5, x_4, y_2, x_1] = \exists x_3. (x_3) \land (\neg x_3)$.

- By UL: $\psi[y_5, x_4, y_2, x_1, x_3] = \bot$.

- By QBCP, we have shown: $\psi[y_5] \equiv_{sat} \psi[y_5, x_4, y_2, x_1, x_3] \equiv_{sat} \bot$.

- Since $y_5$ is universal: $\psi[y_5] \equiv_{sat} \bot \equiv_{sat} \psi$. 
QBCP Example

Example

\[ \psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \]
\[ (\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \]

- No simplifications of \( \psi \) by QBCP possible, make decision: \( A = \{ y_5 \} \).

- \( \psi[y_5] = \exists x_1 \forall y_2 \exists x_3, x_4. (\neg y_5) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \)

- By UL: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1 \lor y_2) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3). \)

- By UR: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3). \)

- By PL: \( \psi[y_5, x_4, y_2] = \exists x_1 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg x_2). \)

- By UL: \( \psi[y_5, x_4, y_2, x_1] = \exists x_3. (x_3) \land (\neg x_3). \)

- By UL: \( \psi[y_5, x_4, y_2, x_1, x_3] = \bot. \)

- By QBCP, we have shown: \( \psi[y_5] \equiv_{sat} \psi[y_5, x_4, y_2, x_1, x_3] \equiv_{sat} \bot. \)

- Since \( y_5 \) is universal: \( \psi[y_5] \equiv_{sat} \bot \equiv_{sat} \psi. \)
Benefits of QBCP

Example

\[ \psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \]
\[ (\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \]

- **Worst case:** $2^5$ branches to be explored by backtracking search.
- **Only one branch explored.**
- **One decision + QBCP.**
- **Goal:** Integrate QBCP in workflow for better assignment generation.
Backtracking Search: Previous Abstract Workflow

PCNF $\psi$

$A = \emptyset$

$A' \subset A$, $A := A'$

Extend Assignment $A$

Subcase $\psi[A]$ Solved?

Analyze: Open Subcases?

UNSAT/SAT

YES

NO

YES

NO
QBCP influences assignment generation and detecting solved subcases.

In the following, we focus on conflicts, i.e., unsatisfiable subcases.

Need better ways of analyzing open subcases.
Definition (Implication Graph (IG))

- Let $\psi$ 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 $\psi$ are recorded.

Implication graphs are constructed on the fly during QBCP.

On the fly construction requires efficient data structures [GGN+04].

**Conflict**: assignment $A$ such that QBCP on $\psi[A]$ produces empty clause $\emptyset$.

**Conflict graph**: implication graph containing empty clause $\emptyset$. 
Example (formula from above)

\[ \psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \]
\[ (\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \]

Make decision: \( A = \{y_5\} \).

\[ \psi[y_5] = \exists x_1 \forall y_2 \exists x_3, x_4. (x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \]

Implication Graph:

\[ y_5 \]

Antecedents:
Constructing IGs On The Fly: Example

Example (formula from above)

\[ \psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \]
\[ (\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \]

By UL: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1 \lor y_2) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3). \)

By UR: \( \psi[y_5, x_4] = \exists x_1 \forall y_2 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg y_2 \lor \neg x_3). \)

Implication Graph:

\[ y_5 \rightarrow x_4 \]

Antecedents:

\[ \text{ante}(x_4) : (\neg y_5 \lor x_4) \]
Example (formula from above)

$$\psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \quad (\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3).$$

By PL: $$\psi[y_5, x_4, y_2] = \exists x_1 \exists x_3. (x_1) \land (\neg x_1 \lor x_3) \land (\neg x_3).$$

By UL: $$\psi[y_5, x_4, y_2, x_1] = \exists x_3. (x_3) \land (\neg x_3).$$

*Implication Graph:*

```
y_5 → x_4 → x_1
```

*Antecedents:*

$$ante(x_4) : (\neg y_5 \lor x_4)$$

$$ante(x_1) : (x_1 \lor y_2 \lor \neg x_4)$$
Constructing IGs On The Fly: Example

Example (formula from above)

\[ \psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \]
\[ (\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \]

By UL: \( \psi[y_5, x_4, y_2, x_1, x_3] = \bot. \)

**Implication Graph:**

- Antecedents:
  - \( \text{ante}(x_4) : (\neg y_5 \lor x_4) \)
  - \( \text{ante}(x_1) : (x_1 \lor y_2 \lor \neg x_4) \)
  - \( \text{ante}(x_3) : (\neg x_1 \lor x_3 \lor \neg x_4) \)
Example (formula from above)

\[ \psi = \forall y_5 \exists x_1 \forall y_2 \exists x_3, x_4. \]

\[ (\neg y_5 \lor x_4) \land (y_5 \lor \neg x_4) \land (x_1 \lor y_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_4) \land (\neg y_2 \lor \neg x_3). \]

By UL: \( \psi[y_5, x_4, y_2, x_1, x_3] = \bot. \)

**Implication Graph:**

\[ y_5 \rightarrow x_4 \rightarrow x_1 \rightarrow x_3 \rightarrow \emptyset \]

**Antecedents:**

\[ \text{ante}(x_4) : (\neg y_5 \lor x_4) \]

\[ \text{ante}(x_1) : (x_1 \lor y_2 \lor \neg x_4) \]

\[ \text{ante}(x_3) : (\neg x_1 \lor x_3 \lor \neg x_4) \]

\[ \text{ante}(\emptyset) : (\neg y_2 \lor \neg x_3) \]
Initially, the solver is at *decision level* $L_0$.

Every decision increases the current level $L_i$ by one to get $L_{i+1}$.

Assignments by QBCP are added to the current level $L_i$.

**Example**

\[
\psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi.
\]

No unit clauses present, level $L_0$ empty.
Decisions: Levelized Implication Graphs

- Initially, the solver is at *decision level* $L_0$.
- Every decision increases the current level $L_i$ by one to get $L_{i+1}$.
- Assignments by QBCP are added to the current level $L_i$.

**Example**

\[
\psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi.
\]

- Decisions on $x_1$.
- QBCP has no effect.

$L_0 :$

$L_1 : x_1$
Decisions: Levelized Implication Graphs

- Initially, the solver is at *decision level* $L_0$.
- Every decision increases the current level $L_i$ by one to get $L_{i+1}$.
- Assignments by QBCP are added to the current level $L_i$.

**Example**

$$
\psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi.
$$

- Decisions on $x_1, x_2$.
- QBCP has no effect.

\[
\begin{align*}
L_0 : \\
L_1 : x_1 \\
L_2 : x_2
\end{align*}
\]
Initially, the solver is at *decision level* $L_0$.

Every decision increases the current level $L_i$ by one to get $L_{i+1}$.

Assignments by QBCP are added to the current level $L_i$.

**Example**

$$\psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi.$$  

- Decisions on $x_1, x_2, x_3$: $A = \{x_1, x_2, x_3\}$.
- By QBCP (UL, UR): conflict $A = \{x_1, x_2, x_3, x_4, x_6\}$ at level $L_3$. 

\[ \begin{array}{c}
L_0 : \\
L_1 : x_1 \\
L_2 : x_2 \\
L_3 : x_3 \rightarrow x_4 \rightarrow x_6 \rightarrow \emptyset \\
\end{array} \]
Assignments:

- Represented as sequence $A = \{l_1, l_2, \ldots, l_n\}$ of literals.
- Assignments due to decision making and QBCP (UL, PL).
- Literals $l_i \in A$ are ordered chronologically as they were assigned.
- *Conflict*: assignment $A$ such that $\psi[A] = \bot$ under QBCP.
- *Solution*: assignment $A$ such that $\psi[A] = \top$ under QBCP.

$\Rightarrow$ we focus on conflicts and unsatisfiable QBFs.
Analyzing Open Subcases (2/2)

Chronological Backtracking:

- Given a conflict $A = \{\ldots, d, \ldots, l_n\}$, let $d$ be the most-recent unflipped existential decision.
- No such $d$ in $A$: formula solved.
- Retract decision $d$ and all later assignments: $A' = A \setminus \{d, \ldots, l_n\}$.
- Set the variable of $d$ to the opposite value (flip): $A' = A' \cup \{\neg d\}$.
- Continue with $A = A'$.

⇒ similar approach for solutions and satisfiable QBFs.
Example

\( \psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi. \)

- Assume that \( \phi \) contains further clauses.
Chronological Backtracking: Example (1/2)

Example

$$\psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi.$$  

- Decisions on $x_1, x_2, x_3$: $A = \{x_1, x_2, x_3\}$.
- $\psi[x_1, x_2, x_3] = \exists x_4 \forall y_5 \exists x_6. (x_4) \land (\neg x_4 \lor x_6) \land (y_5 \lor \neg x_6) \land \phi$.
- By QBCP (UL,UR): conflict $A_1 = \{x_1, x_2, x_3, x_4, x_6\}$.

Implication Graph of conflict $A_1$:

$$\begin{align*}
L_0 : & \\
L_1 : & x_1 \\
L_2 : & x_2 \\
L_3 : & x_3 \rightarrow x_4 \rightarrow x_6 \rightarrow \emptyset
\end{align*}$$
Example

\[ \psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi. \]

- Flip most recent unflipped decision \( x_3 \): \( A = \{x_1, x_2, \neg x_3\} \).
- \( \psi[x_1, x_2, \neg x_3] = \exists x_4 \forall y_5 \exists x_6. (x_4) \land (\neg x_4 \lor x_6) \land (y_5 \lor \neg x_6) \land \phi. \)
- Conflict \( A_2 = \{x_1, x_2, \neg x_3, x_4, x_6\} \), by UL,UR.

