CAQE: A Certifying QBF Solver

Markus N. Rabe[1] Leander Tentrup[2]

[1]University of California at Berkeley, [2]Saarland University

FMCAD
Austin, Texas, September 29 2015

Quantified boolean formulas

- TrueQBF is the prototypical PSPACE problem
- Compact version of SAT
- Verification/synthesis/artificial intelligence

Contribution - A QBF Algorithm

- Simple and CEGAR-based ($\sim$3K loc w/o SAT solver)
- Competitive performance
- Produces certificates
- Handles deep quantifier alternations

QBF - Example

$$\exists x \forall y \exists z : (x \vee y \vee \bar{z}) \wedge (\bar{x} \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee z)$$

QBF - Example

$$\exists x \forall y \exists z : (x \vee y \vee \bar{z}) \wedge (\bar{x} \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee z)$$

$$\text{Choose } x = \text{true} : \quad \forall y \exists z : (y \vee \bar{z}) \wedge (\bar{y} \vee z)$$

QBF - Example

$$\exists x \forall y \exists z : (x \lor y \lor \bar{z}) \land (\bar{x} \lor y \lor \bar{z}) \land (\bar{x} \lor \bar{y} \lor z)$$

$$\begin{array}{ll} \text{Choose } x = \text{true}: & \forall y \exists z : (y \lor \bar{z}) \land (\bar{y} \lor z) \\ \text{Case } y = \text{true}: & \exists z : z \\ \text{Case } y = \text{false}: & \exists z : \bar{z} \end{array}$$

QBF - Example

$$\exists x \forall y \exists z : (x \vee y \vee \bar{z}) \wedge (\bar{x} \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee z)$$

$$\begin{array}{ll} \text{Choose } x = \text{true}: & \forall y \exists z : (y \vee \bar{z}) \wedge (\bar{y} \vee z) \\ \text{Case } y = \text{true}: & \exists z : z \\ \text{Case } y = \text{false}: & \exists z : \bar{z} \end{array}$$

This formula is true!

Clausal abstractions

Construct one SAT solver per quantifier level.

$\exists x \forall y \exists z :$

$$
\begin{array}{lll}
(x \vee & y & \vee \bar{z}) \\
(\bar{x} \vee & y & \vee \bar{z}) \\
(\bar{x} \vee & \bar{y} & \vee z)
\end{array}
$$

Clausal abstractions

Construct one SAT solver per quantifier level.

$\exists x \forall y \exists z :$

$$
\begin{array}{lll}
(x \vee & y & \vee \bar{z}) \\
(\bar{x} \vee & y & \vee \bar{z}) \\
(\bar{x} \vee & \bar{y} & \vee z)
\end{array}
$$

Clausal abstractions

Construct one SAT solver per quantifier level.

$\exists x \forall y \exists z:$

$$
\begin{array}{lll}
(x \vee b_1) & (\overline{t_1} \rightarrow (y \rightarrow \overline{b_1})) & (t_1 \vee \bar{z}) \\
(\bar{x} \vee & y & \vee \bar{z}) \\
(\bar{x} \vee & \bar{y} & \vee z)
\end{array}
$$

Clausal abstractions

Construct one SAT solver per quantifier level.

$\exists x \forall y \exists z :$

$$
\begin{array}{lll}
(x \vee b_1) & (\overline{t_1} \rightarrow (y \rightarrow \overline{b_1})) & (t_1 \vee \bar{z}) \\
(\bar{x} \vee b_2) & (\overline{t_2} \rightarrow (y \rightarrow \overline{b_2})) & (t_2 \vee \bar{z}) \\
(\bar{x} \vee b_3) & (\overline{t_3} \rightarrow (\bar{y} \rightarrow \overline{b_3})) & (t_3 \vee z)
\end{array}
$$

Clausal abstractions

Construct one SAT solver per quantifier level.

$\exists x \forall y \exists z :$

$$
\begin{array}{ccc}
(x \lor b_1) & (t_1 \lor \bar{y}) & (t_1 \lor \bar{z}) \\
(\bar{x} \lor b_2) & (t_2 \lor \bar{y}) & (t_2 \lor \bar{z}) \\
(\bar{x} \lor b_3) & (t_3 \lor y) & (t_3 \lor z)
\end{array}
$$

Clausal abstractions

Construct one SAT solver per quantifier level.

