Circuit satisfiability and Cook-Levin Theorem

Lecture 24

Friday, May 1, 2020

LaTeXed: January 19, 2020 04:28

Recap

NP: languages that have non-deterministic polynomial time algorithms

Recap

NP: languages that have non-deterministic polynomial time algorithms

A language L is **NP-Complete** iff

- L is in **NP**
- for every L' in **NP**, $L' \leq_P L$

Recap

NP: languages that have non-deterministic polynomial time algorithms

A language L is **NP-Complete** iff

- L is in **NP**
- for every L' in **NP**, $L' \leq_P L$

L is **NP-Hard** if for every L' in **NP**, $L' \leq_P L$.

Recap

NP: languages that have non-deterministic polynomial time algorithms

A language L is **NP-Complete** iff

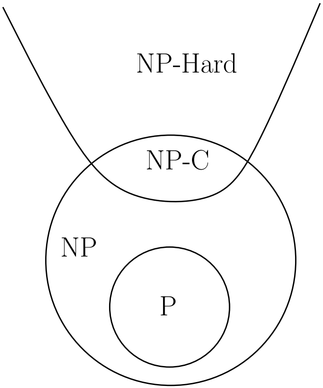
- L is in **NP**
- for every L' in **NP**, $L' \leq_P L$

L is **NP-Hard** if for every L' in **NP**, $L' \leq_P L$.

Theorem (Cook-Levin)

SAT *is* **NP-Complete**.

Pictorial View



P and NP

Possible scenarios:

1. $\mathbf{P} = \mathbf{NP}$.
2. $\mathbf{P} \neq \mathbf{NP}$

P and NP

Possible scenarios:

1. $\mathbf{P} = \mathbf{NP}$.
2. $\mathbf{P} \neq \mathbf{NP}$

Question: Suppose $\mathbf{P} \neq \mathbf{NP}$. Is every problem in $\mathbf{NP} \setminus \mathbf{P}$ also **NP-Complete**?

P and NP

Possible scenarios:

1. **P** $=$ **NP**.
2. **P** $\neq$ **NP**

Question: Suppose **P** $\neq$ **NP**. Is every problem in **NP** $\setminus$ **P** also **NP-Complete**?

Theorem (Ladner)

If **P** $\neq$ **NP** *then there is a problem/language* $X \in$ **NP** $\setminus$ **P** *such that* X *is not* **NP-Complete**.

NP-Complete Problems

Previous lectures:

- **3**-SAT
- Independent Set
- Hamiltonian Cycle
- **3**-Color

Today:

- Circuit SAT
- SAT

Important: understanding the problems and that they are hard.

Proofs and reductions will be sketchy and mainly to give a flavor

Part I

Circuit SAT

Circuits

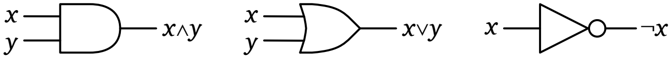


Figure 10.1. An AND gate, an OR gate, and a NOT gate.

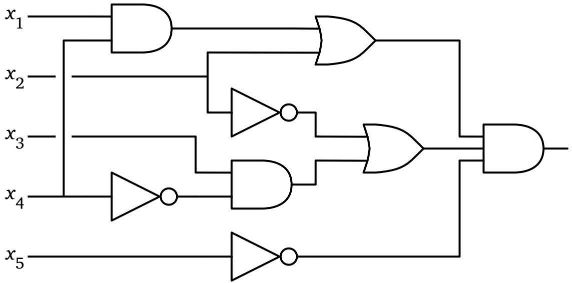
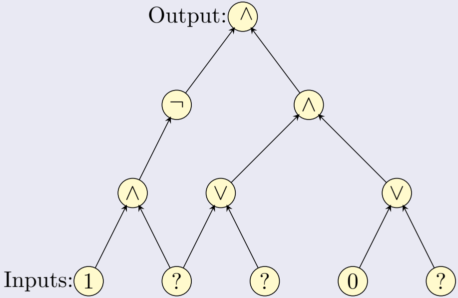


Figure 10.2. A boolean circuit. Inputs enter from the left, and the output leaves to the right.