Implication Graph of conflict \( A_2 \):

\[
\begin{align*}
L_0 : & \\
L_1 : & x_1 \\
L_2 : & x_2 \\
L_3 : & \neg x_3 \rightarrow x_4 \rightarrow x_6 \rightarrow \emptyset
\end{align*}
\]
Example

\[ \psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi. \]

- Flip most recent unflipped \( x_2 \), decision on \( x_3 \): \( A = \{ x_1, \neg x_2, x_3 \} \).
- \( \psi[x_1, \neg x_2, x_3] = \exists x_4 \forall y_5 \exists x_6. (x_4) \land (\neg x_4 \lor x_6) \land (y_5 \lor \neg x_6) \land \phi. \)
- Conflict \( A_3 = \{ x_1, \neg x_2, x_3, x_4, x_6 \} \) by UL, UR.

Implication Graph of conflict \( A_3 \):

\[ L_0 : \]
\[ L_1 : x_1 \]
\[ L_2 : \neg x_2 \]
\[ L_3 : x_3 \rightarrow x_4 \rightarrow x_6 \rightarrow \emptyset \]
Chronological Backtracking: Example (1/2)

Example

\[ \psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi. \]

- Flip most recent unflipped decision \( x_3 \): \( A = \{ x_1, \neg x_2, \neg x_3 \} \).
- \( \psi[x_1, \neg x_2, \neg x_3] = \exists x_4 \forall y_5 \exists x_6. (x_4) \land (\neg x_4 \lor x_6) \land (y_5 \lor \neg x_6) \land \phi. \)
- Conflict \( A_4 = \{ x_1, \neg x_2, \neg x_3, x_4, x_6 \} \) by UL,UR.

Implication Graph of conflict \( A_4 \):

- Repeated assignments \( \{ x_3, x_4, x_6 \}, \{ \neg x_3, x_4, x_6 \} \) in \( A_1, A_3 \) and \( A_2, A_4 \).
Example (continued)

\[ \psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \vee x_4) \land (x_3 \vee x_4) \land (\neg x_4 \vee x_6) \land (\neg x_1 \vee y_5 \vee \neg x_6) \land \phi. \]

Conflicts generated:

- \( A_1 = \{x_1, x_2, x_3, x_4, x_6\} \).
- \( A_2 = \{x_1, x_2, \neg x_3, x_4, x_6\} \).
- \( A_3 = \{x_1, \neg x_2, x_3, x_4, x_6\} \).
- \( A_4 = \{x_1, \neg x_2, \neg x_3, x_4, x_6\} \).

- Same conflicting subtrees after flipping \( x_2 \).
- Decision \( x_2 \) is irrelevant in this context.

**Drawbacks of Chronological Backtracking:**

- Flipping variables which are irrelevant for the current conflict.
- Repeating subassignments of previous conflicts: needless branching.
Non-Chronological Backtracking: Backjumping (1/2)

- Given: conflict $A = \{l_1, l_2, \ldots, l_n\}$ and its implication graph (IG).
- Start at node $\emptyset$ and traverse IG backwards towards decision nodes.
- Compute \textit{conflict set (CS)}: collect all decisions $d_i$ reachable from $\emptyset$.
- $CS := \{d_1, \ldots, d_{i-1}, d_i, \ldots, d_k\}$ where $CS \subseteq A$.

**Example (continued)**

\[
\psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi.
\]

Consider conflict $A_1 = \{x_1, x_2, x_3, x_4, x_6\}$ with decisions $x_1, x_2, x_3$.

- $CS := \{x_1, x_3\}$. 

```
L_0 :
L_1 : x_1
L_2 : x_2
L_3 : x_3 -> x_4 -> x_6 -> \emptyset
```
Non-Chronological Backtracking: Backjumping (1/2)

- Let \( d_i \in CS \) be the most recent unflipped existential decision.
- No such \( d_i \): formula solved (i.e., unsatisfiable).
- Decision \( d_{i-1} \in CS \) was assigned before \( d_i \) most recently in \( CS \).

Example (continued)

\[
\psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \vee x_4) \land (x_3 \vee x_4) \land (\neg x_4 \vee x_6) \land (\neg x_1 \vee y_5 \vee \neg x_6) \land \phi.
\]

Consider conflict \( A_1 = \{x_1, x_2, x_3, x_4, x_6\} \) with decisions \( x_1, x_2, x_3 \).

- \( CS := \{x_1, x_3\} \).
- \( d_{i-1} = x_1, d_i = x_3 \).

\[
\begin{align*}
L_0 : & \\
L_1 : & x_1 \\
L_2 : & x_2 \\
L_3 : & x_3 \rightarrow x_4 \rightarrow x_6 \rightarrow \emptyset
\end{align*}
\]
Non-Chronological Backtracking: Backjumping (1/2)

- Update $A$ by retracting all assignments made after the level of $d_{i-1}$.
- Flip value of $d_i$ by making a new decision: $A := A \cup \{\neg d_i\}$.
- Backjumping relies on a more fine-grained analysis of the IG.
- To emulate chron. backtracking, let $CS$ contain all decisions made.

Example (continued)

$\psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi$.

Consider conflict $A_1 = \{x_1, x_2, x_3, x_4, x_6\}$ with decisions $x_1, x_2, x_3$.

- $d_{i-1} = x_1$, $d_i = x_3$.
- Retract $\{x_2, x_3, x_4, x_6\}$ from $A$.
- Flip $x_3$, $A \cup \{\neg x_3\} = \{x_1, \neg x_3\}$.

$L_0 :$ $L_1 : x_1$

$L_2 : \neg x_3$
Non-Chronological Backtracking: Backjumping (1/2)

- Update $A$ by retracting all assignments made after the level of $d_{i-1}$.
- Flip value of $d_i$ by making a new decision: $A := A \cup \{\neg d_i\}$.
- Backjumping relies on a more fine-grained analysis of the IG.
- To emulate chron. backtracking, let $CS$ contain all decisions made.

Example (continued)

$\psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi$.

Consider conflict $A_1 = \{x_1, x_2, x_3, x_4, x_6\}$ with decisions $x_1, x_2, x_3$.

- New conflict $A = \{x_1, \neg x_3, x_4, x_6\}$.
- $CS := \{x_1, \neg x_3\}$, $\neg x_3$ flipped already.
- $d_i = x_1$, retract entire $A$ (no $d_{i-1}$).
- Flip $x_1$, $A \cup \{\neg x_1\} = \{\neg x_1\}$.
Example (continued)

\[ \psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi. \]

Chronological backtracking:

Non-chronological backtracking:

- Backjumping potentially avoids irrelevant branches.
- Similar approaches for satisfiable QBFs.
Example (continued)

\[
\psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi.
\]

- Assume that the assignment tree on the right is a subtree of a bigger tree.
- Observation: every assignment \(A\) with \(\{x_1, x_4\} \subseteq A\) is a conflict (under QBCP).
- UL extends \(\{x_1, x_4\}\) to \(\{x_1, x_4, x_6\}\).
- \(C := (\neg x_1 \lor y_5 \lor \neg x_6)\) is empty under \(A := \{x_1, x_4, x_6\}\) and QBCP.
- Repeating \(\{x_1, x_4\} \subset A\) in other branches falsifies the same clause \(C\) under QBCP.
- Backjumping cannot avoid this problem.
Example (continued)

\[ \psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi. \]

- \[ C := (\neg x_1 \lor y_5 \lor \neg x_6) \] is empty under \[ A := \{x_1, x_4, x_6\} \] and QBCP.
- Repeating \[ \{x_1, x_4\} \subset A \] falsifies the same clause \[ C \] under QBCP.
- Adding the new clause \[ C_L := (\neg x_1 \lor \neg x_4) \] to the given formula \[ \psi \] prevents repetition of subassignment \[ \{x_1, x_4\} \].
- Assigning \[ x_1 \] (\[ x_4 \]) to true triggers assignment of \[ \neg x_4 \] (\[ \neg x_1 \]) by UL.
Towards Conflict Driven Clause Learning (QCDCL)

Clause Learning:
- Adding new clauses $C_L$ to given PCNF by analyzing a conflict.
- Learned clause prevents subassignments.
- Related to CDCL for SAT solving.
- CDCL: pioneered by solvers like GRASP or Chaff [SS99, MMZ⁺01].
- Correctness requirement: $\hat{Q}.\phi \equiv_{sat} \hat{Q}.(\phi \land C_L)$

$\Rightarrow$ deriving learned clauses by the $Q$-resolution calculus ($QRES$).
Chronological backtracking and backjumping: suboptimal analysis of open subcases.

Clause learning in QCDCL: stronger than backtracking/-jumping.
For now, we focus on unsatisfiable PCNFs.

- Learned clause $C_L$ derived by $QRES$ based on implication graphs.
- Formal foundation of clause learning: proof system $QRES$.
- Termination and backtracking controlled by properties of $C_L$. 
**Definition (Q-Resolution Calculus QRES, c.f. [BKF95])**

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

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(init)</td>
<td>$C$ for all $x \in \hat{Q}$: ${x, \bar{x}} \not\subseteq C$ and $C \in \phi$</td>
</tr>
<tr>
<td>(red)</td>
<td>$C \cup {l}$ for all $x \in \hat{Q}$: ${x, \bar{x}} \not\subseteq (C \cup {l})$, $q(l) = \forall$, and $l' &lt; l$ for all $l' \in C$ with $q(l') = \exists$</td>
</tr>
<tr>
<td>(res)</td>
<td>$C_1 \cup {p} \quad C_2 \cup {\bar{p}}$ for all $x \in \hat{Q}$: ${x, \bar{x}} \not\subseteq (C_1 \cup C_2)$, $\bar{p} \not\in C_1$, $p \not\in C_2$, and $q(p) = \exists$</td>
</tr>
</tbody>
</table>

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

\[ \psi = \exists x_1, x_2, x_3, x_4 \forall y_5 \exists x_6. (\neg x_3 \lor x_4) \land (x_3 \lor x_4) \land (\neg x_4 \lor x_6) \land (\neg x_1 \lor y_5 \lor \neg x_6) \land \phi. \]

Applying QRES:

- Axiom init selects initial clauses.
- Resolution on clauses by res using existential pivots.
- Reduction of trailing universal literals from clauses by red.