$\exists x \, \forall y \, \exists z :$

$\varphi_{\exists x}$	$\varphi_{\forall y}$	$\varphi_{\exists z}$
$(x \vee b_1)$	$(t_1 \vee \bar{y})$	$(t_1 \vee \bar{z})$
$(\bar{x} \vee b_2)$	$(t_2 \vee \bar{y})$	$(t_2 \vee \bar{z})$
$(\bar{x} \vee b_3)$	$(t_3 \vee y)$	$(t_3 \vee z)$

Clausal abstractions - general case

Given $Q_1 X_1 \ldots Q_n X_n : \bigwedge C_i$

$$
\begin{aligned}
\varphi_{\exists X_m} &= \bigwedge_{C_i} \left(\left(\bigvee_{l \in C_i, \mathrm{level}(l) = m} l \right) \vee t_i \vee b_i \right) \\
\varphi_{\forall X_m} &= \bigwedge_{C_i} \left(\bigwedge_{l \in C_i, \mathrm{level}(l) = m} (\bar{l} \vee t_i) \right)
\end{aligned}
$$

Clausal abstractions - general case

Given $Q_1X_1 \ldots Q_nX_n : \bigwedge C_i$

$$
\begin{aligned}
\varphi_{\exists X_m} &= \bigwedge_{C_i} \left(\left(\bigvee_{l \in C_i, \text{level}(l)=m} l \right) \vee t_i \vee b_i \right) \\
\varphi_{\forall X_m} &= \bigwedge_{C_i} \left(\bigwedge_{l \in C_i, \text{level}(l)=m} (\bar{l} \vee t_i) \right)
\end{aligned}
$$

Let $\mathbf{t}$ be a assignment to the variables t_i. Represents the clauses *that have been satisfied already*.

Two algorithms:

- $\textsc{solve}_\exists(\exists X_m \ldots Q_nX_n : \psi, \mathbf{t})$
- $\textsc{solve}_\forall(\forall X_m \ldots Q_nX_n : \psi, \mathbf{t})$

Return value:
(result, minimized assumptions, unsat core over assumptions)

Algorithm

procedure SOLVE$_\exists$($\exists X.\ \Psi$, $\mathbf{t}$)
  while true **do**
    $result, \mathbf{b}, failed \leftarrow$ SAT($\varphi_X, \mathbf{t}$)
    if $result =$ UNSAT **then**
      return UNSAT, _, $failed$
    else if Ψ is propositional **then**
      return SAT, $\mathbf{t}$, _
    $\mathbf{t_b} \leftarrow \{t_i \mid b_i \notin \mathbf{b},\ 1 \leq i \leq k\}$
    $result, \mathbf{t}', failed' \leftarrow$ SOLVE$_\forall$($\Psi, \mathbf{t} \cup \mathbf{t_b}$)
    if $result =$ UNSAT **then**
      $\varphi_X \leftarrow \varphi_X \wedge \left(\bigvee_{t \in failed'} \neg b_t\right)$
    else
      return SAT, $\mathbf{t}'$, _

Algorithm (2)

procedure SOLVE$_\forall(\forall X.\ \Psi, \mathbf{t})$
    while true **do**
        $result, \mathbf{t}', failed \leftarrow$ SAT$(\varphi_X, \mathbf{t}^+)$
        if $result =$ UNSAT **then**
            return SAT, $failed$, _
        $result, \mathbf{t}'', failed' \leftarrow$ SOLVE$_\exists(\Psi, \mathbf{t}')$
        if $result =$ SAT **then**
            $\varphi_X \leftarrow \varphi_X \wedge (\bigvee_{t \in \mathbf{t}''} \neg t)$
        else
            return UNSAT, _, $failed'$

Example (2)

$$(x \vee b_1) \quad (\bar{x} \vee b_2) \quad (\bar{x} \vee b_3)$$

$\varphi \exists x$

$$(t_1 \vee \bar{y}) \quad (t_2 \vee \bar{y}) \quad (t_3 \vee y)$$

$\varphi \forall y$

$$(t_1 \vee \bar{z}) \quad (t_2 \vee \bar{z}) \quad (t_3 \vee z)$$

$\varphi \exists z$

Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$\varphi_{\exists x}$	$\varphi_{\forall y}$	$\varphi_{\exists z}$
$(x \vee b_1)$	$(t_1 \vee \bar{y})$	$(t_1 \vee \bar{z})$
$(\bar{x} \vee b_2)$	$(t_2 \vee \bar{y})$	$(t_2 \vee \bar{z})$
$(\bar{x} \vee b_3)$	$(t_3 \vee y)$	$(t_3 \vee z)$

Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$(x \vee b_1)$	$(t_1 \vee \bar{y})$	$(t_1 \vee \bar{z})$
$(\bar{x} \vee b_2)$	$(t_2 \vee \bar{y})$	$(t_2 \vee \bar{z})$
$(\bar{x} \vee b_3)$	$(t_3 \vee y)$	$(t_3 \vee z)$
$\varphi_{\exists x}$	$\varphi_{\forall y}$	$\varphi_{\exists z}$

Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$\varphi_{\exists x}$	$\varphi_{\forall y}$	$\varphi_{\exists z}$
$(x \vee b_1)$	$(t_1 \vee \bar{y})$	$(t_1 \vee \bar{z})$
$(\bar{x} \vee b_2)$	$(t_2 \vee \bar{y})$	$(t_2 \vee \bar{z})$
$(\bar{x} \vee b_3)$	$(t_3 \vee y)$	$(t_3 \vee z)$

Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$$(x \vee b_1) \quad (\bar{x} \vee b_2) \quad (\bar{x} \vee b_3)$$

$\varphi_{\exists x}$

$$(t_1 \vee \bar{y}) \quad (t_2 \vee \bar{y}) \quad (t_3 \vee y)$$

$\varphi_{\forall y}$

$$(t_1 \vee \bar{z}) \quad (t_2 \vee \bar{z}) \quad (t_3 \vee z)$$

$\varphi_{\exists z}$

Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$$\begin{array}{ccc} (x \vee b_1) & (t_1 \vee \bar{y}) & (t_1 \vee \bar{z}) \\ (\bar{x} \vee b_2) & (t_2 \vee \bar{y}) & (t_2 \vee \bar{z}) \\ (\bar{x} \vee b_3) & (t_3 \vee y) & (t_3 \vee z) \\ \varphi_{\exists x} & \varphi_{\forall y} & \varphi_{\exists z} \end{array}$$

Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$\varphi_{\exists x}$	$\varphi_{\forall y}$	$\varphi_{\exists z}$
$(x \vee b_1)$	$(t_1 \vee \bar{y})$	$(t_1 \vee \bar{z})$
$(\bar{x} \vee b_2)$	$(t_2 \vee \bar{y})$	$(t_2 \vee \bar{z})$
$(\bar{x} \vee b_3)$	$(t_3 \vee y)$	$(t_3 \vee z)$

Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$$\begin{array}{ccc} (x \vee b_1) & (t_1 \vee \overline{y}) & (t_1 \vee \overline{z}) \\ (\overline{x} \vee b_2) & (t_2 \vee \overline{y}) & (t_2 \vee \overline{z}) \\ (\overline{x} \vee b_3) & (t_3 \vee y) & (t_3 \vee z) \\ \varphi_{\exists x} & \varphi_{\forall y} \wedge \overline{t_2} & \varphi_{\exists z} \end{array}$$

refine!

Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$\varphi_{\exists x}$	$\varphi_{\forall y} \wedge \overline{t_2}$	$\varphi_{\exists z}$
$(x \vee b_1)$	$(t_1 \vee \bar{y})$	$(t_1 \vee \bar{z})$
$(\bar{x} \vee b_2)$	$(t_2 \vee \bar{y})$	$(t_2 \vee \bar{z})$
$(\bar{x} \vee b_3)$	$(t_3 \vee y)$	$(t_3 \vee z)$

Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$\varphi_{\exists x}$	$\varphi_{\forall y} \wedge \overline{t_2}$	$\varphi_{\exists z}$
$(x \vee b_1)$	$(t_1 \vee \bar{y})$	$(t_1 \vee \bar{z})$
$(\bar{x} \vee b_2)$	$(t_2 \vee \bar{y})$	$(t_2 \vee \bar{z})$
$(\bar{x} \vee b_3)$	$(t_3 \vee y)$	$(t_3 \vee z)$

Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$$(x \vee b_1) \quad (\bar{x} \vee b_2) \quad (\bar{x} \vee b_3)$$