Circuits

Definition

A circuit is a directed *acyclic* graph with



1. Input vertices (without incoming edges) labelled with **0**, **1** or a distinct variable.
2. Every other vertex is labelled $\vee$, $\wedge$ or $\neg$.
3. Single node output vertex with no outgoing edges.

CSAT: Circuit Satisfaction

Definition (Circuit Satisfaction (CSAT).)

Given a circuit as input, is there an assignment to the input variables that causes the output to get value $\mathbf{1}$?

CSAT: Circuit Satisfaction

Definition (Circuit Satisfaction (**CSAT**).)

Given a circuit as input, is there an assignment to the input variables that causes the output to get value $\mathbf{1}$?

Claim

CSAT *is in* **NP**.

1. Certificate: Assignment to input variables.
2. Certifier: Evaluate the value of each gate in a topological sort of DAG and check the output gate value.

Circuit SAT vs SAT

CNF formulas are a rather restricted form of Boolean formulas.

Circuits are a much more powerful (and hence easier) way to express Boolean formulas

Circuit SAT vs SAT

CNF formulas are a rather restricted form of Boolean formulas.

Circuits are a much more powerful (and hence easier) way to express Boolean formulas

However they are equivalent in terms of polynomial-time solvability.

Theorem

SAT $\leq_P$ **3SAT** $\leq_P$ **CSAT**.

Theorem

CSAT $\leq_P$ **SAT** $\leq_P$ **3SAT**.

Converting a CNF formula into a Circuit

3SAT $\leq_P$ CSAT

Given $3\mathrm{CNF}$ formula φ with n variables and m clauses, create a Circuit C.

- Inputs to C are the n boolean variables $x_1, x_2, \ldots, x_n$
- Use NOT gate to generate literal $\neg x_i$ for each variable x_i
- For each clause $(\ell_1 \vee \ell_2 \vee \ell_3)$ use two OR gates to mimic formula
- Combine the outputs for the clauses using AND gates to obtain the final output

Example

3SAT $\leq_P$ CSAT

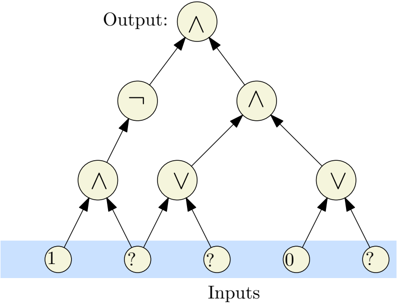
$$\varphi = \left(x_1 \vee x_3 \vee x_4\right) \wedge \left(x_1 \vee \neg x_2 \vee \neg x_3\right) \wedge \left(\neg x_2 \vee \neg x_3 \vee x_4\right)$$

The other direction: **CSAT $\leq_P$ 3SAT**

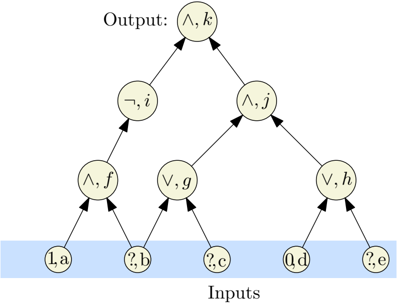
1. Now: **CSAT $\leq_P$ SAT**
2. More "interesting" direction.

Converting a circuit into a CNF formula

Label the nodes



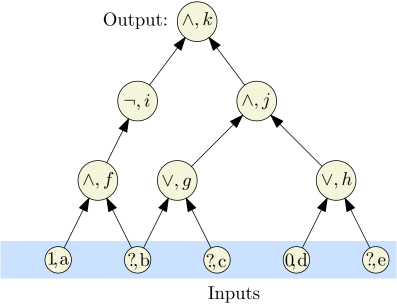
(A) Input circuit



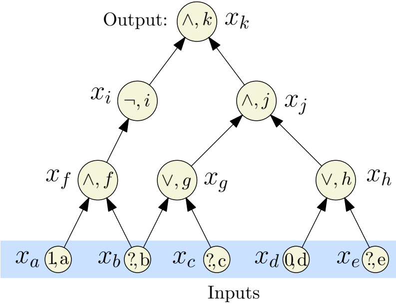
(B) Label the nodes.