For clauses \( C_L \) derived from PCNF \( \hat{Q} \cdot \phi \) by QRES:

\( \hat{Q} \cdot \phi \equiv_{sat} \hat{Q} \cdot (\phi \land C_L) \).

QRES for clause learning: driven by conflicts and implication graphs.

Stronger, more flexible variants of QRES exist.
Example (Clause Learning)

\[ \psi = \exists x_1, x_3, x_4 \forall y_5 \exists x_2. (\bar{x}_1 \lor x_2) \land (x_3 \lor y_5 \lor \bar{x}_2) \land (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \land (\bar{x}_3 \lor \bar{x}_4) \]
### Example (Clause Learning)

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

- Make decision \( A = \{ x_1 \} \):
  \[ \psi[\{ x_1 \}] = \exists x_3, x_4 \forall y_5 \exists x_2. (x_2) \land (x_3 \lor y_5 \lor \bar{x}_2) \land (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \land (\bar{x}_3 \lor \bar{x}_4) \]
### QCDCL: Basic Idea (1/3)

**Example (Clause Learning)**

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

- **Make decision** \( A = \{ x_1 \} \):
  \[ \psi[\{ x_1 \}] = \exists x_3, x_4 \forall y_5 \exists x_2. (x_2) \land (x_3 \lor y_5 \lor \bar{x}_2) \land (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \land (\bar{x}_3 \lor \bar{x}_4) \]

- **By UL:** \( \psi[\{ x_1, x_2 \}] = \exists x_3, x_4 \forall y_5. (x_3 \lor y_5) \land (x_4 \lor \bar{y}_5) \land (\bar{x}_3 \lor \bar{x}_4) \).
QCDCL: Basic Idea (1/3)

Example (Clause Learning)

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

- Make decision \( A = \{x_1\} \):
  \[ \psi[\{x_1\}] = \exists x_3, x_4 \ \forall y_5 \exists x_2. (x_2) \land (x_3 \lor y_5 \lor \bar{x}_2) \land (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \land (\bar{x}_3 \lor \bar{x}_4) \]
- By UL: \[ \psi[\{x_1, x_2\}] = \exists x_3, x_4 \ \forall y_5. (x_3 \lor y_5) \land (x_4 \lor \bar{y}_5) \land (\bar{x}_3 \lor \bar{x}_4). \]
- By UR: \[ \psi[\{x_1, x_2\}] = \exists x_3, x_4. (x_3) \land (x_4) \land (\bar{x}_3 \lor \bar{x}_4) \]
Example (Clause Learning)

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

- Make decision \( A = \{x_1\} \):
  \[ \psi[\{x_1\}] = \exists x_3, x_4 \forall y_5 \exists x_2. (x_2) \land (x_3 \lor y_5 \lor \bar{x}_2) \land (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \land (\bar{x}_3 \lor \bar{x}_4) \]
- By UL: \( \psi[\{x_1, x_2\}] = \exists x_3, x_4 \forall y_5. (x_3 \lor y_5) \land (x_4 \lor \bar{y}_5) \land (\bar{x}_3 \lor \bar{x}_4) \).
- By UR: \( \psi[\{x_1, x_2\}] = \exists x_3, x_4. (x_3) \land (x_4) \land (\bar{x}_3 \lor \bar{x}_4) \)
- By UL: \( \psi[\{x_1, x_2, x_3, x_4\}] = \bot \), clause \((\bar{x}_3 \lor \bar{x}_4)\) conflicting.
QCDCL: Basic Idea (1/3)

Example (Clause Learning)

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

- Make decision \( A = \{x_1\} \):

  \[ \psi[\{x_1\}] = \exists x_3, x_4 \forall y_5 \exists x_2. (x_2) \land (x_3 \lor y_5 \lor \bar{x}_2) \land (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \land (\bar{x}_3 \lor \bar{x}_4) \]

- By UL: \( \psi[\{x_1, x_2\}] = \exists x_3, x_4 \forall y_5. (x_3 \lor y_5) \land (x_4 \lor \bar{y}_5) \land (\bar{x}_3 \lor \bar{x}_4) \).

- By UR: \( \psi[\{x_1, x_2\}] = \exists x_3, x_4. (x_3) \land (x_4) \land (\bar{x}_3 \lor \bar{x}_4) \)

- By UL: \( \psi[\{x_1, x_2, x_3, x_4\}] = \bot \), clause \((\bar{x}_3 \lor \bar{x}_4)\) conflicting.

Implication graph \( G \):

\[ L_i : \ x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow \emptyset \]

\[ \ x_2 \rightarrow x_4 \]

\[ \ x_3 \rightarrow x_4 \]

Antecedent clauses:

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

- 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.

Antecedent clauses:

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

Florian Lonsing (TU Wien)
QCDCL: Basic Idea (3/3)—Avoiding Tautologies

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 \( G \):

\[
L_i : \quad \begin{array}{c}
& x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow \emptyset \\
& \downarrow \quad \downarrow \\
& x_4
\end{array}
\]

- 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:

\[
\begin{align*}
\text{ante}(x_2) : & \quad (\overline{x}_1 \lor x_2) \\
\text{ante}(x_3) : & \quad (x_3 \lor y_5 \lor \overline{x}_2) \\
\text{ante}(x_4) : & \quad (x_4 \lor \overline{y}_5 \lor \overline{x}_2) \\
\text{ante}(\emptyset) : & \quad (\overline{x}_3 \lor \overline{x}_4)
\end{align*}
\]

Florian Lonsing (TU Wien)
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.
- 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].
Example (continued)

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

\[ (\bar{x}_3 \lor \bar{x}_4) \land (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \land (\bar{x}_3 \lor \bar{x}_4) \land (\bar{y}_5 \lor y_5 \lor \bar{x}_2) \]

**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.
Example (Clause Learning, continued)

Prefix: \( \exists x_1, x_3, x_4 \land y_5 \exists x_2 \)

Assignment \( A = \{ x_1, x_2, x_3, x_4 \} \)

Implication graph \( G \):

\[
L_i : x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow \emptyset \quad x_4
\]

- 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: \( x_3 < y_5 \)
- To derive \( C_L := (\neg x_1) \), resolve at most once on a variable.

Antecedent clauses:

\[
\begin{align*}
\text{ante}(x_2) & : (\bar{x}_1 \lor x_2) \\
\text{ante}(x_3) & : (x_3 \lor y_5 \lor \bar{x}_2) \\
\text{ante}(x_4) & : (x_4 \lor \bar{y}_5 \lor \bar{x}_2) \\
\text{ante}(\emptyset) & : (\bar{x}_3 \lor \bar{x}_4)
\end{align*}
\]
So far, we have focused on unsatisfiable QBFs.

- Clause learning: generation of QRES proofs of unsatisfiability.
Abstract Workflow: Adding Cube Learning

- **Cube learning**: solving satisfiable QBFs, similar to clause learning.
- **Cube**: conjunction of literals.
- **QCDCL**: clause and cube learning, driven by implication graphs.
- Derivation of cubes from a given PCNF $\psi$: variant of QRES.
- Termination and backtracking controlled by learned clause/cube $C_L$. 

---

Flowchart:

1. **PCNF $\psi$**
   - $A = \emptyset$
   - $A' \subset A$, $A := A'$

2. **QBCP**
   - $\psi' = \bot$ or $\psi' = \top$?

3. **Conflict/Solution Detection**
   - YES: $C_L \neq \emptyset$
   - NO: $C_L = \emptyset$

4. **Clause/Cube Learning**
   - $A := A \cup \{l\}$

5. **Decision Making**
   - UNSAT/SAT
Cube Learning: Variant of QRES (1/2)

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

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

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

$\underline{C} \quad \text{(cu-init)}$

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 $\phi$, i.e. $C \Rightarrow \phi$.
- Model generation: a new axiom added to QRES.
- QRES for cubes: Q-resolution and existential reduction on cubes.
- PCNF $\psi$ is satisfiable iff the empty cube can be derived from $\psi$. 

Florian Lonsing (TU Wien)
Cube Learning: Variant of QRES (2/2)

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

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

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

**Example**

\[
\psi = \exists x \forall u \exists y. (\bar{x} \lor u \lor \bar{y}) \land (\bar{x} \lor \bar{u} \lor y) \land (x \lor u \lor y) \land (x \lor \bar{u} \lor \bar{y})
\]

\[
(\bar{x} \land u \land \bar{y}) \quad (\bar{x} \land \bar{u} \land y)
\]

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

Cube Learning: Variant of QRES (2/2)

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

Let $C$ be a cube.

$$C \cup \{l\} \quad \text{for all } x \in \hat{Q}: \{x, \bar{x}\} \not\subseteq (C \cup \{l\}), \quad q(l) = \exists,$$

and $l' < l$ for all $l' \in C$ with $q(l') = \forall$.

\[\text{Example}\]

$$\psi = \exists x \forall u \exists y. (\bar{x} \lor u \lor y) \land (\bar{x} \lor \bar{u} \lor y) \land (x \lor u \lor y) \land (x \lor \bar{u} \lor \bar{y})$$

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

- By existential reduction: reduce trailing $\bar{y}$
  from $(\bar{x} \land u \land \bar{y})$, $y$ from $(\bar{x} \land \bar{u} \land y)$.
Cube Learning: Variant of QRES (2/2)

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

Let $C_1, C_2$ be cubes.

\[
\frac{C_1 \cup \{p\} \quad C_2 \cup \{\bar{p}\}}{C_1 \cup C_2}
\]

for all $x \in \hat{Q}$: $\{x, \bar{x}\} \not\subseteq (C_1 \cup C_2)$, $
\bar{p} \not\in C_1$, $p \not\in C_2$, and $q(p) = \forall$

$(cu$-res)

Example

\[
\psi = \exists x \forall u \exists y. (\bar{x} \lor u \lor \bar{y}) \land (\bar{x} \lor \bar{u} \lor y) \land (x \lor u \lor y) \land (x \lor \bar{u} \lor \bar{y})
\]

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

- By existential reduction: reduce trailing $\bar{y}$ from $(\bar{x} \land u \land \bar{y})$, $y$ from $(\bar{x} \land \bar{u} \land y)$.

