Preprocessing QBF: Failed Literals and Quantified Blocked Clause Elimination

Florian Lonsing
(joint work with Armin Biere and Martina Seidl)

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

Deduction at Scale Seminar
Ringberg Castle, Tegernsee, Germany
March 7 - 11, 2011
Motivation

Preprocessing Techniques for Quantified Boolean Formulae (QBF)
- Failed literals (FL) and quantified blocked clause elimination (QBCE).
- Positive effects on search- and elimination-based solvers.

Florian Lonsing (joint work with Armin Biere and Martina Seidl)
Part 1: Preliminaries
- From propositional logic (SAT) to QBF.
- QBF semantics.

Part 2: Failed Literal Detection (FL)
- Paper submitted to SAT’11.
- Necessary assignments and QBF models.

Part 3: Quantified Blocked Clause Elimination (QBCE)
- Paper submitted to CADE’11.
- From BCE for SAT to QBCE for QBF.
Part 1: Preliminaries
Propositional Logic (SAT):
- Our focus: formulae in conjunctive normal form (CNF).
- Set of Boolean variables $V := \{x_1, \ldots, x_m\}$.
- Literals $l := v$ or $l := \neg v$ for $v \in V$.
- Clauses $C_i := (l_1 \lor \ldots \lor l_k)$.
- CNF $\phi := \bigwedge C_i$.

Quantified Boolean Formulae (QBF):
- Prenex CNF: quantifier-free CNF over quantified Boolean variables.
- PCNF $Q_1 S_1 \ldots Q_n S_n. \phi$, where $Q_i \in \{\exists, \forall\}$, scopes $S_i$.
- Scope $S_i$: set of quantified variables.
- $Q_i S_i \leq Q_{i+1} S_{i+1}$: scopes are linearly ordered.

**Example**

Clauses (CNFs) are sets of literals (clauses).
A CNF: $\{x, \overline{y}\}, \{\overline{x}, y\}$ and a PCNF: $\forall x \exists y. \{x, \overline{y}\}, \{\overline{x}, y\}$. 

Florian Lonsing (joint work with Armin Biere and Martina Seidl)
Assignment Trees (AT):
- Assignment $A : V \rightarrow \{true, false\}$ maps variables to truth values.
- Paths from root to a leaf in AT represent assignments.
- Nodes along path (except root) assign truth values to variables.

CNF-Model:
- A path in the assignment tree of a CNF $\phi$ which satisfies all clauses.
- CNF $\phi$ is satisfiable iff it has a CNF-model $m : m \models \phi$.

Example

$\phi := \{e_1, \neg a_2, e_3\},$
$\{e_1, \neg a_2, \neg e_3\},$
$\{\neg e_1, a_2, \neg e_3\},$
$\{\neg e_1, \neg a_2, e_3\}$

Florian Lonsing (joint work with Armin Biere and Martina Seidl)
Assignment Trees (AT):
- Assignment $A : V \rightarrow \{true, false\}$ maps variables to truth values.
- Paths from root to a leaf in AT represent assignments.
- Nodes along path (except root) assign truth values to variables.

CNF-Model:
- A path in the assignment tree of a CNF $\phi$ which satisfies all clauses.
- CNF $\phi$ is satisfiable iff it has a CNF-model $m : m \models \phi$.

Example

$$\phi := \{e_1, \neg a_2, e_3\},$$
$$\{e_1, \neg a_2, \neg e_3\},$$
$$\{\neg e_1, a_2, \neg e_3\},$$
$$\{\neg e_1, \neg a_2, e_3\}$$
PCNF-Model: $\psi := Q_1 S_1 \ldots Q_n S_n \phi$

- An (incomplete) AT where every path is a CNF-model of CNF part $\phi$.
- Restriction: nodes which assign $\forall$-variables have exactly one sibling.
- PCNF $\psi$ is satisfiable iff it has a PCNF-model $m$: $m \models \psi$.