Converting a circuit into a CNF formula

Introduce a variable for each node



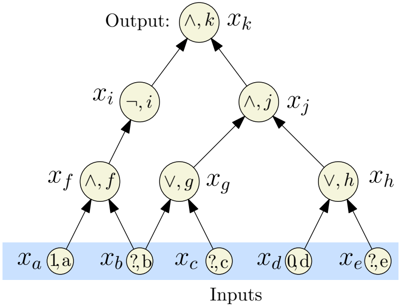
(B) Label the nodes.



(C) Introduce var for each node.

Converting a circuit into a CNF formula

Write a sub-formula for each variable that is true if the var is computed correctly.



(C) Introduce var for each node.

$\mathbf{x_k}$ (Demand a sat' assignment!)

$\mathbf{x_k = x_i \wedge x_j}$

$\mathbf{x_j = x_g \wedge x_h}$

$\mathbf{x_i = \neg x_f}$

$\mathbf{x_h = x_d \vee x_e}$

$\mathbf{x_g = x_b \vee x_c}$

$\mathbf{x_f = x_a \wedge x_b}$

$\mathbf{x_d = 0}$

$\mathbf{x_a = 1}$

(D) Write a sub-formula for each variable that is true if the var is computed correctly.

Reduction: CSAT $\leq_P$ SAT

1. For each gate (vertex) v in the circuit, create a variable x_v
2. Case $\neg$: v is labeled $\neg$ and has one incoming edge from u (so $x_v = \neg x_u$). In **SAT** formula generate, add clauses $(x_u \vee x_v)$, $(\neg x_u \vee \neg x_v)$. Observe that

$$x_v = \neg x_u \text{ is true} \iff \begin{matrix} (x_u \vee x_v) \\ (\neg x_u \vee \neg x_v) \end{matrix} \text{ both true.}$$

Reduction: CSAT $\leq_P$ SAT

Continued...

1. Case $\vee$: So $x_v = x_u \vee x_w$. In **SAT** formula generated, add clauses $(x_v \vee \neg x_u)$, $(x_v \vee \neg x_w)$, and $(\neg x_v \vee x_u \vee x_w)$. Again, observe that

$$\Big(x_v = x_u \vee x_w\Big) \text{ is true} \iff \begin{array}{l} (x_v \vee \neg x_u), \\ (x_v \vee \neg x_w), \\ (\neg x_v \vee x_u \vee x_w) \end{array} \text{ all true.}$$

Reduction: CSAT $\leq_P$ SAT

Continued...

1. Case $\wedge$: So $x_v = x_u \wedge x_w$. In **SAT** formula generated, add clauses $(\neg x_v \vee x_u)$, $(\neg x_v \vee x_w)$, and $(x_v \vee \neg x_u \vee \neg x_w)$. Again observe that

$$x_v = x_u \wedge x_w \text{ is true} \quad \Longleftrightarrow \quad \begin{array}{l} (\neg x_v \vee x_u), \\ (\neg x_v \vee x_w), \\ (x_v \vee \neg x_u \vee \neg x_w) \end{array} \quad \text{all true.}$$

Reduction: CSAT $\leq_P$ SAT

Continued...

1. If v is an input gate with a fixed value then we do the following. If $x_v = 1$ add clause x_v. If $x_v = 0$ add clause $\neg x_v$
2. Add the clause x_v where v is the variable for the output gate