- Resolve $(\bar{x} \land \bar{u})$ and $(\bar{x} \land u)$ on universal $u$.

- Reduce $(\bar{x})$ to derive $\emptyset$. 

Florian Lonsing (TU Wien)
Abstract Workflow: Final QCDCL View

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

- Empty clause (cube) $C_L = \emptyset$: formula proved UNSAT (SAT).
- Q-resolution proofs of (un)satisfiability by QRES.
Conflict detected: select clauses for Q-resolution.

Definition (Clause Axiom of QRES)

Given a PCNF $\psi = \hat{Q}.\phi$, $C \in \phi$ is a clause.
Solution detected: select cubes for Q-resolution.

**Definition (Cube Axiom of QRES)**

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

Clause and Cube Learning:
- PCNF $\psi := \hat{Q}.\phi$ with quantifier prefix $\hat{Q}$ and CNF $\phi$.
- CNFs of learned clauses $\phi_{CL}$ and DNF of cubes $\phi_{CU}$.
- Properties: $\hat{Q}.\phi \equiv_{sat} \hat{Q}.(\phi \land \phi_{CL})$ and $\hat{Q}.\phi \equiv_{sat} \hat{Q}.(\phi \lor \phi_{CU})$.

Interplay Between Clauses and Cubes:
- QBCP applied to $\phi$, $\phi_{CL}$, and $\phi_{CU}$.
- Assignments by unit clauses can trigger unit cubes and vice versa.
- Antecedent clauses and \textit{antecedent cubes} are recorded as usual.

Applying the Q-Resolution Calculus:
- Similar to clause learning, cube rules are driven by implication graph.
- In a derivation, applications of clause and cube rules are never mixed.
Search Space Exploration in QCDCL:

- No explicit flipping of variables in decision making.
- Fundamental difference to traditional backtracking algorithms.
- Backjumping: asserting clauses (cubes) become unit by QBCP.
- Asserting clauses (cubes) cause flipping of variables.
Asserting Learned Clauses and Cubes

Asserting Criteria Applied During Learning:
- Start at empty clause $\emptyset$ or the cube derived by model-gen. at level $k$.
- Let $C$ be the current clause/cube derived by $QRES$.
- $C$ asserting if $C$ becomes unit in QBCP at some level $j < k$.
- If $C$ asserting, then stop derivation, learn $C$, and backjump to level $j$.
- Otherwise, continue applying $QRES$ rules.
Example

\[ \exists z_1, z_2 \forall u \exists y. (u \lor \bar{y}) \land (\bar{u} \lor y) \land (z_1 \lor u \lor \bar{y}) \land (z_2 \lor \bar{u} \lor y) \land (\bar{z}_1 \lor \bar{u} \lor \bar{y}) \land (\bar{z}_2 \lor u \lor y) \]

- Level 0 is empty, no unit clauses present.
- Levels 1, 2: decisions \( z_1 \) and \( z_2 \).
- Level 3: decision \( u \), implies \( y \) by QBCP,
  \( \text{ante}(y) := (\bar{u} \lor y) \).
- Conflict: \( \text{ante}(\emptyset) := (\bar{z}_1 \lor \bar{u} \lor \bar{y}) \).

\[
\begin{align*}
(\bar{z}_1 \lor \bar{u} \lor y) & \quad (\bar{u} \lor y) \\
(\bar{z}_1 \lor u) & \\
(\bar{z}_1) 
\end{align*}
\]

- Learn clause \( C_{L,1} := (\bar{z}_1) \), asserting at \( L_0 \).
Example

$\exists z_1, z_2 \forall u \exists y. (u \lor \overline{y}) \land (\overline{u} \lor y) \land (z_1 \lor u \lor \overline{y}) \land (z_2 \lor \overline{u} \lor y) \land (\overline{z}_1 \lor \overline{u} \lor y) \land (\overline{z}_2 \lor u \lor y)$

- Backjump to $L_0$, $C_{L,1} = (\overline{z}_1)$ unit.
- Level 1: decision $\overline{z}_2$.
- Level 2: decision $\overline{u}$, implies $\overline{y}$ by QBCP.
- All clauses satisfied.

Cube learning: model generation, existential reduction.

Learn cube $C_{L,2} := (\overline{z}_1 \land \overline{z}_2 \land \overline{u})$, asserting at $L_1$.
Example

\[ \exists z_1, z_2 \forall u \exists y. (u \lor \bar{y}) \land (\bar{u} \lor y) \land (z_1 \lor u \lor \bar{y}) \land (z_2 \lor \bar{u} \lor y) \land (\bar{z}_1 \lor u \lor \bar{y}) \land (\bar{z}_2 \lor u \lor y) \]

- Backjump to \( L_1 \), \( C_{L,2} := (\bar{z}_1 \land \bar{z}_2 \land \bar{u}) \) unit.
- Level 1: \( ante(u) := C_{L,2} \), implies \((y)\).
- All clauses satisfied.

\[
\begin{align*}
L_0 : & \quad \bar{z}_1 \\
L_1 : & \quad \bar{z}_2 \rightarrow u \rightarrow y
\end{align*}
\]

- Cube learning: derive empty cube, proving satisfiability.
QCDCL Properties (by Construction):

- Implication graph, i.e., assignment order, guides QRES rules.
- Graph may contain assignments from unit clauses and cubes.
- At conflict: only clauses are derived, but never cubes.
- At solution: only cubes are derived, but never clauses.
- Empty clause (cube) potentially derived at any level (termination).
Cube Learning Worst Case: [RBM97, Let02]

- $\psi = \forall u_1 \exists x_1 \ldots \forall u_n \exists x_n. \land_{i=1}^{n}[(u_i \lor \bar{x}_i) \land (\bar{u}_i \lor x_i)]$
- Easy satisfiable formula: as the value of $x_i$, always choose $f(u_i) := u_i$.
- However: all cube resolution proofs are exponential (worst case DNF).
Typical QBF Workflow: Generating Proofs and Certificates

Solver Correctness: How to verify the result?
Typical QBF Workflow: Generating Proofs and Certificates

Problems

Encodings

Preprocessing

Solving

Proofs and Certificates

QBF model $\leadsto$ problem solution

(Counter-)Models: How to obtain solution to original problem?
Models of Satisfiable QBFs

Definition (Skolem Function)
Let $\psi$ be a PCNF, $y$ a 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 $y$: $D^\psi(y) = \{x_1, \ldots, x_k\}$.
- Function $f_y$ depends on all universal variables smaller than $y$.

Definition (Skolem Function Model)
A PCNF $\psi$ 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.
Models of Satisfiable QBFs

Example (Skolem Function Model)

\[ \psi = \exists x \forall u \exists y. (\bar{x} \lor u \lor \bar{y}) \land (\bar{x} \lor \bar{u} \lor y) \land (x \lor u \lor y) \land (x \lor \bar{u} \lor \bar{y}) \]

- Skolem function \( f_x = \bot \) of \( x \) with \( D_\psi(x) = \emptyset \).
- Skolem function \( f_y(u) = \bar{u} \) of \( y \) with \( D_\psi(y) = \{u\} \).
- \( \psi[x/f_x, y/f_y(u)] = \forall u. (\bot \lor u \lor \bar{u}) \land (\bot \lor \bar{u} \lor u) \)
- Satisfiable: \( \psi[x/f_x, y/f_y(u)] = \top \)

Checking Skolem Function Models:

- Observe: \( \psi[x/f_x, y/f_y(u)] \) contains only \( \forall \)-variables.
- Use a SAT solver to check whether \( \neg(\psi[x/f_x, y/f_y(u)]) \) is unsatisfiable.
Definition (Herbrand Function)

Let $\psi$ be a PCNF, $x$ a universal variable.

- Let $D^\psi(v) := \{ w \in \psi \mid q(v) \neq q(w) \text{ and } w < v \}$, $q(v) \in \{\forall, \exists\}$.
- Herbrand function $f_x(y_1, \ldots, y_k)$ of $x$: $D^\psi(x) = \{y_1, \ldots, y_k\}$.
- Function $f_x$ depends on all existential variables smaller than $x$.

Definition (Herbrand Function Countermodel)

A PCNF $\psi$ 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.
Countermodels of Unsatisfiable QBFs

Example (Herbrand Function Countermodel)

\[ \psi = \exists x \forall u \exists y. (x \lor u \lor y) \land (x \lor u \lor \bar{y}) \land (\bar{x} \lor \bar{u} \lor y) \land (\bar{x} \lor \bar{u} \lor \bar{y}) \]

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

Checking Herbrand Function Countermodels:

- Observe: \( \psi[x/f_x, y/f_y(u)] \) contains only \( \exists \)-variables.
- Use a SAT solver to check whether \( \psi[x/f_x, y/f_y(u)] \) is unsatisfiable.
Generating (Counter) Models from Proofs

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 $C'$. 

Generating (Counter)Models from Proofs
Example (Extracting Herbrand Functions [BJ11, BJ12])

\[ \psi = \exists x \forall u \exists y. (x \lor u \lor y) \land (x \lor u \lor \bar{y}) \land (\bar{x} \lor \bar{u} \lor y) \land (\bar{x} \lor \bar{u} \lor \bar{y}) \]

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

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

- If $\psi$ satisfiable: all Skolem functions are constant.
- If $\psi'$ 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 $\Sigma_2^P$ and $\Pi_2^P$.
- QDIMACS output format specification.
Typical QBF Workflow: Preprocessing

Problems → Encodings → Preprocessing → Solving → Proofs and Certificates
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 $\overline{l} \in C'$, there exists $x$ with $x \leq l$ such that $\{x, \overline{x}\} \subseteq (C \cup (C' \setminus \{\overline{l}\}))$.

