Beyond SAT
QSAT: Quantified propositional logic

Marco Maratea, Wolfgang Faber

Ragionamento Automatico, Esercitazione III
2006

Outline

1. Beyond SAT : QSAT: Quantified SAT
   - Quantified Boolean formulas (QBFs) satisfiability
   - Applications of QBFs and QBF reasoning
   - Solving methods for QBFs
   - State of the art in QBF reasoning

Outline

The syntax of QBFs

$$\overbrace{Q_1 z_1 \cdots Q_n z_n}^{\text{prefix}} \; \overbrace{\phi(z_1, \ldots, z_n)}^{\text{matrix}} \qquad n \geq 0$$

- Every Q_i ($1 \leq i \leq n$) is a quantifier, either existential $\exists$ or universal $\forall$
- Every z_i is a Boolean variable
- ϕ is a Boolean formula over the set of variables $\{z_1, \ldots z_n\}$ using standard Boolean connectives and the constants $\bot$ and $\top$

Examples

$\forall y \exists x.(x \leftrightarrow y)$
forall values of y, is there a value for x such that $x \leftrightarrow y$ is true?

$\exists x \forall y.(x \leftrightarrow y)$
Is there a value for x such that for all values of y, $x \leftrightarrow y$ is true?

$\exists x_1 \forall y \exists x_2.(x_1 \wedge y) \rightarrow x_2$
Is there a value for x_1 such that for all values of y, there exists a value of x_2, such that x_1 and y imply x_2?

$\exists x_1 \exists x_2 \exists x_3.(x_1 \wedge x_2) \leftrightarrow x_3$
Is the Boolean formula $(x_1 \wedge x_2) \leftrightarrow x_3$ satisfiable?

$\forall y_1 \forall y_2.\neg(y_1 \wedge y_2) \leftrightarrow (\neg y_1 \vee \neg y_2)$
Is the Boolean formula $\neg(y_1 \wedge y_2) \leftrightarrow (\neg y_1 \vee \neg y_2)$ a tautology?

The semantics of QBFs

Truth of a QBF φ

- If the prefix is empty then φ's truth is defined according to the semantics of Boolean logic
- If $\varphi = \exists x \psi$ then φ is true iff $(\varphi_x \vee \varphi_{\neg x})$ is true
- If $\varphi = \forall y \psi$ then φ is true iff $(\varphi_y \wedge \varphi_{\neg y})$ is true

Definition of φ_x and $\varphi_{\neg x}$

Given $\varphi = Q_1 z_1 \cdots Q_n z_n \phi(z_1, \ldots, z_n)$ and $x = z_i$ then

- $\varphi_x = Q_1 z_1 \cdots Q_{i-1} z_{i-1} Q_{i+1} z_{i+1} \cdots Q_n z_n \phi[\top / z_i]$
- $\varphi_{\neg x} = Q_1 z_1 \cdots Q_{i-1} z_{i-1} Q_{i+1} z_{i+1} \cdots Q_n z_n \phi[\bot / z_i]$

Reasoning and complexity (I)

$$Q_1 z_1 \cdots Q_n z_n \phi(z_1, \ldots, z_n) \qquad n \geq 0$$

$$\bigtriangledown$$

$$Q_1 Z_1 \cdots Q_k Z_k \phi(Z_1, \ldots, Z_k) \qquad k \geq 0$$

- Every Q_i ($1 \leq i \leq k$) is a quantifier, sucht that $Q_i \neq Q_{i+1}$
- $Z_1, \ldots, Z_k$ define a partition of Z
- $k - 1$ is the number of alternations

Reasoning and complexity (II)

Let φ be an expression of the form $Q_1 Z_1 \cdots Q_k Z_k \phi(Z_1, \ldots, Z_k)$

Problems

QSAT	Is φ true?
QSAT_k	Is φ true with k known a priori?

Complexity

QSAT	is the prototypical PSPACE-Complete problem
QSAT_k	is $\Sigma_k P$-Complete if $Q_1 = \exists$ and $\Pi_k P$-Complete if $Q_1 = \forall$

Outline

Applications: overview

Theory & Practice

In theory every problem in PSPACE can be encoded efficiently into some QBF reasoning problem. In practice QSAT solvers must be competitive w.r.t. specialized algorithms

Domains

- Equivalence of partially specified circuits
- Conformant/Conditional planning
- Symbolic reachability
- Games, reasoning about knowledge, ...