Example

$\psi := \exists e_1 \forall a_2 \exists e_3 . \phi$

$\phi := \{ e_1, \neg a_2, e_3 \},$
$\{ e_1, \neg a_2, \neg e_3 \},$
$\{ \neg e_1, a_2, \neg e_3 \},$
$\{ \neg e_1, \neg a_2, e_3 \}$
**PCNF-Model:** $\psi := Q_1 S_1 \ldots Q_n S_n. \phi$

- An (incomplete) AT where every path is a CNF-model of CNF part $\phi$.
- Restriction: nodes which assign $\forall$-variables have exactly one sibling.
- PCNF $\psi$ is satisfiable iff it has a PCNF-model $m$: $m \models \psi$.

**Example**

\[
\begin{align*}
\psi &:= \exists e_1 \forall a_2 \exists e_3. \phi \\
\phi &:= \{ e_1, \neg a_2, e_3 \}, \quad \{ e_1, \neg a_2, \neg e_3 \}, \\
&\quad \{ \neg e_1, a_2, e_3 \}, \quad \{ \neg e_1, \neg a_2, e_3 \}
\end{align*}
\]
Definition (Assignments of literals)

Given a PCNF $\psi$, the *assignment of a literal* $l$ yields the formula $\psi[l]$ where clauses $\text{Occs}(l)$ and literals $\neg l$ in $\text{Occs}(\neg l)$ are deleted.

Example

\[
\psi := \exists e_1 \forall a_2 \exists e_3, e_4. \phi \\
\phi := \{e_1, a_2, e_3, e_4\}, \\
\{e_1, a_2, \neg e_4\}, \\
\{\neg e_1, e_3, \neg e_4\}, \\
\{\neg a_2, \neg e_3\} \\
\psi[e_4]
\]
QBF Inference Rules (2/5)

**Definition (Universal Reduction)**

Given a clause \( C \), \( UR(C) := C \setminus \{ l_u \in L\forall(C) \mid \exists l_e \in L\exists(C), l_u < l_e \} \).

**Example**

\[
\psi := \exists e_1 \forall a_2 \exists e_3, e_4. \phi \\
\phi := UR(\{e_1, a_2\})
\]

- \( \{e_1, a_2\} \),
- \( \neg e_1, e_3 \),
- \( \neg a_2, \neg e_3 \)
Definition (Pure Literal Rule)

Given a PCNF $\psi$, a literal $l$ where $\text{Occs}(l) \neq \emptyset$ and $\text{Occs}(\neg l) = \emptyset$ is pure: if $q(l) = \exists$ then $\psi \equiv \psi[l]$, and if $q(l) = \forall$ then $\psi \equiv \psi[\neg l]$.

Example

\[ \psi := \exists e_1 \forall a_2 \exists e_3, e_4. \phi \]
\[ \phi := \{ e_1 \}, \{ \neg e_1, e_3 \}, \{ \neg a_2, \neg e_3 \} \]

Variable $a_2$ is pure: $\psi[a_2]$ (shortening clauses).
Definition (Unit Clause Rule)

Given a PCNF $\psi$. A clause $C \in \psi$ where $UR(C) = \{l\}$ is unit and $\psi \equiv \psi[l]$.

Example

\[
\begin{align*}
\psi & := \exists e_1 \forall a_2 \exists e_3, e_4. \phi \\
\phi & := \\
\{ e_1 \}, \\
\{ \neg e_1, e_3 \}, \\
\{ \neg e_3 \} \\
\end{align*}
\]

Clauses $\{ e_1 \}$ and $\{ \neg e_3 \}$ are unit: $\psi[e_1][\neg e_3]$. 
Definition (Boolean Constraint Propagation)

Given a PCNF $\psi$ and a literal $x$ called *assumption*. Formula $BCP(\psi, x)$ is obtained from $\psi[x]$ by applying UR, unit clause and pure literal rule.

Example

$$\psi := \exists e_1 \forall a_2 \exists e_3, e_4. \phi$$

$$\phi := \{\}$$

Empty clause derived from assumption $e_4$:

$$\emptyset \in BCP(\psi, e_4).$$
Part 2: Failed Literal Detection (FL)
Models and Necessary Assignments

Definition

Given PCNF $\psi$ and $x_i \in V$. Assignment $x_i \mapsto t$, where $t \in \{false, true\}$, is necessary for satisfiability of $\psi$ iff $x_i \mapsto t$ is part of every path in every PCNF-model of $\psi$.