- A clause $C$ is *blocked* if it contains a blocking literal.

- Removing blocked clauses preserves satisfiability.
Blocked Clause Elimination (QBCE) (1/2)

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

\[
\begin{align*}
(C'_1 \cup \ldots \cup C'_n) &= (x_1 \lor \ldots \lor x_i \lor \ldots \lor x_n \lor \ldots) \\
\{x_i, \overline{x_i}\} &\subseteq (C \cup (C'_i \setminus \{\overline{l}\})) \quad x_i \leq l
\end{align*}
\]
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 $\overline{l} \in C'$, there exists $x$ with $x \leq l$ such that $\{x, \overline{x}\} \subseteq (C \cup (C' \setminus \{\overline{l}\}))$.
- A clause $C$ is *blocked* if it contains a blocking literal.
- Removing blocked clauses preserves satisfiability.

\[
\begin{align*}
\left(\ldots \lor \overline{x}_1 \lor \ldots \lor \overline{l} \ldots\right) & \quad \ldots \\
\left(\ldots \lor \overline{x}_i \lor \ldots \lor \overline{l} \ldots\right) & \quad \ldots \\
\left(\ldots \lor \overline{x}_n \lor \ldots \lor \overline{l} \ldots\right)
\end{align*}
\]

\[
\{x_n, \overline{x}_n\} \subseteq C \cup (C'_n \setminus \{\overline{l}\}) \quad x_n \leq l
\]

$C' = (x_1 \lor \ldots \lor x_i \lor \ldots \lor x_n \lor \ldots \lor l \lor \ldots)$
Important Facts:

- Blocking literal \( y \): existentially quantified.

Example

\[
\psi := \exists y \forall x \exists z. (y \lor \lnot x \lor z) \land (\lnot y \lor x \lor z) \land (y) \land (\lnot z)
\]

- \( \psi \) is unsatisfiable.
- Universal \( x \) cannot be a blocking literal.
- Otherwise, first two clauses would erroneously be blocked.
- Unsoundness: \( \psi \) becomes satisfiable.
Important Facts:
- Blocking literal $l$: existentially quantified.
- Tautology-producing variable $x$: $\leq l$ in prefix ordering.

Example
\[ \psi := \exists y \forall x. (y \lor \bar{x}) \land (\bar{y} \lor x). \]
- $\psi$ is unsatisfiable.
- Prefix ordering matters.
- Literals of $y$ are not blocking literals since $y \leq x$.
- Erroneous removal of any clause makes formula satisfiable.
Important Facts:

- Blocking literal $l$: existentially quantified.
- Tautology-producing variable $x$: $\leq l$ in prefix ordering.
- Check all potential resolution candidates on $l$.
  - Pure $\exists$-literals: vacuously blocking.

Example

$$\psi = \exists y \forall x \exists z. (\bar{y} \lor z) \land (\bar{y} \lor \bar{z}) \land (\bar{x} \lor z) \land (x \lor \bar{z}).$$

- $\exists$-literal $\bar{y}$ is pure.
- No resolution candidates on clauses containing $y$.
- Condition of blocking literal is vacuously satisfied.
- Clauses containing $\bar{y}$ can be removed.
Expansion (1/4)

\[ \psi_0 \rightarrow \psi_1 \rightarrow \psi_2 \rightarrow \ldots \rightarrow \psi_n = \bot / \top \]

- Successively eliminate variables from a given PCNF \( \psi_0 \).
- Elimination produces satisfiability-equivalent PCNFs \( \psi_i \equiv_{sat} \psi_{i+1} \).
- Worst case exponential space procedure.
- Redundancy elimination on \( \psi_i \) (depending on formula representation).
- Stop if \( \psi_i \) reduces to truth constant \( \top \) or \( \bot \).
- Call a SAT solver if \( \psi_i \) contains only \( \exists \)-variables.
- Lazy expansion by counter example guided abstraction refinement (CEGAR) [CGJ+03, JM15b, JKMSC16, RT15].
Expansion (2/4)

Example

\[ \psi = \exists x \forall u \exists y. (\bar{x} \lor y) \land (x \lor \bar{y}) \land (\bar{u} \lor y) \land (u \lor \bar{y}) \]

- Eliminate rightmost \( y \):
  \[ \psi = \exists x \forall u. \left[ (\bar{x}) \land (\bar{u}) \right] \lor \left[ (x) \land (u) \right] \]
  \( y \) replaced by \( \bot \) \( y \) replaced by \( \top \)

- Convert back to PCNF (distributivity):
  \[ \psi = \exists x \forall u. (\bar{x} \lor x) \land (\bar{x} \lor u) \land (x \lor \bar{u}) \land (u \lor \bar{u}) \]

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

- Eliminate rightmost variables by Shannon expansion [Sha49].
- Replace \( \hat{Q}\exists x.\phi \) by \( \hat{Q}.(\phi[x/\bot] \lor \phi[x/\top]) \).
- Based on CNF, NNF, and-inverter graphs [AB02, LB08, PS09].
Example (continued)

- Eliminate rightmost $y$:
  \[
  \psi = \exists x \forall u. \left[ (\bar{x}) \land (\bar{u}) \right] \lor \left[ (x) \land (u) \right]
  \]
  $y$ replaced by $\bot$ $y$ replaced by $\top$

- Convert to back PCNF:
  \[
  \psi = \exists x \forall u. (\bar{x} \lor x) \land (\bar{x} \lor u) \land (x \lor \bar{u}) \land (u \lor \bar{u})
  \]

- Simplify and reduce $u$: $\psi = \exists x. (\bar{x}) \land (x)$

Special Case – $\psi$ in PCNF:

- Eliminate leftmost $\forall$-variables by universal reduction.
- Implemented in early expansion-based solvers, cf. [AB02, Bie04].
Example (continued)

\[ \psi = \exists x \forall u \exists y. (\overline{x} \lor y) \land (x \lor \overline{y}) \land (\overline{u} \lor y) \land (u \lor \overline{y}) \]

- Expand \( u \): copy CNF and replace \( y \) by fresh \( y_d \) in copy of CNF.
- \[ \psi' = \exists x, y, y_d. (\overline{x} \lor y) \land (x \lor \overline{y}) \land (\overline{y}) \land (\overline{x} \lor y_d) \land (x \lor \overline{y}_d) \land (y_d) \]
  - \( u \) replaced by \( \bot \)
  - \( u \) replaced by \( \top \), \( y \) replaced by \( y_d \)
- Obtain \( \overline{x} \) from \( (\overline{x} \lor y) \) and \( (\overline{y}) \), \( x \) from \( (x \lor \overline{y}_d) \) and \( (y_d) \).

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

- Eliminate all universal variables by Shannon expansion.
- Finally, apply SAT solving.
- If \( x \) innermost: replace \( \hat{\forall} x.\phi \) by \( \hat{\forall}.(\phi[x/\bot] \land \phi[x/\top]) \).
- Otherwise, duplicate existential variables inner to \( x \) [Bie04, BK07].
Let $\psi := \exists X \forall Y. \phi$ be a one-alternation QBF, $\phi$ a non-CNF formula.

$\psi$ is satisfiable iff $\psi' := \exists X. (\bigwedge_{y \in B | Y} \phi[Y/y])$ is satisfiable.

Full expansion $\psi'$ of $\forall Y$ by set $B | Y$ of all possible assignments $y$ of $Y$.

Idea: consider a partial expansion of $\forall Y$ as an abstraction of $\psi'$.
Lazy Expansion by CEGAR

\[ \psi := \exists X \forall Y. \phi \]

\[ \text{Abs}(\psi) := \top \]

\[ \psi' := \exists X. (\wedge_{y \in B \mid Y} \phi[Y/y]) \]

\[ \text{Abs}(\psi) := \exists X. (\wedge_{y \in U} \phi[Y/y]) \]

- Subset \( U \subseteq B \mid Y \) of set \( B \mid Y \) of all possible assignments \( y \) of \( Y \).
- Partial expansion: given \( U \), define \( \text{Abs}(\psi) := \exists X. (\wedge_{y \in U} \phi[Y/y]) \).
- Abstraction \( \text{Abs}(\psi) \): if \( \text{Abs}(\psi) \) unsatisfiable, then also \( \psi \) unsatisfiable.
- Initially, set \( U := \emptyset \) and \( \text{Abs}(\psi) := \top \).
Lazy Expansion by CEGAR

\[ \psi := \exists X \forall Y. \phi \]

\[ Abs(\psi) := \top \]

\[ \psi' := \exists X. (\bigwedge_{y \in B \mid Y} \phi[Y/y]) \]

\[ Abs(\psi) := \exists X. (\bigwedge_{y \in U} \phi[Y/y]) \]

Check satisfiability of \( Abs(\psi) \) using a SAT solver.

If \( Abs(\psi) \) unsatisfiable: also \( \psi \) unsatisfiable, terminate.

If \( Abs(\psi) \) satisfiable: let \( x \in B^{|X|} \) be a model of \( Abs(\psi) \).