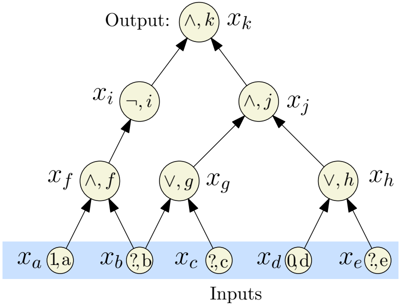
Converting a circuit into a CNF formula

Convert each sub-formula to an equivalent CNF formula

x_k	x_k
$x_k = x_i \wedge x_j$	$(\neg x_k \vee x_i) \wedge (\neg x_k \vee x_j) \wedge (x_k \vee \neg x_i \vee \neg x_j)$
$x_j = x_g \wedge x_h$	$(\neg x_j \vee x_g) \wedge (\neg x_j \vee x_h) \wedge (x_j \vee \neg x_g \vee \neg x_h)$
$x_i = \neg x_f$	$(x_i \vee x_f) \wedge (\neg x_i \vee \neg x_f)$
$x_h = x_d \vee x_e$	$(x_h \vee \neg x_d) \wedge (x_h \vee \neg x_e) \wedge (\neg x_h \vee x_d \vee x_e)$
$x_g = x_b \vee x_c$	$(x_g \vee \neg x_b) \wedge (x_g \vee \neg x_c) \wedge (\neg x_g \vee x_b \vee x_c)$
$x_f = x_a \wedge x_b$	$(\neg x_f \vee x_a) \wedge (\neg x_f \vee x_b) \wedge (x_f \vee \neg x_a \vee \neg x_b)$
$x_d = 0$	$\neg x_d$
$x_a = 1$	x_a

Converting a circuit into a CNF formula

Take the conjunction of all the CNF sub-formulas



$$x_k \wedge (\neg x_k \vee x_i) \wedge (\neg x_k \vee x_j)$$
$$\wedge (x_k \vee \neg x_i \vee \neg x_j) \wedge (\neg x_j \vee x_g)$$
$$\wedge (\neg x_j \vee x_h) \wedge (x_j \vee \neg x_g \vee \neg x_h)$$
$$\wedge (x_i \vee x_f) \wedge (\neg x_i \vee \neg x_f)$$
$$\wedge (x_h \vee \neg x_d) \wedge (x_h \vee \neg x_e)$$
$$\wedge (\neg x_h \vee x_d \vee x_e) \wedge (x_g \vee \neg x_b)$$
$$\wedge (x_g \vee \neg x_c) \wedge (\neg x_g \vee x_b \vee x_c)$$
$$\wedge (\neg x_f \vee x_a) \wedge (\neg x_f \vee x_b)$$
$$\wedge (x_f \vee \neg x_a \vee \neg x_b) \wedge (\neg x_d) \wedge x_a$$

We got a CNF formula that is satisfiable if and only if the original circuit is satisfiable.

Correctness of Reduction

Need to show circuit C is satisfiable iff φ_C is satisfiable

$\Rightarrow$ Consider a satisfying assignment a for C

1. Find values of all gates in C under a
2. Give value of gate v to variable x_v; call this assignment a'
3. a' satisfies φ_C (exercise)

$\Leftarrow$ Consider a satisfying assignment a for φ_C

1. Let a' be the restriction of a to only the input variables
2. Value of gate v under a' is the same as value of x_v in a
3. Thus, a' satisfies C

Part II

Proof of Cook-Levin Theorem

Cook-Levin Theorem

Theorem (Cook-Levin)

SAT *is* **NP-Complete**.

We have already seen that **SAT** is in **NP**.

Need to prove that *every* language $L \in \mathbf{NP}$, $L \leq_P$ **SAT**

Cook-Levin Theorem

Theorem (Cook-Levin)

SAT *is* **NP-Complete**.

We have already seen that **SAT** is in **NP**.

Need to prove that *every* language $L \in$ **NP**, $L \leq_P$ **SAT**

Difficulty: Infinite number of languages in **NP**. Must *simultaneously* show a *generic* reduction strategy.

High-level Plan

What does it mean that $L \in \mathbf{NP}$?

$L \in NP$ implies that there is a non-deterministic TM M and polynomial $p()$ such that

$$L = \{x \in \Sigma^* \mid M \text{ accepts } x \text{ in at most } p(|x|) \text{ steps}\}$$

High-level Plan

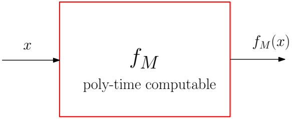
What does it mean that $L \in \mathbf{NP}$?
$L \in NP$ implies that there is a non-deterministic TM M and polynomial $p()$ such that

$$L = \{x \in \Sigma^* \mid M \text{ accepts } x \text{ in at most } p(|x|) \text{ steps}\}$$