Example

\[
\psi := \exists e_1 \forall a_2 \exists e_3. \phi
\]
\[
\phi := \{e_1, \neg a_2, e_3\},
\{e_1, \neg a_2, \neg e_3\},
\{\neg e_1, a_2, \neg e_3\},
\{\neg e_1, \neg a_2, e_3\}
\]

- $e_1 \mapsto true$ is necessary for satisfiability of $\psi$.

GOAL: Detection of (Subset of) Necessary Assignments in QBFs.
- Exponential reduction of search space.

Florian Lonsing (joint work with Armin Biere and Martina Seidl)
Models and Necessary Assignments

Definition

Given PCNF $\psi$ and $x_i \in V$. Assignment $x_i \mapsto t$, where $t \in \{false, true\}$, is necessary for satisfiability of $\psi$ iff $x_i \mapsto t$ is part of every path in every PCNF-model of $\psi$.

Example

\[
\psi := \exists e_1 \forall a_2 \exists e_3. \phi
\]
\[
\phi := \{e_1, \neg a_2, e_3\},
\{e_1, \neg a_2, \neg e_3\},
\{\neg e_1, a_2, \neg e_3\},
\{\neg e_1, \neg a_2, e_3\}
\]

- $e_1 \mapsto true$ is necessary for satisfiability of $\psi$.

**GOAL:** Detection of (Subset of) Necessary Assignments in QBFs.
- Exponential reduction of search space.

Florian Lonsing (joint work with Armin Biere and Martina Seidl)
Preprocessing QBF: FL and QBCE
Failed Literal Detection (FL) for SAT:
- BCP-based approach to detect subset of necessary assignments.
- Def. failed literal $x$ for CNF $\phi$: if $\emptyset \in BCP(\phi, x)$ then $\phi \equiv \phi \land \{\neg x\}$.
- FL based on deriving empty clause from assumption and BCP.

FL for QBF:
- Def.: failed literal $x$ for PCNF $\psi$: if $\psi \equiv \psi \land \{\neg x\}$.
- Problem: BCP-based approach like for SAT is unsound due to $\exists/\forall$ prefix.

Example

$\psi := \forall x \exists y. \{x, \neg y\}, \{\neg x, y\}$. We have $\emptyset \in BCP(\psi, y)$ but $\psi \not\equiv \psi \land \{\neg y\}$.

Our Work:
- Two orthogonal FL approaches for QBF.
- Soundness established by abstraction and Q-resolution.
Failed Literal Detection (FL) for SAT:

- BCP-based approach to detect subset of necessary assignments.
- Def. failed literal $x$ for CNF $\phi$: if $\emptyset \in BCP(\phi, x)$ then $\phi \equiv \phi \land \{\neg x\}$.
- FL based on deriving empty clause from assumption and BCP.

FL for QBF:

- Def.: failed literal $x$ for PCNF $\psi$: if $\psi \equiv \psi \land \{\neg x\}$.
- Problem: BCP-based approach like for SAT is unsound due to $\exists/\forall$ prefix.

Example

$$\psi := \forall x \exists y. \{x, \neg y\}, \{-x, y\}. \text{ We have } \emptyset \in BCP(\psi, y) \text{ but } \psi \not\equiv \psi \land \{\neg y\}.$$ 

Our Work:

- Two orthogonal FL approaches for QBF.
- Soundness established by abstraction and Q-resolution.
Abstraction-Based FL

**Problem:** $BCP(\psi, x)$ with assumption $x$ for FL on PCNF $\psi$ is unsound.

**Definition (Quantifier Abstraction)**

For $\psi := Q_1S_1 \ldots Q_{i-1}S_{i-1}Q_iS_i \ldots \ldots Q_nS_n. \phi$, the quantifier abstraction of $\psi$ with respect to $S_i$ is $Abs(\psi, i) := \exists(S_1 \cup \ldots \cup S_{i-1})Q_iS_i \ldots Q_nS_n. \phi$.