\( x \in B^{|X|} \): candidate solution of full exp. \( \psi' := \exists X. (\bigwedge_{y \in B \mid Y} \phi[Y/y]) \).
Lazy Expansion by CEGAR

\[ \psi := \exists X \forall Y. \phi \]

\[ \text{Abs}(\psi) := \top \]

\[ \psi' := \exists X. (\bigwedge_{y \in B|Y|} \phi[Y/y]) \]
\[ \text{Abs}(\psi) := \exists X. (\bigwedge_{y \in U} \phi[Y/y]) \]

- If \( x \) is also a model of the full expansion \( \psi' \), then \( \psi \) is satisfiable.
- \( x \) is a model of full expansion \( \psi' \) iff \( \forall Y. \phi[X/x] \) is satisfiable.
- \( \forall Y. \phi[X/x] \) is satisfiable iff \( \exists Y. \neg \phi[X/x] \) is unsatisfiable.
- Check satisfiability of \( \exists Y. \neg \phi[X/x] \) using a SAT solver.
Lazy Expansion by CEGAR

\( \psi := \exists X \forall Y. \phi \)

Abs(\( \psi \)) := \top

\( \psi \) UNSAT

Find Candidate Solution

\( \psi \) SAT

Check Candidate Solution

\( \psi' := \exists X. (\land_{y \in B | Y|} \phi[Y/y]) \)

Abs(\( \psi \)) := \exists X. (\land_{y \in U} \phi[Y/y])

Refine Abs(\( \psi \))

Find Counterexample

If \( \exists Y. \neg \phi[X/x] \) unsatisfiable: \( \psi \) is satisfiable, return \( x \) and terminate.

If \( \exists Y. \neg \phi[X/x] \) satisfiable: let \( y \in B^{|Y|} \) be a model of \( \exists Y. \neg \phi[X/x] \).

Note: \( y \) is an assignment to \( \forall \)-variables in \( \psi \).

\( y \) is a counterexample to candidate solution \( x \) of full expansion \( \psi' \).
Lazy Expansion by CEGAR

\[ \psi := \exists X \forall Y. \phi \]

Abs(\( \psi \)) := ⊤

\[ \psi :\text{UNSAT} \]

Find Candidate Solution

\[ \psi :\text{SAT} \]

Check Candidate Solution

psib := \exists X. (\bigwedge_{y \in B \mid Y} \phi[Y/y])

Abs(\( \psi \)) := \exists X. (\bigwedge_{y \in U} \phi[Y/y])

Refine Abs(\( \psi \))

Find Counterexample

\[ \text{Refine abstraction Abs}(\psi) \text{ by counterexample } y. \]

\[ \text{Let } U := U \cup \{y\} \text{ and } \text{Abs}(\psi) := \exists X. (\bigwedge_{y \in U} \phi[Y/y]). \]

\[ \text{Adding } y \text{ to } \text{Abs}(\psi) \text{ prevents repetition of candidate solution } x. \]

\[ \text{E.g. for 2QBF [RTM04, BJS}^{+}16\], \text{ RAReQS (recursive) [JKMSC16].} \]
Experiments (1/6)

Benchmark Set from QBFEVAL’16:
- 825 prenex CNF instances, 1800 seconds, 7 GB memory limits.

QBF Solvers:
- Top ranked solvers from QBFEVAL’16.
- Five different solving paradigms.
- Some solvers are based on orthogonal proof systems.
- Theory: exponential gap in solving capabilities.
Alternation Bias in QBFEVAL’16 Benchmarks: cf. [LE17]

- 56% of the benchmarks have no more than two quantifier alternations.
- Theory: numbers of alternations $\approx$ levels in polynomial hierarchy.
- Focus: 402 instances not solved by preprocessing using Bloqger [BLS11].
- Analysis wrt. instances having few/many alternations.
Experiments (2/6): 402 Filtered Instances

<table>
<thead>
<tr>
<th>Solved</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>GhostQ</td>
<td>176</td>
</tr>
<tr>
<td>AIGSolve</td>
<td>138</td>
</tr>
<tr>
<td>QSTS</td>
<td>136</td>
</tr>
<tr>
<td>RAReQS</td>
<td>76</td>
</tr>
<tr>
<td>DQ</td>
<td>69</td>
</tr>
<tr>
<td>QESTO</td>
<td>66</td>
</tr>
<tr>
<td>DQ-n</td>
<td>52</td>
</tr>
<tr>
<td>CAQE</td>
<td>43</td>
</tr>
</tbody>
</table>

- 261 instances (65%), \( \leq 2 \) alternations, filtered but *not* preprocessed.
141 instances (35%), ≥ 3 alternations, filtered but not preprocessed.

QCDCL, e.g. DepQBF (DQ), performs better on many alternations.
Experiments (3/6): 402 Filtered Instances

<table>
<thead>
<tr>
<th>Solved</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RAReQS</td>
<td>157</td>
</tr>
<tr>
<td>QESTO</td>
<td>138</td>
</tr>
<tr>
<td>QSTS</td>
<td>136</td>
</tr>
<tr>
<td>CAQE</td>
<td>118</td>
</tr>
<tr>
<td>GhostQ</td>
<td>111</td>
</tr>
<tr>
<td>DQ</td>
<td>107</td>
</tr>
<tr>
<td>DQ-n</td>
<td>105</td>
</tr>
<tr>
<td>AIGSolve</td>
<td>102</td>
</tr>
</tbody>
</table>

- 270 instances (67%), ≤ 2 alternations, filtered and preprocessed.
Experiments (3/6): 402 Filtered Instances

<table>
<thead>
<tr>
<th>Solved</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ</td>
<td>81</td>
</tr>
<tr>
<td>QSTS</td>
<td>75</td>
</tr>
<tr>
<td>DQ-n</td>
<td>75</td>
</tr>
<tr>
<td>QESTO</td>
<td>69</td>
</tr>
<tr>
<td>CAQE</td>
<td>64</td>
</tr>
<tr>
<td>RAReQS</td>
<td>62</td>
</tr>
<tr>
<td>AIGSolve</td>
<td>51</td>
</tr>
<tr>
<td>GhostQ</td>
<td>46</td>
</tr>
</tbody>
</table>

- 132 instances (33%), ≥ 3 alternations, filtered and preprocessed.
- QCDCL, e.g. DepQBF (DQ), performs better on many alternations.

Florian Lonsing (TU Wien)
Table: Solved instances ($S$), solved unsatisfiable ($\bot$) and satisfiable ones ($\top$), and total wall clock time including time outs on 437 filtered instances from QBFEVAL’17 without (a) and with preprocessing by Bloqcer (b).

<table>
<thead>
<tr>
<th>Solver</th>
<th>$S$</th>
<th>$\bot$</th>
<th>$\top$</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIGSolve</td>
<td>177</td>
<td>121</td>
<td>56</td>
<td>489K</td>
</tr>
<tr>
<td>Rev-Qfun</td>
<td>174</td>
<td>106</td>
<td>68</td>
<td>497K</td>
</tr>
<tr>
<td>GhostQ</td>
<td>145</td>
<td>79</td>
<td>66</td>
<td>547K</td>
</tr>
<tr>
<td>RAReQS</td>
<td>126</td>
<td>94</td>
<td>32</td>
<td>577K</td>
</tr>
<tr>
<td>CAQE</td>
<td>126</td>
<td>87</td>
<td>39</td>
<td>578K</td>
</tr>
<tr>
<td>Heretic</td>
<td>122</td>
<td>95</td>
<td>27</td>
<td>580K</td>
</tr>
<tr>
<td>DepQBF-opt</td>
<td>115</td>
<td>78</td>
<td>37</td>
<td>603K</td>
</tr>
<tr>
<td>Ijtihad</td>
<td>110</td>
<td>88</td>
<td>22</td>
<td>599K</td>
</tr>
<tr>
<td>QSTS-d</td>
<td>103</td>
<td>75</td>
<td>28</td>
<td>618K</td>
</tr>
<tr>
<td>Qute-random</td>
<td>77</td>
<td>47</td>
<td>30</td>
<td>658K</td>
</tr>
<tr>
<td>QESTO</td>
<td>76</td>
<td>56</td>
<td>20</td>
<td>661K</td>
</tr>
<tr>
<td>DynQBF</td>
<td>47</td>
<td>27</td>
<td>20</td>
<td>714K</td>
</tr>
</tbody>
</table>

(a) Not preprocessed.

<table>
<thead>
<tr>
<th>Solver</th>
<th>$S$</th>
<th>$\bot$</th>
<th>$\top$</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAReQS</td>
<td>175</td>
<td>127</td>
<td>48</td>
<td>499K</td>
</tr>
<tr>
<td>CAQE</td>
<td>169</td>
<td>114</td>
<td>55</td>
<td>514K</td>
</tr>
<tr>
<td>Heretic</td>
<td>164</td>
<td>119</td>
<td>45</td>
<td>513K</td>
</tr>
<tr>
<td>AIGSolve</td>
<td>138</td>
<td>98</td>
<td>40</td>
<td>555K</td>
</tr>
<tr>
<td>Ijtihad</td>
<td>136</td>
<td>103</td>
<td>33</td>
<td>555K</td>
</tr>
<tr>
<td>Rev-Qfun</td>
<td>135</td>
<td>92</td>
<td>43</td>
<td>563K</td>
</tr>
<tr>
<td>QSTS-d</td>
<td>127</td>
<td>98</td>
<td>29</td>
<td>576K</td>
</tr>
<tr>
<td>QESTO</td>
<td>115</td>
<td>84</td>
<td>31</td>
<td>601K</td>
</tr>
<tr>
<td>DepQBF-opt</td>
<td>102</td>
<td>64</td>
<td>38</td>
<td>624K</td>
</tr>
<tr>
<td>GhostQ</td>
<td>82</td>
<td>47</td>
<td>35</td>
<td>661K</td>
</tr>
<tr>
<td>Qute-random</td>
<td>73</td>
<td>56</td>
<td>17</td>
<td>672K</td>
</tr>
<tr>
<td>DynQBF</td>
<td>65</td>
<td>37</td>
<td>28</td>
<td>684K</td>
</tr>
</tbody>
</table>

(b) Preprocessed by Bloqcer.
Table: Instances solved in 437 filtered instances not preprocessed by Bloqger with respect to classes by number of quantifier blocks (#q) and number of formulas in each class (#f).