We will describe a reduction f_M that depends on M, p such that:

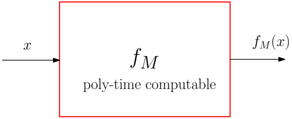
- f_M takes as input a string x and outputs a SAT formula $f_M(x)$
- f_M runs in time polynomial in $|x|$
- $x \in L$ if and only if $f_M(x)$ is satisfiable

Plan continued



$f_M(x)$ is satisfiable if and only if $x \in L$
$f_M(x)$ is satisfiable if and only if nondeterministic M accepts x in $p(|x|)$ steps

Plan continued



$f_M(x)$ is satisfiable if and only if $x \in L$
$f_M(x)$ is satisfiable if and only if nondeterministic M accepts x in $p(|x|)$ steps

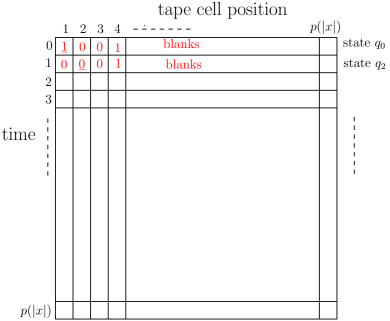
BIG IDEA

- $f_M(x)$ will express "M on input x accepts in $p(|x|)$ steps"
- $f_M(x)$ will encode a computation history of M on x

$f_M(x)$ will be a carefully constructed CNF formula s.t if we have a satisfying assignment to it, then we will be able to see a complete accepting computation of M on x *down to the last detail* of where the head is, what transition is chosen, what the tape contents are, at each step.

Tableau of Computation

M runs in time ***p*(|*x*|)** on ***x***. Entire computation of ***M*** on ***x*** can be represented by a "tableau"



Row i gives contents of all cells at time i

At time $\mathbf{0}$ tape has input x followed by blanks

Each row long enough to hold all cells M might ever have scanned.

Variables of $f_M(x)$

Four types of variable to describe computation of M on x

Variables of $f_M(x)$

Four types of variable to describe computation of M on x

- $T(b, h, i)$: tape cell at position h holds symbol b at time i.
  $1 \leq h \leq p(|x|)$, $b \in \Gamma$, $0 \leq i \leq p(|x|)$

Variables of $f_M(x)$

Four types of variable to describe computation of M on x

- $T(b, h, i)$: tape cell at position h holds symbol b at time i.
  $1 \leq h \leq p(|x|)$, $b \in \Gamma$, $0 \leq i \leq p(|x|)$
- $H(h, i)$: read/write head is at position h at time i.
  $1 \leq h \leq p(|x|)$, $0 \leq i \leq p(|x|)$

Variables of $f_M(x)$

Four types of variable to describe computation of M on x

- $T(b, h, i)$: tape cell at position h holds symbol b at time i.
  $1 \leq h \leq p(|x|)$, $b \in \Gamma$, $0 \leq i \leq p(|x|)$
- $H(h, i)$: read/write head is at position h at time i.
  $1 \leq h \leq p(|x|)$, $0 \leq i \leq p(|x|)$
- $S(q, i)$ state of M is q at time i $q \in Q$, $0 \leq i \leq p(|x|)$

Variables of $f_M(x)$

Four types of variable to describe computation of M on x

- $T(b, h, i)$: tape cell at position h holds symbol b at time i.
  $1 \leq h \leq p(|x|)$, $b \in \Gamma$, $0 \leq i \leq p(|x|)$
- $H(h, i)$: read/write head is at position h at time i.
  $1 \leq h \leq p(|x|)$, $0 \leq i \leq p(|x|)$
- $S(q, i)$ state of M is q at time i $q \in Q$, $0 \leq i \leq p(|x|)$
- $I(j, i)$ instruction number j is executed at time i
  M is non-deterministic, need to specify transitions in some way. Number transitions as $1, 2, \ldots, \ell$ where jth transition is $< q_j, b_j, q'_j, b'_j, d_j >$ indication $(q'_j, b'_j, d_j) \in \delta(q_j, b_j)$, direction $d_j \in \{-1, 0, 1\}$.