Equivalence of circuits

Setting

$\varphi_i(X)$ $(1 \leq i \leq m)$ is the i-th output of the specification φ over the inputs X
$\psi_i(X, Y)$ $(1 \leq i \leq m)$ is the i-th output of the circuit ψ over the inputs X and the black box variables Y

Problem

Does the circuit ψ satisfy the specification φ?

QBF encoding

If the QBF $\exists X \forall Y \bigvee_{i=1}^{m} \varphi_i(X) \oplus \psi_i(X, Y)$ is true then ψ does not fullfill the specification φ

Conformant planning

Setting

F is the set of fluents, A is the set of actions
$I(F)$, $G(F)$ encode the set of initial and goal states, resp.
$\tau(F, A, F')$ is the set of possible transitions

Problem

Given a non-deterministic action domain, is there a sequence of actions that is guaranteed to achieve the goal (in k steps)?

QBF encoding

$$\exists A_0 \cdots A_{k-1} \forall F_0 \cdots \forall F_k (I(F_0) \wedge \bigwedge_{t=0}^{k-1} \tau(F_t, A_t, F_{t+1}) \rightarrow G(F_k))$$

Symbolic reachability

Setting

Vertices are set of boolean variables, and $\tau(S, T)$ is a Boolean formula which is true when there is an edge between S and T

Problem

Is there a walk between two (sets of) states?

QBF encoding

$$\begin{cases} \varphi^i(S,T) = \exists Z^i \forall y^i \exists S^i \exists T^i (\, (y^i \rightarrow (S \leftrightarrow S^i \wedge Z^i \leftrightarrow T^i)) \wedge \\ \qquad (\neg y^i \rightarrow (Z^i \leftrightarrow S^i \wedge T \leftrightarrow T^i)) \wedge \\ \qquad \varphi^{i-1}(S^i, T^i)) \\ \varphi^0 = \tau(S,T) \end{cases}$$

Outline

Basics

Input formula

$\varphi = Q_1 Z_1 \cdots Q_k Z_k \phi(Z_1, \ldots, Z_k)$ with $k \geq 0$ where ϕ is a Boolean formula in conjunctive normal form

More notation

- $level(z)$ denotes the value i s.t. $z \in Z_i$
- $|l|$ denotes the variable occurring in l
- $level(l) = level(|l|)$ is the level of a literal l

DLL-based search algorithm

DLL-QSAT(φ)
1 **if** $\varphi = \emptyset$ **then return** TRUE
2 **if** $\emptyset \in \varphi$ **then return** FALSE
3 **if** "l is unit in φ" **then**
  return DLL-QSAT(φ_l)
4 **if** $\varphi = \exists x \psi$ **then**
  return DLL-QSAT(φ_x) **or**
    DLL-QSAT($\varphi_{\neg x}$)
 else
  return DLL-QSAT(φ_x) **and**
    DLL-QSAT($\varphi_{\neg x}$)

φ_x is $assign(x, \phi)$!

Unit literal

A literal l is unit in φ iff it is the only existential in some clause $c \in \phi$ and all the universal literals $l' \in c$ are s.t. $level(l') > level(l)$

An example about search

$$\exists x_1 \forall y \exists x_2 \exists x_3 \{\{\bar{x}_1, \bar{y}, x_2\}, \{x_1, \bar{y}, \bar{x}_3\}, \{\bar{y}, \bar{x}_2\}, \{y, x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

An example about search

$$\exists x_1 \forall y \exists x_2 \exists x_3 \{\{\bar{x}_1, \bar{y}, x_2\}, \{x_1, \bar{y}, \bar{x}_3\}, \{\bar{y}, \bar{x}_2\}, \{y, x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

$x_1 = 0$ OR node

$$\forall y \exists x_2 \exists x_3 \{\{\bar{y}, \bar{x}_3\}, \{\bar{y}, \bar{x}_2\}, \{y, x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

An example about search

$$\exists x_1 \forall y \exists x_2 \exists x_3 \{\{\bar{x}_1, \bar{y}, x_2\}, \{x_1, \bar{y}, \bar{x}_3\}, \{\bar{y}, \bar{x}_2\}, \{y, x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

$x_1 = 0$

OR node

$$\forall y \exists x_2 \exists x_3 \{\{\bar{y}, \bar{x}_3\}, \{\bar{y}, \bar{x}_2\}, \{y, x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

$y = 0$

AND node

$$\exists x_2 \exists x_3 \{\{x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

An example about search

$$\exists x_1 \forall y \exists x_2 \exists x_3 \{\{\bar{x}_1, \bar{y}, x_2\}, \{x_1, \bar{y}, \bar{x}_3\}, \{\bar{y}, \bar{x}_2\}, \{y, x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