<table>
<thead>
<tr>
<th>#q</th>
<th>#f</th>
<th>AIGSolve</th>
<th>Rev-Qfun</th>
<th>GhostQ</th>
<th>RAReQS</th>
<th>CAQE</th>
<th>Heretic</th>
<th>DepQBF-opt</th>
<th>Ijtihad</th>
<th>QSTS-d</th>
<th>Qute-random</th>
<th>QESTO</th>
<th>DynQBF</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>63</td>
<td>33</td>
<td>17</td>
<td>32</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>4</td>
<td>18</td>
</tr>
<tr>
<td>3</td>
<td>215</td>
<td>83</td>
<td>101</td>
<td>89</td>
<td>62</td>
<td>56</td>
<td>50</td>
<td>47</td>
<td>49</td>
<td>43</td>
<td>36</td>
<td>35</td>
<td>19</td>
</tr>
<tr>
<td>4–6</td>
<td>36</td>
<td>27</td>
<td>16</td>
<td>3</td>
<td>16</td>
<td>20</td>
<td>16</td>
<td>6</td>
<td>16</td>
<td>14</td>
<td>6</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>7–9</td>
<td>27</td>
<td>19</td>
<td>9</td>
<td>1</td>
<td>17</td>
<td>5</td>
<td>18</td>
<td>8</td>
<td>16</td>
<td>7</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>10–15</td>
<td>15</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>16–20</td>
<td>21</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>7</td>
<td>10</td>
<td>4</td>
<td>8</td>
<td>5</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>21–</td>
<td>60</td>
<td>13</td>
<td>25</td>
<td>17</td>
<td>23</td>
<td>30</td>
<td>25</td>
<td>28</td>
<td>23</td>
<td>23</td>
<td>22</td>
<td>23</td>
<td>5</td>
</tr>
<tr>
<td>2–3</td>
<td>278</td>
<td>116</td>
<td>118</td>
<td>121</td>
<td>64</td>
<td>61</td>
<td>52</td>
<td>53</td>
<td>51</td>
<td>51</td>
<td>38</td>
<td>39</td>
<td>37</td>
</tr>
<tr>
<td>4–</td>
<td>159</td>
<td>61</td>
<td>56</td>
<td>24</td>
<td>62</td>
<td>65</td>
<td>70</td>
<td>62</td>
<td>59</td>
<td>52</td>
<td>39</td>
<td>37</td>
<td>10</td>
</tr>
</tbody>
</table>
### Table: Instances solved in 437 filtered instances preprocessed by Bloqquer with respect to classes by number of quantifier blocks (#q) and number of formulas in each class (#f).

<table>
<thead>
<tr>
<th>#q</th>
<th>#f</th>
<th>RAReQS</th>
<th>CAQE</th>
<th>Heretic</th>
<th>AIGSolve</th>
<th>Ijtihad</th>
<th>Rev-Qfun</th>
<th>QSTS-d</th>
<th>QESTO</th>
<th>DepQBF-opt</th>
<th>GhostQ</th>
<th>Qute-random</th>
<th>DynQBF</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>65</td>
<td>16</td>
<td>15</td>
<td>13</td>
<td>12</td>
<td>10</td>
<td>6</td>
<td>11</td>
<td>14</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>3</td>
<td>218</td>
<td>80</td>
<td>81</td>
<td>65</td>
<td>65</td>
<td>59</td>
<td>65</td>
<td>46</td>
<td>50</td>
<td>34</td>
<td>53</td>
<td>23</td>
<td>18</td>
</tr>
<tr>
<td>4–6</td>
<td>32</td>
<td>18</td>
<td>20</td>
<td>17</td>
<td>24</td>
<td>17</td>
<td>19</td>
<td>17</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>7–9</td>
<td>27</td>
<td>19</td>
<td>6</td>
<td>21</td>
<td>19</td>
<td>18</td>
<td>10</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>10–15</td>
<td>25</td>
<td>13</td>
<td>9</td>
<td>14</td>
<td>3</td>
<td>8</td>
<td>10</td>
<td>8</td>
<td>7</td>
<td>15</td>
<td>6</td>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td>16–20</td>
<td>28</td>
<td>12</td>
<td>16</td>
<td>17</td>
<td>4</td>
<td>9</td>
<td>12</td>
<td>16</td>
<td>11</td>
<td>15</td>
<td>3</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>21–</td>
<td>42</td>
<td>17</td>
<td>22</td>
<td>17</td>
<td>11</td>
<td>15</td>
<td>13</td>
<td>21</td>
<td>20</td>
<td>17</td>
<td>12</td>
<td>15</td>
<td>6</td>
</tr>
<tr>
<td>2–3</td>
<td>283</td>
<td>96</td>
<td>96</td>
<td>78</td>
<td>77</td>
<td>69</td>
<td>71</td>
<td>57</td>
<td>64</td>
<td>41</td>
<td>56</td>
<td>27</td>
<td>42</td>
</tr>
<tr>
<td>4–</td>
<td>154</td>
<td>79</td>
<td>73</td>
<td>86</td>
<td>61</td>
<td>67</td>
<td>64</td>
<td>70</td>
<td>51</td>
<td>61</td>
<td>26</td>
<td>46</td>
<td>23</td>
</tr>
</tbody>
</table>
Outlook and Future Work
Outlook and Future Work (1/2)

**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.
QBF is still an emerging field with plenty of applications.
Assuming that \( \text{NP} \neq \text{PSPACE} \), QBF is more difficult than SAT...
...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.
Please document and publish your tools and benchmarks!
Appendix
[Appendix] Expansion and Instantiation

Definition (\(\forall\text{Exp}+\text{RES} \ [JM13, BCJ14, JM15a]\))

- **Axiom:** \(\frac{\text{C}}{\text{C}}\) for all \(x \in \hat{Q}: \{x, \bar{x}\} \not\subseteq C\) and \(C \in \phi\)

- **Instantiation:** \(\frac{\text{C}}{\{l^A| l \in C, q(l) = \exists\}}\)

  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:** \(\frac{C_1 \cup \{p^A\} \quad C_2 \cup \{\bar{p}^A\}}{C_1 \cup C_2 \quad \text{for all } x \in \hat{Q}: \{x, \bar{x}\} \not\subseteq (C_1 \cup C_2)}\)

  - 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.
Example (continued)

\[ \psi = \exists x \forall u \exists y. (\bar{x} \lor y) \land (x \lor \bar{y}) \land (\bar{u} \lor y) \land (u \lor \bar{y}) \]

- Complete assignments: \( A = \{\bar{u}\} \) and \( A' = \{u\} \).
- Instantiate: \((\bar{x} \lor y\bar{u}) \land (x \lor \bar{y}u) \land (y^u) \land (\bar{y}\bar{u})\)
- Note: cannot resolve \((y^u)\) and \((\bar{y}\bar{u})\) due to mismatching annotations.
- Obtain \(x\) from \((x \lor \bar{y}u)\) and \((y^u)\), \(\bar{x}\) from \((\bar{x} \lor y\bar{u})\) 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].
Definition (QBF to FOL Translation [SLB12])

Mapping $\lbrack \cdot \rbrack : QBF \rightarrow FOL$ with respect to unary FOL predicate $p$:

\[
\begin{align*}
\lbrack \exists x. \phi \rbrack & = \exists x. \lbrack \phi \rbrack \\
\lbrack \phi \lor \psi \rbrack & = \lbrack \phi \rbrack \lor \lbrack \psi \rbrack \\
\lbrack x \rbrack & = p(x) \\
\lbrack \top \rbrack & = p(\text{true}) \\
\lbrack \forall x. \phi \rbrack & = \forall x. \lbrack \phi \rbrack \\
\lbrack \phi \land \psi \rbrack & = \lbrack \phi \rbrack \land \lbrack \psi \rbrack \\
\lbrack \neg \psi \rbrack & = \neg \lbrack \psi \rbrack \\
\lbrack \bot \rbrack & = p(\text{false})
\end{align*}
\]

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

Proposition ([SLB12])

The QBF $\psi$ is satisfiable iff $\lbrack \psi \rbrack \land p(\text{true}) \land \neg p(\text{false})$ is satisfiable.
[Appendix] Typical QBF Workflow

Problems → Encodings → Preprocessing → Solving → Proofs and Certificates
# 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:

```
∀z. z ∨ ∃x₁.∃x₂. (x₁ ∧ x₂ ∧ z)
g₁ = and(x₁, x₂, z)
g₂ = exists(x₁, x₂; g₁)
g₃ = or(z, g₂)
```

[Appendix] Encodings (2)

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

\[(Qx. \phi) \circ \psi \equiv Qx. (\phi \circ \psi), \psi \text{ a QBF, } Q \in \{\forall, \exists\}, \circ \in \{\land, \lor\}, x \notin \text{Var}(\psi)\]

