Efficiently Representing Existential Dependency Sets for Expansion-based QBF Solvers

Florian Lonsing and Armin Biere

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

MEMICS’08
November 14 - 16, 2008
Znojmo, Czech Republic
Quantified Boolean Formulae (QBF)
- propositional formula, $\forall/\exists$ quantification
- PSPACE-completeness: natural modelling language

Variable Dependencies in QBF
- two types: $\forall\exists$ and $\exists\forall$
- influence on decision procedures for QBF
- our focus: expansion-based solvers, case $\forall\exists$

Results
- given: syntactic dependency relation $D$ for case $\forall\exists$
- average-case compact representation for directed variant of $D$
  - equivalence relation on $\exists$-variables
  - efficient retrieval of $\exists$-dependencies for $\forall$-variables
- experimental results: benchmarks from QBF competitions 2005 - 2008
Quantified Boolean Formulae (QBF): $S_1 \ldots S_n \phi$

- prenex conjunctive normal form (PCNF), e.g. $\forall x_1 \exists x_2 x_3 \phi$
- propositional formula $\phi$ in CNF over variables $V = V_\forall \cup V_\exists$
- quantifier prefix $S_1 \ldots S_n$
  - scopes $S_i$, $q(S_i) \in \{\forall, \exists\}$: quantified variables
  - linear orderings: $\delta(S_1) = 1 < \ldots < \delta(S_n) = n$, $\delta(x) = \delta(S_i)$ if $x \in S_i$

Variable Dependencies in QBF

- $\delta(S_1) < \ldots < \delta(S_n)$: often pessimistic
- dependency computation in practice: optimality vs. efficiency
  - polynomial time: syntactic analysis of formula

Example

$$\forall x \exists y \left( \neg x \lor y \right) \land \left( x \lor \neg y \right)$$ is satisfiable

Value of $y$ depends on $x$: $x = \top \rightarrow y = \top$, $x = \bot \rightarrow y = \bot$
**Motivation**

**Universal Expansion:** \( \forall x \phi \equiv \phi[x/0] \land \phi[x/1] \)
- existential dependencies for \( x \in V_\forall : D(x) \subseteq \{ y \in V_\exists \mid \delta(x) < \delta(y) \} \)

**Computing \( D(x) \) via Syntactic Connection Relation**
- \( y, z \in V \): \( y \) locally connected to \( z \) if \( y, z \in C \) for clause \( C \in \phi \)
- inf.: \( y \in D(x) \) if \( x \) transitively connected to \( y \) via common clauses
- recursive computation: \( O(|\phi|) \) for one \( x \in V_\forall \)

**Goal: Avoiding Recomputation of Connection Relation**
- building a global connection relation wrt. common clauses
- idea: extract once from \( \phi \), exploiting shared parts for all \( x \in V_\forall \)
- compact representation and retrieval of \( D(x), |D(x)| \)
Towards a Directed Dependency Relation (1/3)

Definition (local dependence/connection)
For \( x, y \in V \): \( x \rightarrow_i y \iff q(y) = \exists \) and \( x, y \in C, C \in \phi \) and \( \delta(y) \geq i \).
Connecting sets of variables and clauses by refl. and trans. closure \( \rightarrow_i^* \).

"connection": write \( x \sim_i y \) if \( q(x) = q(y) = \exists \) and \( x \rightarrow_i^* y \).

Example
\[ \begin{array}{c}
A_2 \quad A_4 \quad A_6
\end{array} \]
\[ \begin{array}{c}
\forall_1 \quad \exists_2 \quad \forall_3 \quad \exists_4 \quad \forall_5 \quad \exists_6
\end{array} \]

- trans. edges not shown
- \( a_1 \rightarrow_1 e_6, e_6 \rightarrow_1 e_9 \)
- \( e_9 \rightarrow_1 e_6 \)
- \( a_1 \rightarrow_1^* e_7 \) by \( e_6, e_9, e_3 \)

(Application) For \( x \in V_\forall, i = \delta(x) : D(x) = \{y \in V_\exists \mid x \rightarrow_i^* y\} \).
Definition (equivalence relation)

For \( x, y \in V \): \( x \approx y \iff x = y \) and \( q(x) = \forall \) or \( \delta(x) = \delta(y) = i, q(x) = q(y) = \exists \) and \( x \sim_i y \). [\( x \)] denotes the class of \( x \).

Theorem (compatibility of \( \rightarrow_i^\ast \) with \( \approx \))

For \( x, y \in V \): \( x \rightarrow_i^\ast y \iff \forall x' \in [x], y' \in [y] : x' \rightarrow_i^\ast y' \).