Variables of $f_M(x)$

Four types of variable to describe computation of M on x

- $T(b, h, i)$: tape cell at position h holds symbol b at time i.
  $1 \leq h \leq p(|x|)$, $b \in \Gamma$, $0 \leq i \leq p(|x|)$
- $H(h, i)$: read/write head is at position h at time i.
  $1 \leq h \leq p(|x|)$, $0 \leq i \leq p(|x|)$
- $S(q, i)$ state of M is q at time i $q \in Q$, $0 \leq i \leq p(|x|)$
- $I(j, i)$ instruction number j is executed at time i
  M is non-deterministic, need to specify transitions in some way. Number transitions as $1, 2, \ldots, \ell$ where jth transition is $< q_j, b_j, q'_j, b'_j, d_j >$ indication $(q'_j, b'_j, d_j) \in \delta(q_j, b_j)$, direction $d_j \in \{-1, 0, 1\}$.

Number of variables is $O(p(|x|)^2)$ where constant in $O()$ hides dependence on fixed machine M.

Notation

Some abbreviations for ease of notation

$\bigwedge_{k=1}^{m} x_k$ means $x_1 \wedge x_2 \wedge \ldots \wedge x_m$

$\bigvee_{k=1}^{m} x_k$ means $x_1 \vee x_2 \vee \ldots \vee x_m$

$\bigoplus(x_1, x_2, \ldots, x_k)$ is a formula that means exactly one of $x_1, x_2, \ldots, x_m$ is true. Can be converted to CNF form

Clauses of $f_M(x)$

$f_M(x)$ is the conjunction of **8** clause groups:

$$f_M(x) = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \varphi_5 \wedge \varphi_6 \wedge \varphi_7 \wedge \varphi_8$$

where each φ_i is a CNF formula. Described in subsequent slides.
Property: $f_M(x)$ is satisfied iff there is a truth assignment to the variables that simultaneously satisfy $\varphi_1, \ldots, \varphi_8$.

φ_1

φ_1 asserts (is true iff) the variables are set T/F indicating that M starts in state q_0 at time $\mathbf{0}$ with tape contents containing x followed by blanks.

Let $x = a_1 a_2 \ldots a_n$

$\varphi_1 = S(q_0, \mathbf{0})$ state at time $\mathbf{0}$ is q_0

φ_1

φ_1 asserts (is true iff) the variables are set T/F indicating that M starts in state q_0 at time $\mathbf{0}$ with tape contents containing x followed by blanks.

Let $x = a_1 a_2 \ldots a_n$

$\varphi_1 = S(q_0, \mathbf{0})$ state at time $\mathbf{0}$ is q_0
$\bigwedge$ and

φ_1

φ_1 asserts (is true iff) the variables are set T/F indicating that M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

Let $x = a_1 a_2 \ldots a_n$

$\varphi_1 = S(q_0, 0)$ state at time 0 is q_0

$\bigwedge$ and

$\bigwedge_{h=1}^{n} T(a_h, h, 0)$ at time 0 cells 1 to n have a_1 to a_n

φ_1

φ_1 asserts (is true iff) the variables are set T/F indicating that M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

Let $x = a_1 a_2 \dots a_n$

$\varphi_1 = S(q_0, 0)$ state at time 0 is q_0
$\bigwedge$ and
$\bigwedge_{h=1}^{n} T(a_h, h, 0)$ at time 0 cells 1 to n have a_1 to a_n
$\bigwedge_{h=n+1}^{p(|x|)} T(B, h, 0)$ at time 0 cells $n+1$ to $p(|x|)$ have blanks

φ_1

φ_1 asserts (is true iff) the variables are set T/F indicating that M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

Let $x = a_1 a_2 \ldots a_n$

$\varphi_1 = S(q_0, 0)$ state at time 0 is q_0

$\bigwedge$ and

$\bigwedge_{h=1}^{n} T(a_h, h, 0)$ at time 0 cells 1 to n have a_1 to a_n

$\bigwedge_{h=n+1}^{p(|x|)} T(B, h, 0)$ at time 0 cells $n+1$ to $p(|x|)$ have blanks