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

- Given a prenex QBF \(\psi := \hat{Q}.\phi\), subformulas \(\psi_i\) of \(\psi\).
- \(\psi_i = (\psi_{i,l} \circ \psi_{i,r}), \circ \in \{\lor, \land, \to, \leftrightarrow, \otimes\}\).
- 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 \(\psi'\): satisfiability-equivalent to \(\psi\), size linear in \(|\psi|\).
- Safe: quantify each \(t_i\) innermost [GMN09]: \(\psi := \hat{Q} \exists t_i.\phi\).
[Appendix] 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 $\psi$, fresh existential variable $v$.
- Add definition $v \leftrightarrow (\bar{x}_i \lor \bar{x}_j)$ in CNF: $(\bar{v} \lor \bar{x}_i \lor \bar{x}_j) \land (v \lor x_i) \land (v \lor x_j)$.
- 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.
[Appendix] 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 \( \phi \) 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.
\[ \hat{Q}_W.\phi := \exists x_1, x_2, x_3 \forall y \]

\[ \text{XOR}_3(\text{XOR}_2(\text{XOR}_1(x_1, x_2), x_3), y) \]

\[
\begin{align*}
t_1 & : (\neg t_1 \lor x_1 \lor x_2) \land \\
& \quad (\neg t_1 \lor \neg x_1 \lor \neg x_2) \land \\
& \quad (t_1 \lor \neg x_1 \lor x_2) \land \\
& \quad (t_1 \lor x_1 \lor \neg x_2) \land
\end{align*}
\]

\[
\begin{align*}
t_2 & : (\neg t_2 \lor t_1 \lor x_3) \land \\
& \quad (\neg t_2 \lor \neg t_1 \lor \neg x_3) \land \\
& \quad (t_2 \lor \neg t_1 \lor x_3) \land \\
& \quad (t_2 \lor t_1 \lor \neg x_3) \land
\end{align*}
\]

\[
\begin{align*}
t_3 & : (\neg t_3 \lor t_2 \lor y) \land \\
& \quad (\neg t_3 \lor \neg t_2 \lor \neg y) \land \\
& \quad (t_3 \lor \neg t_2 \lor y) \land \\
& \quad (t_3 \lor t_2 \lor \neg y) \land
\end{align*}
\]

\[ \text{out} : (t_3) \]
\[ \hat{Q}_W \cdot \phi := \exists x_1, x_2, x_3 \forall y \exists t_1, t_2, t_3. \text{XOR}_3(\text{XOR}_2(\text{XOR}_1(x_1, x_2), x_3), y) \]

\[ t_1 := (\bar{t}_1 \lor x_1 \lor x_2) \land (\bar{t}_1 \lor \bar{x}_1 \lor \bar{x}_2) \land (t_1 \lor \bar{x}_1 \lor x_2) \land (t_1 \lor x_1 \lor \bar{x}_2) \land \]

\[ t_2 := (\bar{t}_2 \lor t_1 \lor x_3) \land (\bar{t}_2 \lor \bar{t}_1 \lor \bar{x}_3) \land (t_2 \lor \bar{t}_1 \lor x_3) \land (t_2 \lor t_1 \lor \bar{x}_3) \land \]

\[ t_3 := (\bar{t}_3 \lor t_2 \lor y) \land (\bar{t}_3 \lor \bar{t}_2 \lor \bar{y}) \land (t_3 \lor \bar{t}_2 \lor y) \land (t_3 \lor t_2 \lor \bar{y}) \land \]

\[ \text{out} := (t_3) \]
\[ \hat{Q}_S \phi := \exists x_1, x_2, x_3 \quad \forall y \quad . \quad \text{XOR}_3(\text{XOR}_2(\text{XOR}_1(x_1, x_2), x_3), y) \]

\[
\begin{align*}
\otimes t_3 \\
\otimes t_2 \quad y \\
\otimes t_1 \\
x_1 \\
x_2 \\
x_3
\end{align*}
\]

\[
\begin{align*}
t_1 & : (\bar{t}_1 \lor x_1 \lor x_2) \land \\
& \quad (\bar{t}_1 \lor \bar{x}_1 \lor \bar{x}_2) \land \\
& \quad (t_1 \lor \bar{x}_1 \lor x_2) \land \\
& \quad (t_1 \lor x_1 \lor \bar{x}_2) \land \\
t_2 & : (\bar{t}_2 \lor t_1 \lor x_3) \land \\
& \quad (\bar{t}_2 \lor \bar{t}_1 \lor \bar{x}_3) \land \\
& \quad (t_2 \lor \bar{t}_1 \lor x_3) \land \\
& \quad (t_2 \lor t_1 \lor \bar{x}_3) \land \\
t_3 & : (\bar{t}_3 \lor t_2 \lor y) \land \\
& \quad (\bar{t}_3 \lor \bar{t}_2 \lor \bar{y}) \land \\
& \quad (t_3 \lor \bar{t}_2 \lor y) \land \\
& \quad (t_3 \lor t_2 \lor \bar{y}) \land \\
\text{out} & : (t_3)
\end{align*}
\]
\[ \hat{Q}_5.\phi := \exists x_1, x_2, x_3, t_1, t_2 \forall y \exists t_3. \text{XOR}_3(\text{XOR}_2(\text{XOR}_1(x_1, x_2), x_3), y) \]

\[ t_1 : (\bar{t}_1 \lor x_1 \lor x_2) \land \\
(\bar{t}_1 \lor \bar{x}_1 \lor \bar{x}_2) \land \\
(t_1 \lor \bar{x}_1 \lor x_2) \land \\
(t_1 \lor x_1 \lor \bar{x}_2) \land \]

\[ t_2 : (\bar{t}_2 \lor t_1 \lor x_3) \land \\
(\bar{t}_2 \lor \bar{t}_1 \lor \bar{x}_3) \land \\
(t_2 \lor \bar{t}_1 \lor x_3) \land \\
(t_2 \lor t_1 \lor \bar{x}_3) \land \]

\[ t_3 : (\bar{t}_3 \lor t_2 \lor y) \land \\
(\bar{t}_3 \lor \bar{t}_2 \lor \bar{y}) \land \\
(t_3 \lor \bar{t}_2 \lor y) \land \\
(t_3 \lor t_2 \lor \bar{y}) \land \]

\[ \text{out} : (t_3) \]
Example (Clause Selection and Clausal Abstraction [JM15b, RT15])

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

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

[BB09] Hans Kleine Büning and Uwe Bubeck. 
Theory of Quantified Boolean Formulas.

Symbolic Model Checking without BDDs.

On unification of QBF resolution-based calculi.
References II

Proof Complexity of Resolution-based QBF Calculi.

Extension Variables in QBF Resolution.

[Bie04] Armin Biere.
Resolve and Expand.

Resolution Proofs and Skolem Functions in QBF Evaluation and Applications.

Unified QBF certification and its applications.


### References VI

<table>
<thead>
<tr>
<th>Reference</th>
<th>Authors</th>
<th>Title</th>
<th>Conference/Proceedings</th>
<th>Volume/Application</th>
<th>Pages</th>
<th>Publisher</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>[FR05]</td>
<td>Wolfgang Faber and Francesco Ricca.</td>
<td>Solving Hard ASP Programs Efficiently.</td>
<td>LPNMR</td>
<td>LNCS</td>
<td>3662</td>
<td>Springer</td>
<td>2005</td>
</tr>
</tbody>
</table>
References VII

[FT14] Bernd Finkbeiner and Leander Tentrup. 
Fast DQBF Refutation. 

Detecting Unrealizability of Distributed Fault-tolerant Systems. 

A Uniform Approach for Generating Proofs and Strategies for Both True and False 
QBF Formulas. 

[GGN+04] Ian P. Gent, Enrico Giunchiglia, Massimo Narizzano, Andrew G. D. Rowley, and 
Armando Tacchella. 
Watched Data Structures for QBF Solvers. 

http://www.cs.cmu.edu/~wklieber/ghostq/.
<table>
<thead>
<tr>
<th>Reference</th>
<th>Authors</th>
<th>Title</th>
<th>Details</th>
</tr>
</thead>
</table>


References X


References XI

On Propositional QBF Expansions and Q-Resolution.

Expansion-based QBF solving versus Q-resolution.

Solving QBF by Clause Selection.

On Deciding MUS Membership with QBF.

A Non-prenex, Non-clausal QBF Solver with Game-State Learning.

[Kul99] Oliver Kullmann.
On a Generalization of Extended Resolution.
[LB08] Florian Lonsing and Armin Biere. 
Nenofex: Expanding NNF for QBF Solving. 

Integrating Dependency Schemes in Search-Based QBF Solvers. 

[LE14] Florian Lonsing and Uwe Egly. 
Incremental QBF Solving. 

[LE17] Florian Lonsing and Uwe Egly. 
Evaluating QBF Solvers: Quantifier Alternations Matter. 
technical report.

Efficient clause learning for quantified boolean formulas via QBF pseudo unit propagation. 
References XIII

[Let02] Reinhold Letz.
Lemma and Model Caching in Decision Procedures for Quantified Boolean Formulas.

[Lib05] Paolo Liberatore.
Redundancy in logic I: CNF propositional formulae.

Verification of partial designs using incremental QBF solving.

Chaff: Engineering an Efficient SAT Solver.

The Equivalence Problem for Regular Expressions with Squaring Requires Exponential Space.
References XIV


References XV

CNF and DNF Considered Harmful for Computing Prime Implicants/Implicates. 

[Rin07] Jussi Rintanen.
Asymptotically Optimal Encodings of Conformant Planning in QBF.

Incremental Determinization.

CAQE: A Certifying QBF Solver.

A Comparative Study of 2QBF Algorithms.

The Complexity of Propositional Linear Temporal Logics.
[Sch78] Thomas J Schaefer.  
On the Complexity of Some Two-Person Perfect-Information Games.  

[Sha49] Claude Elwood Shannon.  
The Synthesis of Two-Terminal Switching Circuits.  

qbf2epr: A Tool for Generating EPR Formulas from QBF.  

GRASP - a new search algorithm for satisfiability.  

GRASP: A Search Algorithm for Propositional Satisfiability.  

Backdoor Sets of Quantified Boolean Formulas.  
*JAR, 42(1):77–97, 2009.*
References XVII

[Sto76] Larry J. Stockmeyer.
The Polynomial-Time Hierarchy.

[Tse68] G. S. Tseitin.
On the Complexity of Derivation in Propositional Calculus.

Variable Independence and Resolution Paths for Quantified Boolean Formulas.

Contributions to the Theory of Practical Quantified Boolean Formula Solving.

[Wra76] Celia Wrathall.
Complete Sets and the Polynomial-Time Hierarchy.

[ZM02a] Lintao Zhang and Sharad Malik.
Conflict Driven Learning in a Quantified Boolean Satisfiability Solver.
[ZM02b] Lintao Zhang and Sharad Malik.
Towards a Symmetric Treatment of Satisfaction and Conflicts in Quantified
Boolean Formula Evaluation.