CS 7220 – Computational Complexity and Algorithm Analysis

Spring 2016

Proof of Cook’s Theorem

Pascal Hitzler
Data Semantics Laboratory
Wright State University, Dayton, OH
http://www.pascal-Hitzler.de

Contents

1. SAT is in NP
2. SAT is NP-hard

SAT is in NP

$$F = \left(\bigwedge_{i=1}^{n} \left(\bigvee_{j=1}^{m} L_{i,j} \right) \right)$$

- **Non-deterministically pick a truth assignment.**
  Represent this in a look-up table.
  [linear in number of literals]
- **Check if truth assignment satisfies F.**
  [quadratic – because of comparison of input with table entries]

- **Formally, we need to do this on a TM – the encoding is a bit unwieldy, but straightforward.**

Contents

1. SAT is in NP
2. SAT is NP-hard

Idea

- Give a logical formula which transforms computations of a TM M with input string u into a formula f(u) s.t.

  u is accepted iff f(u) is satisfiable.

- + show that transformation is polynomial.

- [f(u) doesn’t have to be in CNF because of Exercise 30]

Encoding

ND TM M:

- states: $q_0,\ldots,q_m$
- alphabet: $B=a_0,\ldots,a_t$
- accepting state: q_m
- rejecting state: q_{m-1} (only one)

p(n) polynomial which is upper bound to number of computations

Boolean variables:

- $Q_{i,k}$ M is in state q_i at time k
- $P_{j,k}$ Tape head is in position j at time k
- $S_{j,r,k}$ Tape position j contains symbol a_r at time k

SAT is NP-hard: Clauses *i* & *ii*

	Clause	Conditions	Interpretation
i)	State $\bigvee_{i=0}^{m} Q_{i,k}$	$0 \leq k \leq p(n)$	For each time k, M is in at least one state [p(n) clauses, m literals each]
	$\neg Q_{i,k} \vee \neg Q_{i',k}$	$0 \leq i < i' \leq m$ $0 \leq k \leq p(n)$	M is in at most one state at any time [$O(m^2) \times p(n)$ clauses]
ii)	Tape head $\bigvee_{j=0}^{p(n)} P_{j,k}$	$0 \leq k \leq p(n)$	For each time k, the tape head is in at least one position [p(n) clauses, p(n) literals each]
	$\neg P_{j,k} \vee \neg P_{j',k}$	$0 \leq j < j' \leq p(n)$ $0 \leq k \leq p(n)$	…and at most one position [$O(p(n)^3)$ clauses]

SAT is NP-hard: Clause *iii*

	Clause	Conditions	Interpretation
iii)	Symbols $\bigvee_{r=0}^{t} S_{j,r,k}$	$0 \leq j \leq p(n)$ $0 \leq k \leq p(n)$	For each time k and position j, position j contains at least one symbol [$p(n)^2$ clauses, t literals each]
	$\neg S_{j,r.k} \vee \neg S_{j,r',k}$	$0 \leq j \leq p(n)$ $0 \leq r < r' \leq t$ $0 \leq k \leq p(n)$	… and at most one symbol [$O(t^2) \times p(n)^2$ clauses]

SAT is NP-hard: Clauses *iv* & *v*

	Clause	Interpretation
iv)	Initialization $Q_{0,0}$	Begin in state 0
	$P_{0,0}$	…reading leftmost tape cell (position 0)
	$S_{0,0,0}$	…which contains a blank (symbol 0)
	$S_{1,r1,0}$	The next n symbols contain the input string,
	$S_{2,r2,0}$	which we'll denote $a_{r1}, a_{r2}, \ldots a_{rn}$
	…	
	$S_{n,rn,0}$	
	$S_{n+1,0,0}$	And the rest of the tape contains blanks…
	…	
	$S_{p(n),0,0}$	… for the entire accessible portion
v)	Final state $Q_{m,p(n)}$	The computation ends in q_m – the accepting state

A computation that satisfies all of these clauses still doesn't necessarily follow the rules of the machine, M.

Each state/symbol/position after time 0 must be obtained from the transition rules of M.

Tape Consistency

	Clause	Conditions	Interpretation
vi)	Tape Changes $\neg S_{j,r,k} \vee P_{j,k} \vee S_{j,r,k+1}$	$0 \leq j \leq p(n)$ $0 \leq r \leq t$ $0 \leq k \leq p(n)$	Symbols not at the position of the tape head are unchanged [$p(n)^2 \times t$ clauses]

WRIGHT STATE UNIVERSITY

Converting rules in δ to clauses

$$\neg Q_{i,k} \vee \neg P_{j,k} \vee \neg S_{j,r,k} \vee Q_{i',k+1}$$

If none of these are satisfied, then we are in state q_i and position j scanning symbol a_r at time k

In that case, the next state must be $Q_{i'}$ or the clause is not satisfied.

For each $\delta(q_i, a_r) = [q_{i'}, ?, ?]$

Same thing for tape symbols

$$\neg Q_{i,k} \vee \neg P_{j,k} \vee \neg S_{j,r,k} \vee S_{j,r',k+1}$$

If none of these are satisfied, then we are in state Q_i and position P_j scanning symbol S_r at time k

In that case, the next symbol at position j must be $S_{r'}$ or the clause is not satisfied.

For each $\delta(q_i, a_r) = [?, a_{r'}, ?]$

Same thing for tape head position

$$\neg Q_{i,k} \vee \neg P_{j,k} \vee \neg S_{j,r,k} \vee P_{j+n(d),k+1}$$

If none of these are satisfied, then we are in state Q_i and position P_j scanning symbol S_r at time k

In that case, the tape head will move either one position left or one position right.

Where n(L) = -1, and n(R) = +1
For each $\delta(q_i, a_r)$ = [?, ?, L/R]

Consistency

The conjunction of these three clause types ensures that if we are in a certain state, reading a particular symbol at a particular time, we must be in the right configuration, according to δ in the following time step.

These are machine dependent.

Consistency clauses

Consistency clauses are constructed for every time, state, tape head position and tape symbol.

However, if we are scanning position 0 and attempt to move left, we go directly to the rejecting state.

Hey, wait a minute!

We've been talking like there is only one transition for each state/symbol pair, but this is a non-deterministic Turing machine, right?

Let trans(i, j, r, k) be the disjunction of all the consistency clause sets for i, j, r, k. The resulting clause ensures that we are in some valid configuration following each transition.

And now we're done

Clause		Interpretation
vi)	Halted	
	$\neg Q_{i,k} \vee \neg P_{j,k} \vee \neg S_{j,r,k} \vee Q_{i,k+1}$	same state
	$\neg Q_{i,k} \vee \neg P_{j,k} \vee \neg S_{j,r,k} \vee P_{j,k+1}$	same tape head position
	$\neg Q_{i,k} \vee \neg P_{j,k} \vee \neg S_{j,r,k} \vee S_{j,r,k+1}$	same symbol at position r

For all appropriate j, r, k, and $i = q_{m-1}$,
and $i = q_m$

What we've done so far…

We've defined a set of wff that are satisfiable if (and only if) some computation of ND TM M leads to an accepting final state.

Polynomial transformation?

Can the formula be created from any NDTM M *in polynomial time?*

- **The values *m* and *t* are independent of the size of the input string. They <u>don't grow with *n*</u>.**
- **The number of clauses is polynomial in *p*(*n*).**

qed