$x_1 = 0$ — OR node

$$\forall y \exists x_2 \exists x_3 \{\{\bar{y}, \bar{x}_3\}, \{\bar{y}, \bar{x}_2\}, \{y, x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

$y = 0$ — AND node

$$\exists x_2 \exists x_3 \{\{x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

$x_2 = 1$

$$\{\}$$

✓ solution

An example about search

$$\exists x_1 \forall y \exists x_2 \exists x_3 \{\{\bar{x}_1, \bar{y}, x_2\}, \{x_1, \bar{y}, \bar{x}_3\}, \{\bar{y}, \bar{x}_2\}, \{y, x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

$x_1 = 0$

OR node

$$\forall y \exists x_2 \exists x_3 \{\{\bar{y}, \bar{x}_3\}, \{\bar{y}, \bar{x}_2\}, \{y, x_2, \bar{x}_3\}, \{x_2, x_3\}\}$$

$y = 0$ AND node $y = 1$

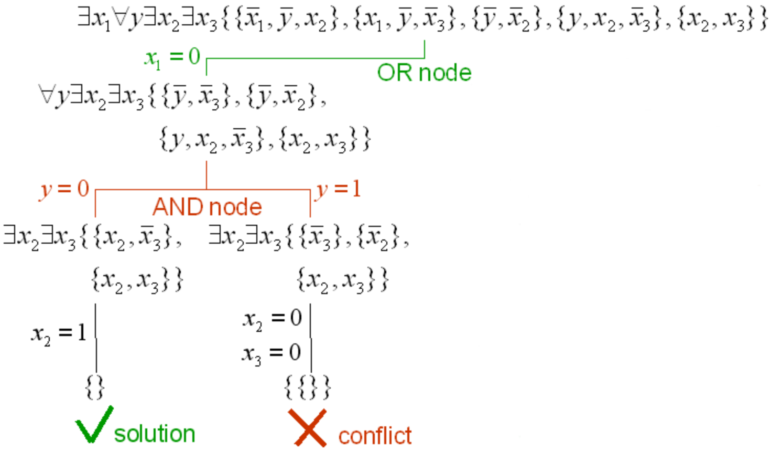
$$\exists x_2 \exists x_3 \{\{x_2, \bar{x}_3\}, \{x_2, x_3\}\} \qquad \exists x_2 \exists x_3 \{\{\bar{x}_3\}, \{\bar{x}_2\}, \{x_2, x_3\}\}$$

$x_2 = 1$

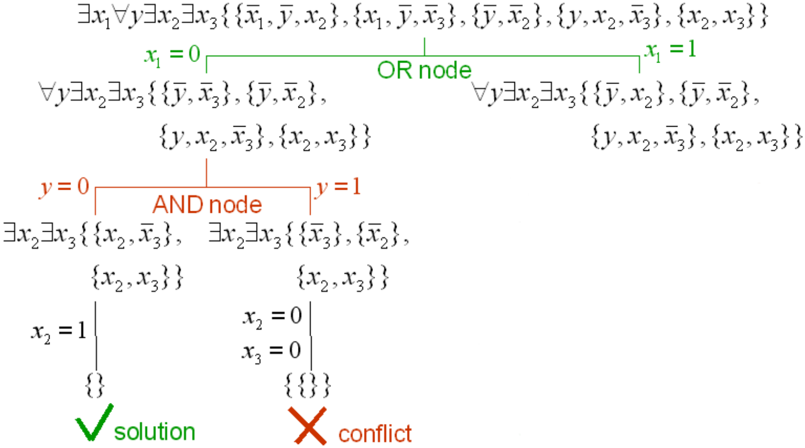
$$\{\}$$

✓ solution

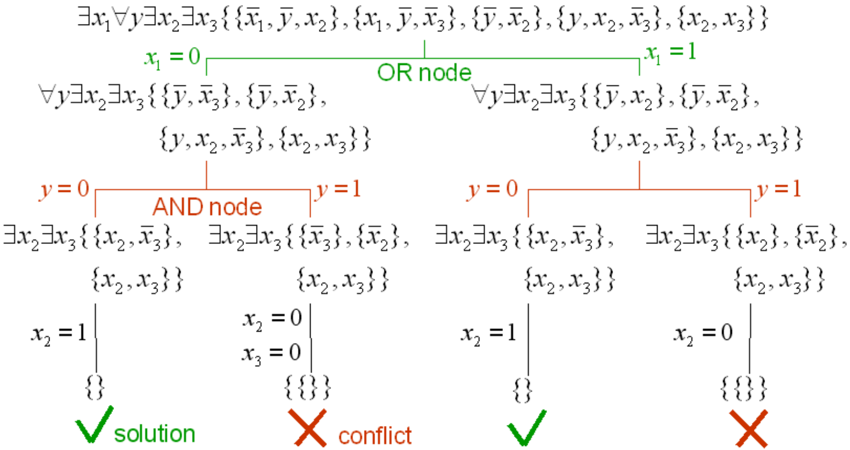
An example about search



An example about search



An example about search



Backjumping

Problem

Time spent visiting parts of the search space in vain because some choices may not be responsible for the result of the search

