MODAL LOGIC SATISFIABILITY PROBLEM AND ITS COMPLEXITY

Basudeb Roy

AGENDA

Introduction

Basic Definition

PSPACE

Forceing binary trees in modal logic

PSPACE algorithm

Conclusion

INTRODUCTION

- Expressing and proving constraints on mathematical models
- Modal logic is one of them.
- Obvious question is about computability and complexity.
- Investigate the computability of satisfiability problem
- Modal Satisfiability problem belongs to PSPACE.

BASIC DEFINITION

- Modal Logic:-
  - A set of W of states or word
  - R $\subseteq W \times W$.
  - F = (W, R)
  - M = F, V
  - $V : W \rightarrow 2^P$
- P and Q propositional variables.
- Formula:- P| $\neg P \mid P \wedge Q \mid P \vee Q \mid P \rightarrow Q \mid \Box P \mid \diamond P$
- QBF:-
  - Generalised Boolean formula
  - existential quantifiers
  - Universal quantifiers
  - Both applied to each variables
- Example: - $\forall x\ \exists y\ \exists z\ ((x \vee z) \wedge y)$

PSPACE

- Class of problem
    - Solvable by deterministic Turing mechine
    - Polynomial space
- Relevent complexity class
- Intuitively PSPACE-complete problems are the haredest problem.
- Some PSPACE-complete problem:
    - Given a regular expression, does it match all possible strings, or is there a string it doesn’t match?
    - Given a formula with no free variables, check whether it is true.
- Modal satisfiability problem is belongs to PSPACE.

FORCING BINARY TREE

- For any natural number m, devise $\varphi^\beta(m)$:
  - Size of $\varphi^\beta(m)$ is polynomial in m
  - If satisfied in M at a node w_0, then
    - Submodel generated by w_0 contains isometric copy of binary tree of depth m.
- binary branching tree of depth m contains 2^m nodes
- smallest satisfying model of φ^β (m) is exponential in $|\varphi^\beta(m)|$
- $\varphi^\beta(m)$ will be constructed out of the following variables:
  - q_1, q_2, q3, …., q_m
  - $P_1, p_2, p_3, \ldots, p_m$
- q_i used to mark the level.
- every possible combination of truth values for p_i's will be realized at some node

MACROS:-

- $B_i := q_i \rightarrow (\diamond(q_{i+1} \land p_{i+1}) \land \diamond(q_{i+1} \land \neg p_{i+1}))$
    - q_i's to mark the levels
    - force a branching to occur at level i
- $S(p_i, \neg p_{i+1}) := (p_i \rightarrow \Box p_i) \land (\neg p_i \rightarrow \neg \Box p_i)$
    - sends the truth values assigned to pi and its negation one level down.
    - once B_i has forced a branching in the model
        - ensures that these newly set truth values are sent further down the tree

- ultimately we want them to reach the leaves

DEFINE $\Phi^B(M)$

- Conjunction of the formula listed below:-
  - q_0
  - $\Box^m(q_i \wedge_{i \neq j} q_j)$ $(0 \geq i \geq m)$
  - $B_0 \wedge \Box B_1 \wedge \Box^2 B_2 \wedge \Box^3 B_3 \wedge \cdots \wedge \Box^{m-1} B_{m-1}$
  - $\Box S(p_1, \neg p_1) \wedge \Box^2 S(p_1, \neg p_1) \wedge \cdots \wedge \Box^{m-1} S(p_1, \neg p_1)$
    $\wedge \Box^2 S(p_2, \neg p_2) \wedge \cdots \wedge \Box^{m-1} S(p_2, \neg p_2)$
    $\wedge \Box^{m-1} S(p_{m-1}, \neg p_{m-1})$

- Theorem 1:- Modal logic lacks the polysize model property.

DEFINITIONS:-

- Closed:- A set of formulas Σ is said to be closed if it is closed under subformulas and single negations.
- Closure(Γ):- If Γ is a set of formulas, then Cl(Γ), the closure of Γ, is the smallest closed set of formulas containing Γ. Note that if Γ is finite then so is Cl(Γ)
- Hintikka Set:- Let Σ be a subformula closed set of formulas. A Hintikka set Hover Σ is a maximal subset of Σ that satisfies the following conditions:
  1. $\perp$ not belongs to H.
  2. If $\neg\varphi \in \Sigma$, then $\neg\varphi \in H$ iff $\varphi \notin H$.
  3. If $\varphi \wedge \psi \in \Sigma$, then $\varphi \wedge \psi \in \Sigma$ iff $\varphi \in H$ and $\psi \in H$.

PSPACE ALGORITHM WITNESS(H, Σ)

- Require: H, Σ
  Ensure: Boolean
- if (H is a Hintikka set over Σ and for each subformula ⋄ψ ∈ H there is a set of formula I ∈ H⋄ψ such that witness(I, Cl(Dem(H, ⋄ψ))) then
    - return true
- else
    - return false

SUMMARY

- Modal satisfiability problem is PSPACE-complete.
- We left the proof the algorithm.

THANK YOU

Basudeb Roy
basudebroy82@gamil.com