**Idea:** carry out BCP on abstraction of $\psi$.

- If $x \in S_i$ then treat all variables smaller than $x$ as existentially quantified.
- Example: $Abs(\exists x \forall y \exists z. \phi, 3) = \exists x \exists y \exists z. \phi$.
- Overapproximation: if $m \models \psi$ then $m \models Abs(\psi, i)$.

**Theorem**

*Given PCNF $\psi := Q_1S_1 \ldots Q_nS_n. \phi$ and literal $x$ where $v(x) \in S_i$. If $\emptyset \in BCP(Abs(\psi, i), x)$ then $\psi \equiv \psi \land \{\neg x\}$.*

**Practical Application:**

- FL using BCP on abstraction is sound and runs in polynomial-time.
BCP-Guided Q-Resolution (1/2)

Definition (Q-resolution)

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

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

**Q-Resolution:** combination of propositional resolution and UR.

- For PCNF $\psi$, clause $C$: if $\psi \vdash^* C$ then $\psi \equiv \psi \land C$.

**Idea:** (heuristically) validate $\emptyset \in BCP(\psi, x)$ on original PCNF.

- Try to derive the negated assumption $\{\neg x\}$ by Q-resolution.
- Resolution candidates are selected from clauses “touched” by BCP.
- Like conflict-driven clause learning (CDCL) in search-based solvers.
Novel Approach: BCP-Guided Q-Resolution (2/2)

Corollary

Given PCNF $\psi := Q_1 S_1 \ldots Q_n S_n. \phi$ and literal $x$ where $v(x) \in S_i$. If $\emptyset \in BCP(\psi, x)$ and $\psi \vdash^* \{\neg x\}$ then $\psi \equiv \psi \land \{\neg x\}$.

Example

$\psi := \exists e_1, e_2 \forall a_3 \exists e_4, e_5. \{a_3, e_5\}, \{-e_2, e_4\}, \{-e_1, e_4\}, \{e_1, e_2, \neg e_5\}$. With assumption $\neg e_4$ we get $\emptyset \in BCP(\psi, \neg e_4)$ since $\{-e_1\}$, $\{-e_2\}$ and $\{-e_5\}$ become unit. Finally $\{a_3, e_5\}$ is empty by UR. The negated assumption $\{e_4\}$ is then derived by resolving clauses in reverse-chronological order as they were affected by assignments: $(\{a_3, e_5\}, \{e_1, e_2, \neg e_5\}) \vdash \{e_1, e_2\}$, $(\{e_1, e_2\}, \{-e_2, e_4\}) \vdash \{e_1, e_4\}$, $(\{e_1, e_4\}, \{-e_1, e_4\}) \vdash \{e_4\}$.

Practical Application:

- Advantage: original prefix allows full propagation power in BCP.
- BCP-based selection of resolution candidates is only a heuristic.
### Proposition

*Abstraction-based FL and BCP-guided Q-resolution are orthogonal to each other with respect to detecting necessary assignments.*

### Consequences:
- There are PCNFs where one approach can detect a necessary assignment the other one cannot.
- No approach can detect all necessary assignments.
- Crucial observation: Q-resolution for CDCL is not optimal (see below)!
- (How) Can we apply quantifier abstraction for clause learning?

### Example

\[ \psi := \forall a_1 \exists e_2, e_3 \forall a_4 \exists e_5. \ \{ a_1, e_2 \}, \ \{ \neg a_1, e_3 \}, \ \{ e_3, \neg e_5 \}, \ \{ a_1, e_2, \neg e_3 \}, \ \{ \neg e_2, a_4, e_5 \}. \]

We have \( \emptyset \in BCP(Abs(\psi, 2), \neg e_3) \) but \( \psi \not\models^* \{ e_3 \} \): assignment \( \{ e_3 \} \mapsto \text{true} \) is necessary but Q-resolution can *not* derive clause \( \{ e_3 \} \).
**Tool “QxBF”:** FL-based preprocessor operating in rounds.
**SAT-Based FL:** using SAT solver to detect necessary assignments.