Solution

1. for each node of the search tree, compute a subset (called “reason”) of the assigned variables which are responsible for the current result; and
2. while backtracking, skip nodes which do not belong to the reason for the discovered conflicts/solutions:
   - CBJ Conflict backjumping
   - SBJ Solution backjumping

Learning

Problem

CBJ and SBJ may do the same wrong choices in different branches

Solution

Learn (some of) the reasons computed during backjumping:

CBJ $\Rightarrow$ conflict learning of "nogoods" (as in SAT)

SBJ $\Rightarrow$ solution learning of "goods" (specific of QBF):

- a good is a term (conjunction of literals)
- goods are to be treated as if in disjunction with the matrix

Learning in practice

Problem

The number of computed reasons can be exponential, thus it can be practically unfeasible to learn all the reasons

Solution

We must introduce criteria for:

- deciding when to store a computed reason ; and
- deciding when to forget a stored reason (e.g., periodically clean up the learned constraints storage)

Alternative approaches

Resolution & Expansion

- eliminate existential variables using resolution
- expand universal variables $\forall x F(x) = F_x \wedge F_{\neg x}$

Symbolic algorithms

- use ZDDs to represent clauses
- implement resolution and expansion as ZDD operations

Skolemization

- based on skolemization and symbolic representation (with OBDD) of the constraints about Skolem functions
- implements and interleaves various strategies (e.g., expansion, search, ...)

Outline

Short history of QBF evaluations

S. Margherita '03: 11 solvers

Vancouver '04: 16 solvers, 2 generators

St. Andrews 2005: 13 solvers, 3 generators

Seattle 2006: first competitions!

QDIMACS input format

Ex: $\forall x_1 \exists x_2 \forall x_3 \exists x_4 x_5 \ (x_1 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge$
$(x_1 \vee \neg x_4 \vee x_5) \wedge (\neg x_1 \vee x_2 \vee x_5) \wedge (x_1 \vee \neg x_3 \vee x_4 \vee \neg x_5) \wedge$
$(\neg x_1 \vee x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee x_5) \wedge (x_1 \vee \neg x_4) \wedge (x_3 \vee \neg x_2 \vee x_1)$

Example

c This is a CNF in QDIMACS
c
p cnf 5 9
a 1 0
e 2 0
a 3 0
e 4 5 0
1 3 4 0
-1 3 4 0
1 -4 5 0
-1 2 5 0
1 -3 4 -5 0
-1 3 -4 0
-1 -2 -3 -5 0
1 -4 0
3 -2 1 0

QDIMACS input format in BNF grammar

BNF grammar

$< input > ::= < preamble > < prefix > < matrix > EOF$

$< preamble > ::= \ [< commentlines >] < problemline >$
$< commentlines > ::= < commentline > < commentlines > | < commentline >$
$< commentline > ::= \ c \ < text > EOL$
$< problemline > ::= \ p \ cnf \ < pnum > < pnum > EOL$

$< prefix > ::= \ [< quantsets >]$
$< quantsets > ::= < quantset > < quantsets > | < quantset > EOL$
$< quantset > ::= < quantifier > < atomset > \ 0$
$< quantifier > ::= \ e \mid a$
$< atomset > ::= < pnum > < atomset > \ | \ < pnum >$

$< matrix > ::= < clauselist >$
$< clauselist > ::= < clause > < clauselist > | < clause >$
$< clause > ::= < literal > < clause > | < literal > 0$
$< literal > ::= < num >$

$< text > ::= \ A \ sequence \ of \ non - special \ ASCII \ characters$
$< pnum > ::= \ A \ signed \ integer \ greater \ than \ 0$
$< num > ::= \ A \ signed \ integer \ different \ from \ 0$

Challenges and ongoing work

Hot topics

- certificates of truth/falsity (done?!)
- heuristics
- search vs symbolic vs skolemization

Stay tuned:

- `www.qbflib.org`
- `www.qbflib.org/qbfeval`