A Problem That Is Complete for PSPACE (Polynomial Space)

BY

TEJA SUDHA GARIGANTI
A Problem That Is Complete for $\text{PS} : \text{QBF}$

1. PS-Completeness.
2. The TQBF Problem.
3. Evaluating QBF.
4. TQBF is PSPACE-Complete.
5. Example: GG is PS-complete.
We define a problem $P$ to be complete for PS (PS-Complete) if:

1. $P$ is in PS.
2. All Languages $L$ in PS are polynomial-time reducible to $P$. 

**PSPACE:**

If we denote by $\text{SPACE}(t(n))$, the set of all problems that can be solved by Turing machines using $O(t(n))$ space for some function $t$ of the input size $n$, then we can define PSPACE as

$$\text{PSPACE} = \bigcup_{k \in \mathbb{N}} \text{SPACE}(n^k).$$
Theorem: Suppose $P$ is in $PSPACE$-Complete problem. Then:

a) If $P$ is in $P$, then $P = PSPACE$.
b) If $P$ is in $NP$, then $NP = PSPACE$.

$P$ is no larger than $PSPACE$

**Example:**

An algorithm that uses 30 tape cells must use at least 30 time steps in $P$.

An algorithm that uses 30 tape cells may use many more steps in $PSPACE$.
The TQBF PROBLEM

**True Quantified Boolean Formula** is an example of **PSPACE-Complete** which involves a generalization of **SAT** problem.

- **BOOLEAN FORMULA:**
  - Boolean Variables say 0,1
  - Boolean Operations say \(\land, \lor\) and \(\neg\)

- **QUANTIFIERS:**
  - Universal Quantifier: \(\forall\) (for all)
  - Existential Quantifier: \(\exists\) (there exists)

- Boolean formulas with quantifiers are called **Quantified Boolean Formulas**

\(\forall x \, \phi\) means, for every value for the variable \(x\), the statement \(\phi\) is true.

\(\exists x \, \phi\) means for some value for the variable \(x\), the statement \(\phi\) is true.

**Example:** \(\phi = \forall x \, \exists y \left[ (x \lor y) \land (\neg x \lor \neg y) \right]\)
Statements may contain several quantifiers.

Order of the quantifiers is important.

A quantifier may appear anywhere in the statement.

Example: If we consider the natural numbers, the statement
\[ \forall x \exists y \ [y > x] \] means
that the successor \( x+1 \) of every natural number
\( x \) is greater than the number itself. Obviously, the statement is true. However,
the statement
\[ \exists y \forall x \ [y > x] \] obviously is false.

In preceding cases we have considered natural numbers, but if we took real
numbers instead, the existential statement would be true.
Evaluate the QBF of 

\((\forall x) ((\exists y)(xy) + (\forall z)(\neg x + z))\)

for \(E_0(x=0)\) and \(E_1(x=1)\).

Note the QBF is of the form \((\forall x)(E)\).
PROOF:
∃x (0y) + ∀z (¬0 + z)
we have 2 expressions which are connected by the OR:
∃x (0y) and ∀z (¬0 + z)

Case1: E₀(x=0), ∃x (0y)

- Substitute y=0 and y=1 in sub expression 0y, check at least one of them has the value 1.
  Both 0 ∧ 0 and 0 ∧ 1 have the value 0, so this expression has value 0.
- By substituting z=0 and z=1 in (¬0 + z), 1 ∨ 0 and 1 ∨ 1, it has value 1

Case2: E₁(x=1), ∃x (1y) + ∀z (¬1 + z)

- Substitute y=0 and y=1, it has value 1. so, we conclude entire equation has value 1.

(∀x) (((∃y)(xy)) + (∀z) (¬x + z))
has a value ‘1’.
TQBF is PSPACE-Complete

To show that TQBF is in PSPACE we give polynomial space algorithm deciding TBQF

$T=$ “On input $(\varnothing)$, a fully quantified Boolean formula:
1. If $\varnothing$ contains no quantifiers, then it is an expression with only constants, so evaluate $\varnothing$ and accept if it is true; otherwise reject.
2. If $\varnothing$ equals $\exists x \, \Psi$, recursively call $T$ on $\Psi$, first with 0 substituted for $x$ and then with 1 substituted for $x$. If either result is accept then accept; otherwise reject.
3. If $\varnothing$ equals $\forall x \, \Psi$, recursively call $T$ on $\Psi$, first with 0 substituted for $x$ and then with 1 substituted for $x$. If either result is accept then accept; otherwise reject.”

This can be clearly explained by formula-game or GG
The following algorithm decides whether Player 1 has a winning strategy in instances of generalized geography; it decides GG.

M="On input <G,b>, where G is a directed graph and b is a node of G:
1. If b has outdegree 0, reject, because player 1 loses immediately.
2. Remove node b and all arrows touching it to get a new graph G₁.
3. For each of the nodes b₁,b₂,...,bₖ that b originally pointed at, recursively call M on <G₁,bi>.
4. If all of these accept, Player 2 has a winning strategy in the original game, so reject. Otherwise, Player 2 doesn’t have a winning strategy, so Player 1 must; therefore accept."

The only space required by this algorithm is for storing the recursion stack. Each level of the recursion adds a single node to the stack, and atmost m levels occur, where m is the number of nodes in G.
The reduction maps the formula

$$\varnothing = \exists x_1 \forall x_2 \exists x_3..\exists x_\kappa [\varphi]$$

**CONCLUSION:**

These problems will be solved with introducing new ideas
THANK YOU