**QBF-EVAL’10: 568 formulae**

<table>
<thead>
<tr>
<th>Preprocessing</th>
<th>Solver</th>
<th>Solved</th>
<th>Time (Preproc.)</th>
<th>SAT</th>
<th>UNSAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAT</td>
<td>DepQBF</td>
<td>379</td>
<td>322.31 (7.17)</td>
<td>167</td>
<td>212</td>
</tr>
<tr>
<td>QRES+SAT</td>
<td></td>
<td>378</td>
<td>322.83 (6.22)</td>
<td>167</td>
<td>211</td>
</tr>
<tr>
<td>ABS+SAT</td>
<td></td>
<td>378</td>
<td>323.19 (7.21)</td>
<td>167</td>
<td>211</td>
</tr>
<tr>
<td>ABS</td>
<td>DepQBF</td>
<td>375</td>
<td>327.64 (3.33)</td>
<td>168</td>
<td>207</td>
</tr>
<tr>
<td>QRES</td>
<td></td>
<td>374</td>
<td>327.63 (1.83)</td>
<td>167</td>
<td>207</td>
</tr>
<tr>
<td>None</td>
<td></td>
<td>372</td>
<td>334.60 (—)</td>
<td>166</td>
<td>206</td>
</tr>
<tr>
<td>ABS+SAT</td>
<td>Quantor</td>
<td>229</td>
<td>553.65 (7.21)</td>
<td>112</td>
<td>117</td>
</tr>
<tr>
<td>none</td>
<td>Nenofex</td>
<td>224</td>
<td>553.37 (7.21)</td>
<td>104</td>
<td>120</td>
</tr>
<tr>
<td>none</td>
<td>Quantor</td>
<td>211</td>
<td>573.65 (—)</td>
<td>103</td>
<td>108</td>
</tr>
<tr>
<td>ABS+SAT</td>
<td>squolem</td>
<td>154</td>
<td>658.28 (7.21)</td>
<td>63</td>
<td>91</td>
</tr>
<tr>
<td>None</td>
<td></td>
<td>124</td>
<td>708.80 (—)</td>
<td>53</td>
<td>71</td>
</tr>
</tbody>
</table>

**Table:** Solver performance with(out) time-limited failed literal preprocessing. Search-based solver DepQBF, elimination-based solvers Quantor, squolem, Nenofex. No preprocessing ("none"), SAT-based FL ("SAT"), abstraction-based FL ("ABS") and BCP-guided Q-resolution ("QRES").
Part 3: Quantified Blocked Clause Elimination (QBCE)
Blocked Clause Elimination (BCE) for SAT [JBH10]
- Allows CNF-level simplifications after circuit-to-CNF transformation.
- At least as effective as many circuit-level preprocessing techniques.
- Simulates pure literal rule, Plaisted-Greenbaum encoding, . . .

Quantified Blocked Clause Elimination (QBCE) for QBF
- Paper submitted to CADE’11 (joint work with Armin Biere, Martina Seidl).
- Generalizes BCE to QBF: minor but crucial adaption of BCE definition.
- Implementation: tool “bloqqer” combines QBCE and extensions with variable elimination, self-subsuming resolution, subsumption, . . .

Definition of QBCE: based checking possible Q-resolvents.

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

1. Tentative resolvent: $C_1 \otimes C_2 := (UR(C_1) \cup UR(C_2)) \setminus \{v, \neg v\}$.
Blocked Clause Elimination (BCE) for SAT [JBH10]
- Allows CNF-level simplifications after circuit-to-CNF transformation.
- At least as effective as many circuit-level preprocessing techniques.
- Simulates pure literal rule, Plaisted-Greenbaum encoding, . . .

Quantified Blocked Clause Elimination (QBCE) for QBF
- Paper submitted to CADE’11 (joint work with Armin Biere, Martina Seidl).
- Generalizes BCE to QBF: minor but crucial adaption of BCE definition.
- Implementation: tool “bloqcer” combines QBCE and extensions with variable elimination, self-subsuming resolution, subsumption, . . .

Definition of QBCE: based checking possible Q-resolvents.