$\varphi_{\exists x}$

$$(t_1 \vee \bar{y}) \quad (t_2 \vee \bar{y}) \quad (t_3 \vee y)$$

$\varphi_{\forall y} \wedge \overline{t_2}$

$$(t_1 \vee \bar{z}) \quad (t_2 \vee \bar{z}) \quad (t_3 \vee z)$$

$\varphi_{\exists z}$

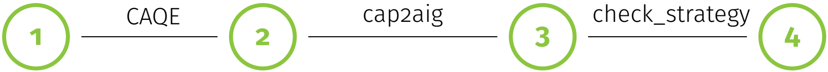
Variable assignments
Interface variable assignments
Interface variable assumptions

Example (2)

$(x \vee b_1)$ $(\bar{x} \vee b_2)$ $(\bar{x} \vee b_3)$	$(t_1 \vee \bar{y})$ $(t_2 \vee \bar{y})$ $(t_3 \vee y)$	$(t_1 \vee \bar{z})$ $(t_2 \vee \bar{z})$ $(t_3 \vee z)$
$\varphi_{\exists x}$	$\varphi_{\forall y} \wedge \overline{t_2} \wedge \overline{t_3}$ refine!	$\varphi_{\exists z}$

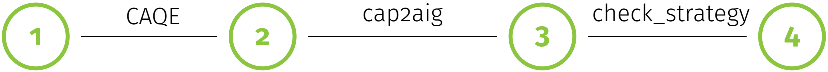
Variable assignments
Interface variable assignments
Interface variable assumptions

Certification




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

Certification




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

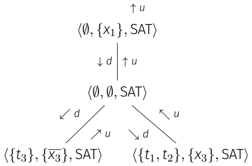
```
p cap 3 3
d
d
6 -3
u SAT
d
4 5 3
u SAT
u SAT
1
u SAT
r SAT
```

Certification

1 — CAQE — 2 — cap2aig — 3 — check_strategy — 4

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

```
p cap 3 3
d
d
6 -3
u SAT
d
4 5 3
u SAT
u SAT
1
u SAT
r SAT
```



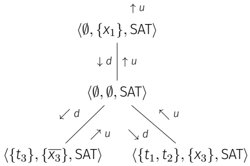
`skolem.aig`

Certification

1 — CAQE — 2 — cap2aig — 3 — check_strategy — 4

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

```
p cap 3 3
d
d
6 -3
u SAT
d
4 5 3
u SAT
u SAT
1
u SAT
r SAT
```



skolem.aig

✓

Experimental Evaluation

Implementation

- CAQE (Clausal Abstraction for Quantifier Elimination)
- $\sim$3K loc w/o SAT solver
- `https://www.react.uni-saarland.de/tools/caqe/`

Evaluation

- Compared against state-of-the-art QBF solvers DepQBF, RAReQS, GhostQ
- Benchmark: QBFGallery2014
- With/without preprocessing
- PicoSAT/MiniSAT

Performance - with preprocessing

Number of instances solved within 10 minutes.

Family	total	CAQE picosat+bloqqer	CAQE minisat+bloqqer	RAReQS rareqs+bloqqer	GhostQ ghostq*	DepQBF depqbf+bloqqer
eval2012r2	276	112	98	**129**	124	128
bomb	132	74	59	**82**	75	80
complexity	104	67	67	**91**	26	57
dungeon	107	31	**69**	62	45	66
hardness	114	**103**	94	68	57	81
planning	147	79	55	**135**	31	47
testing	131	77	84	92	**102**	76
all	1011	543	526	**659**	460	535

- Second-best performance

Performance - without preprocessing

Number of instances solved within 10 minutes.

Family	total	CAQE picosat	CAQE minisat	RAReQS rareqs	GhostQ ghostq	DepQBF depqbf
eval2012r2	276	75	55	81	**124**	88
bomb	132	**91**	75	84	75	67
complexity	104	50	60	**75**	26	49
dungeon	107	46	22	**57**	45	44
hardness	114	**78**	58	15	57	8
planning	147	84	50	**146**	31	57
testing	131	54	25	36	**102**	57
all	1011	478	345	**494**	460	370

- Competitive performance

Performance - certification

Number of instances solved within 10 minutes and certified within another 10 minutes.

Solver	# solved	# verified	# unique
CAQE	428	340	146
DepQBF	312	239	44
virtual best	468	384	-

- Significant improvement in certification performance.

Conclusions

Contributions

- New CEGAR algorithm[1]
- Competitive performance
- Best certification performance

Questions

- Quantification as a theory in SMT solvers?

[1] Similar: Janota, Marques-Silva, "Solving QBF by Clause Selection", IJCAI'15