Example (continued)

\[ \begin{array}{c}
\begin{array}{ccccccc}
\text{a1} & \text{e2} & \text{a4} & \text{a5} & \text{e6} & \text{a8} & \text{e9} \\
\text{e2} & \text{a4} & \text{a5} & \text{e6} & \text{a8} & \text{e9} & \text{e10} \\
\text{a5} & \text{e6} & \text{a8} & \text{e9} & \text{e10} & \text{a8} & \text{e10} \\
\end{array}
\end{array} \]

- Partition of scopes
- \( e_2 \approx e_3 \)
- \( e_3 \not\approx e_9 \)
- \( e_6 \not\approx e_7 \) since \( e_6 \not\approx 4 e_7 \)

(Application) For \( x \in V_\forall, i = \delta(x) : D(x) = \{ y \in V_\exists \mid [x] \rightarrow_i^\ast [y] \} \).
**Definition (directed dependence/connection)**

For $x, y \in V$: $[x] \rightsquigarrow^* [y] \iff \delta(x) \leq \delta(y)$ and $x \rightarrow^*_i y$ for $i = \delta(x)$.

**(Application)** For $x \in V_\forall$, $i = \delta(x)$: $D(x) = \{ y \in V_\exists \mid [x] \rightsquigarrow^* [y] \}$.

**Theorem (computing dependency sets)**

For $x \in V_\forall$, $i = \delta(x)$:

$D(x) = \{ y \in V_\exists \mid x \rightarrow^*_i y \} = \{ y \in V_\exists \mid [x] \rightarrow^*_i [y] \} = \{ y \in V_\exists \mid [x] \rightsquigarrow^* [y] \}$.

**Example (continued)**

- $\rightsquigarrow^*$ defined on classes
- $e_2 \rightsquigarrow^* e_9$, but $e_9 \nrightarrow^* e_2$
- dashed: transitive edges
- solid: transitive reduction
An Efficient Tree Representation for $\leadsto^*$

**Lemma**

*For $\leadsto^*$ on $V_\exists$, the transitive reduction $\leadsto$ can be represented as forest.*

**Connection Forest of a QBF**

- representation of global, shared connection relation for $V_\exists$
- for $y, z \in V_\exists$: edge $([y], [z]) \iff [y] \leadsto [z]$
- for $y, z \in V_\exists$: path from $[y]$ to $[z] \iff [y] \leadsto^* [z]$

**Augmented Connection Forest**

- additionally: set of “entry points” $H(x)$ for all $x \in V_\forall$
- $H(x)$ derived from clauses containing literals of $x$

**Computing $D(x)$ by Connection Forest**

1. collect descendant classes: $H^*(x) := \{[y] \mid [z] \leadsto^* [y], [z] \in H(x)\}$
2. collect members of descendants: $D(x) = \{z \mid z \in [y], [y] \in H^*(x)\}$
<table>
<thead>
<tr>
<th></th>
<th>QBFEVAL'05</th>
<th>QBFEVAL'06</th>
<th>QBFEVAL'07</th>
<th>QBFEVAL'08</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td>211</td>
<td>216</td>
<td>1136</td>
<td>3328</td>
</tr>
<tr>
<td>max. $</td>
<td>H^*(x)</td>
<td>$</td>
<td>797</td>
<td>5</td>
</tr>
<tr>
<td>avg. $</td>
<td>H^*(x)</td>
<td>$</td>
<td>19.51</td>
<td>1.21</td>
</tr>
<tr>
<td>max. $</td>
<td>D(x)</td>
<td>$</td>
<td>256535</td>
<td>9993</td>
</tr>
<tr>
<td>avg. $</td>
<td>D(x)</td>
<td>$</td>
<td>82055.87</td>
<td>4794.60</td>
</tr>
<tr>
<td>avg. $\frac{</td>
<td>H^*(x)</td>
<td>}{</td>
<td>D(x)</td>
<td>}$</td>
</tr>
<tr>
<td>$\approx_\exists$</td>
<td>3.08 %</td>
<td>3.95 %</td>
<td>2.20 %</td>
<td>7.37 %</td>
</tr>
</tbody>
</table>

- structured QBF formulae from QBF competitions 2005 - 2008
- comparing forest representation with $|D(x)|$
- number of successors $|H^*(x)|$ much smaller than $|D(x)|$
- line $\approx_\exists$: number of $\exists$-classes per $\exists$-variable in whole formula set
- compression by $\approx$: few, but large classes for $S_i, q(S_i) = \exists$
Conclusion

Variable Dependencies in QBF
- influence solver performance
- common approach: syntactic connection relation (connecting clauses)
- focus: expansion-based solvers, ∀∃ dependencies

Augmented Connection Forests
- directed version of connection relation, equivalence relation on $V_\exists$
- average-case compact representation
- sharing connection information between all $x \in V_\forall$
- computation of $D(x)$, $|D(x)|$ for all $x \in V_\forall$

Future Work
- dynamic vs. static version
- extension to ∃∀ dependencies
- combination with search-based solvers