Definition

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

1. Tentative resolvent: $C_1 \otimes C_2 := (UR(C_1) \cup UR(C_2)) \setminus \{v, \neg v\}$. 
**QBCE Definition**

**Definition (Quantified Blocking Literal)**

Given PCNF $\psi := Q_1 S_1 \ldots Q_n S_n$. $\phi$, a literal $l$ in a clause $C \in \psi$ is called *quantified blocking literal* if for every clause $C'$ with $\neg l \in C'$, there exists a literal $k$ such that $\{k, \neg k\} \subseteq C \otimes C'$ with $k \leq l$.

**Definition (Quantified Blocked Clause)**

Given PCNF $Q_1 S_1 \ldots Q_n S_n$. $(\phi \land C)$. Clause $C$ is *quantified blocked* if it contains a quantified blocking literal.

Then $Q_1 S_1 \ldots Q_n S_n$. $(\phi \land C) \equiv Q_1 S_1 \ldots Q_n S_n. \phi$.

<table>
<thead>
<tr>
<th>$C_1 \in Occs(l)$ blocked?</th>
<th>$C_2 \in Occs(\neg l)$</th>
<th>$C_1 \otimes C_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(x_1 \lor x_2 \lor \ldots \lor x_n \lor \ldots \lor l \lor \ldots)$</td>
<td>$(\ldots \neg x_1 \lor \ldots \lor \neg l \lor \ldots)$</td>
<td>${x_1, \neg x_1} \in C_1 \otimes C_2$</td>
</tr>
<tr>
<td></td>
<td>$(\ldots \neg x_2 \lor \ldots \lor \neg l \lor \ldots)$</td>
<td>${x_2, \neg x_2} \in C_1 \otimes C_2$</td>
</tr>
<tr>
<td></td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td></td>
<td>$(\ldots \neg x_n \lor \ldots \lor \neg l \lor \ldots)$</td>
<td>${x_n, \neg x_n} \in C_1 \otimes C_2$</td>
</tr>
</tbody>
</table>

**Example**

All clauses blocked: $\forall x \exists y((x \lor \neg y) \land (\neg x \lor y))$.

No clause blocked: $\exists x \forall y((x \lor \neg y) \land (\neg x \lor y))$.

Florian Lonsing (joint work with Armin Biere and Martina Seidl)  
Preprocessing QBF: FL and QBCE
QBCE Definition

Definition (Quantified Blocking Literal)
Given PCNF $\psi := Q_1 S_1 \ldots Q_n S_n$. $\phi$, a literal $l$ in a clause $C \in \psi$ is called quantified blocking literal if for every clause $C'$ with $\neg l \in C'$, there exists a literal $k$ such that $\{k, \neg k\} \subseteq C \otimes C'$ with $k \leq l$.

Definition (Quantified Blocked Clause)
Given PCNF $Q_1 S_1 \ldots Q_n S_n$. $(\phi \land C)$. Clause $C$ is quantified blocked if it contains a quantified blocking literal.
Then $Q_1 S_1 \ldots Q_n S_n. (\phi \land C) \equiv Q_1 S_1 \ldots Q_n S_n. \phi$.

<table>
<thead>
<tr>
<th>$C_1 \in Occs(l)$ blocked?</th>
<th>$C_2 \in Occs(\neg l)$</th>
<th>$C_1 \otimes C_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>($x_1 \lor x_2 \lor \ldots \lor x_n \lor \ldots \lor l \lor \ldots$)</td>
<td>($\ldots \neg x_1 \lor \ldots \lor \neg l \lor \ldots$)</td>
<td>${x_1, \neg x_1} \in C_1 \otimes C_2$</td>
</tr>
<tr>
<td>($\ldots \neg x_2 \lor \ldots \lor \neg l \lor \ldots$)</td>
<td>($\ldots \neg x_2 \lor \ldots \lor \neg l \lor \ldots$)</td>
<td>${x_2, \neg x_2} \in C_1 \otimes C_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>($\ldots \neg x_n \lor \ldots \lor \neg l \lor \ldots$)</td>
<td>${x_n, \neg x_n} \in C_1 \otimes C_2$</td>
<td></td>
</tr>
</tbody>
</table>

Example

All clauses blocked: $\forall x \exists y((x \lor \neg y) \land (\neg x \lor y))$.
No clause blocked: $\exists x \forall y((x \lor \neg y) \land (\neg x \lor y))$. 
QBCE Definition

Definition (Quantified Blocking Literal)
Given PCNF $\psi := Q_1 S_1 \ldots Q_n S_n$. $\phi$, a literal $l$ in a clause $C \in \psi$ is called quantified blocking literal if for every clause $C'$ with $\neg l \in C'$, there exists a literal $k$ such that $\{k, \neg k\} \subseteq C \otimes C'$ with $k \leq l$.

Definition (Quantified Blocked Clause)
Given PCNF $Q_1 S_1 \ldots Q_n S_n$. $(\phi \land C)$. Clause $C$ is quantified blocked if it contains a quantified blocking literal.
Then $Q_1 S_1 \ldots Q_n S_n. (\phi \land C) \equiv Q_1 S_1 \ldots Q_n S_n. \phi$.

<table>
<thead>
<tr>
<th>$C_1 \in Occs(l)$ blocked?</th>
<th>$C_2 \in Occs(\neg l)$</th>
<th>$C_1 \otimes C_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(x_1 \lor x_2 \lor \ldots \lor x_n \lor \ldots \lor l \lor \ldots)$</td>
<td>$(\ldots \neg x_1 \lor \ldots \lor \neg l \lor \ldots)$</td>
<td>${x_1, \neg x_1} \in C_1 \otimes C_2$</td>
</tr>
<tr>
<td></td>
<td>$(\ldots \neg x_2 \lor \ldots \lor \neg l \lor \ldots)$</td>
<td>${x_2, \neg x_2} \in C_1 \otimes C_2$</td>
</tr>
<tr>
<td></td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td></td>
<td>$(\ldots \neg x_n \lor \ldots \lor \neg l \lor \ldots)$</td>
<td>${x_n, \neg x_n} \in C_1 \otimes C_2$</td>
</tr>
</tbody>
</table>

Example
All clauses blocked: $\forall x \exists y((x \lor \neg y) \land (\neg x \lor y))$.
No clause blocked: $\exists x \forall y((x \lor \neg y) \land (\neg x \lor y))$. 
Definition (Quantified Blocking Literal)

Given PCNF $\psi := Q_1 S_1 \ldots Q_n S_n \cdot \phi$, a literal $l$ in a clause $C \in \psi$ is called \textit{quantified blocking literal} if for every clause $C'$ with $\neg l \in C'$, there exists a literal $k$ such that $\{k, \neg k\} \subseteq C \otimes C'$ with $k \leq l$.

Definition (Quantified Blocked Clause)

Given PCNF $Q_1 S_1 \ldots Q_n S_n \cdot (\phi \land C)$. Clause $C$ is \textit{quantified blocked} if it contains a quantified blocking literal.

Then $Q_1 S_1 \ldots Q_n S_n \cdot (\phi \land C) \equiv Q_1 S_1 \ldots Q_n S_n \cdot \phi$.

<table>
<thead>
<tr>
<th>$C_1 \in Occs(l)$ blocked?</th>
<th>$C_2 \in Occs(\neg l)$</th>
<th>$C_1 \otimes C_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1 \in Occs(l)$ blocked?</td>
<td>$C_2 \in Occs(\neg l)$</td>
<td>$C_1 \otimes C_2$</td>
</tr>
<tr>
<td>$(x_1 \lor x_2 \lor \ldots \lor x_n \lor \ldots \lor l \lor \ldots)$</td>
<td>$(\ldots \neg x_1 \lor \ldots \lor \neg l \lor \ldots)$</td>
<td>${x_1, \neg x_1} \in C_1 \otimes C_2$</td>
</tr>
<tr>
<td>$(\ldots \neg x_2 \lor \ldots \lor \neg l \lor \ldots)$</td>
<td>${x_2, \neg x_2} \in C_1 \otimes C_2$</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$(\ldots \neg x_n \lor \ldots \lor \neg l \lor \ldots)$</td>
<td>${x_n, \neg x_n} \in C_1 \otimes C_2$</td>
<td></td>
</tr>
</tbody>
</table>

Example

All clauses blocked: $\forall x \exists y((x \lor \neg y) \land (\neg x \lor y))$.

No clause blocked: $\exists x \forall y((x \lor \neg y) \land (\neg x \lor y))$. 

Florian Lonsing (joint work with Armin Biere and Martina Seidl)
Table: Bloqqr (QBCE, extensions and related techniques) combined with search-(DepQBF, QuBE) and elimination-based (Nenofex, Quantor) solvers.

<table>
<thead>
<tr>
<th>preprocessor</th>
<th># formulas</th>
<th>runtime (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SOLVED</td>
<td>SAT</td>
</tr>
<tr>
<td><strong>DepQBF</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bloqqr</td>
<td>467</td>
<td>224</td>
</tr>
<tr>
<td>no preprocessing</td>
<td>373</td>
<td>167</td>
</tr>
<tr>
<td><strong>QuBE</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bloqqr</td>
<td>444</td>
<td>200</td>
</tr>
<tr>
<td>no preprocessing</td>
<td>332</td>
<td>135</td>
</tr>
<tr>
<td><strong>Quantor</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bloqqr</td>
<td>288</td>
<td>145</td>
</tr>
<tr>
<td>no preprocessing</td>
<td>206</td>
<td>100</td>
</tr>
<tr>
<td><strong>Nenofex</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bloqqr</td>
<td>268</td>
<td>132</td>
</tr>
<tr>
<td>no preprocessing</td>
<td>221</td>
<td>107</td>
</tr>
</tbody>
</table>

Florian Lonsing (joint work with Armin Biere and Martina Seidl)

Preprocessing QBF: FL and QBCE
BL: bloqger with QBCE, extensions and related techniques.

Florian Lonsing (joint work with Armin Biere and Martina Seidl)

Preprocessing QBF: FL and QBCE
Conclusions

Preprocessing QBF:
- Positive effects on elimination- and search-based QBF solvers.

Failed Literal Detection (FL):
- Detecting a subset of necessary assignments.
- Exponential reduction of search-space.
- Soundness by abstraction and Q-resolution.
- Orthogonality: current CDCL approaches in QBF are not optimal.

Quantified Blocked Clause Elimination (QBCE):
- Generalizes BCE for SAT to QBF.
- Best performance when combined with variable elimination, ...  

Work in Progress:
- Papers submitted to SAT’11 (FL) and CADE’11 (QBCE).
- Source code of our preprocessors will be published.
- Dynamic applications of FL and QBCE.
QxBF (FL) and bloqquer (QBCE)

Florian Lonsing (joint work with Armin Biere and Martina Seidl)

Preprocessing QBF: FL and QBCE
References
U. Bubeck and H. Kleine Büning.
Bounded Universal Expansion for Preprocessing QBF.

D. Le Berre.
Exploiting the Real Power of Unit Propagation Lookahead.

A. Biere.
Resolve and Expand.

H. Kleine Büning, M. Karpinski, and A. Flögel.
Resolution for Quantified Boolean Formulas.

F. Bacchus and J. Winter.
Effective Preprocessing with Hyper-Resolution and Equality Reduction.

E. Giunchiglia, P. Marin, and Massimo Narizzano.
sQueezeBF: An Effective Preprocessor for QBFs Based on Equivalence Reasoning.

M. Järvisalo, A. Biere, and M. Heule.
Blocked Clause Elimination.

I. Lynce and J. P. Marques Silva.
Probing-Based Preprocessing Techniques for Propositional Satisfiability.

F. Pigorsch and C. Scholl.
An AIG-Based QBF-Solver Using SAT for Preprocessing.

J. Rintanen.
Improvements to the Evaluation of Quantified Boolean Formulae.

H. Samulowitz and F. Bacchus.
Using SAT in QBF.
H. Samulowitz and F. Bacchus.
Binary Clause Reasoning in QBF.

H. Samulowitz, J. Davies, and F. Bacchus.
Preprocessing QBF.