$\bigwedge$ and

$H(1, 0)$ head at time 0 is in position 1

φ_2

φ_2 asserts M in exactly one state at any time i

$$\varphi_2 = \bigwedge_{i=0}^{p(|x|)} \left(\oplus(S(q_0, i), S(q_1, i), \ldots, S(q_{|Q|}, i))\right)$$

φ_3

φ_3 asserts that each tape cell holds a unique symbol at any given time.

$$\varphi_3 = \bigwedge_{i=0}^{p(|x|)} \bigwedge_{h=1}^{p(|x|)} \oplus(T(b_1,h,i), T(b_2,h,i), \ldots, T(b_{|\Gamma|},h,i))$$

For each time i and for each cell position h exactly one symbol $b \in \Gamma$ at cell position h at time i

φ_4

φ_4 asserts that the read/write head of M is in exactly one position at any time i

$$\varphi_4 = \bigwedge_{i=0}^{p(|x|)} \left(\oplus \left(H(1,i), H(2,i), \ldots, H(p(|x|), i)\right)\right)$$

φ_5

φ_5 asserts that M accepts

- Let q_a be unique accept state of M
- without loss of generality assume M runs all $p(|x|)$ steps

$$\varphi_5 = S(q_a, p(|x|))$$

State at time $p(|x|)$ is q_a the accept state.

φ_5

φ_5 asserts that M accepts

- Let q_a be unique accept state of M
- without loss of generality assume M runs all $p(|x|)$ steps

$$\varphi_5 = S(q_a, p(|x|))$$

State at time $p(|x|)$ is q_a the accept state.
If we don't want to make assumption of running for all steps

$$\varphi_5 = \bigvee_{i=1}^{p(|x|)} S(q_a, i)$$

which means M enters accepts state at some time.

φ_6

φ_6 asserts that M executes a unique instruction at each time

$$\varphi_6 = \bigwedge_{i=0}^{p(|x|)} \oplus(I(1,i), I(2,i), \ldots, I(m,i))$$

where m is max instruction number.

φ_7

φ_7 ensures that variables don't allow tape to change from one moment to next if the read/write head was not there.

"If head is **not** at position h at time i then at time $i+1$ the symbol at cell h must be unchanged"

φ_7

φ_7 ensures that variables don't allow tape to change from one moment to next if the read/write head was not there.

"If head is **not** at position h at time i then at time $i+1$ the symbol at cell h must be unchanged"

$$\varphi_7 = \bigwedge_i \bigwedge_h \bigwedge_{b \neq c} \left(\overline{H(h,i)} \Rightarrow \overline{T(b,h,i) \bigwedge T(c,h,i+1)} \right)$$

φ_7

φ_7 ensures that variables don't allow tape to change from one moment to next if the read/write head was not there.

"If head is **not** at position h at time i then at time $i+1$ the symbol at cell h must be unchanged"

$$\varphi_7 = \bigwedge_i \bigwedge_h \bigwedge_{b \neq c} \left(\overline{H(h,i)} \Rightarrow \overline{T(b,h,i) \bigwedge T(c,h,i+1)} \right)$$

since $A \Rightarrow B$ is same as $\neg A \vee B$, rewrite above in CNF form

$$\varphi_7 = \bigwedge_i \bigwedge_h \bigwedge_{b \neq c} (H(h,i) \vee \neg T(b,h,i) \vee \neg T(c,h,i+1))$$

φ_8

φ_8 asserts that changes in tableau/tape correspond to transitions of M (as Lenny says, this is the big cookie).

Let jth instruction be $< q_j, b_j, q'_j, b'_j, d_j >$

φ_8

φ_8 asserts that changes in tableau/tape correspond to transitions of M (as Lenny says, this is the big cookie).

Let jth instruction be $< q_j, b_j, q'_j, b'_j, d_j >$

$\varphi_8 = \bigwedge_i \bigwedge_j (I(j,i) \Rightarrow S(q_j, i))$ If instr j executed at time i then state must be correct to do j

φ_8

φ_8 asserts that changes in tableau/tape correspond to transitions of M (as Lenny says, this is the big cookie).

Let jth instruction be $< q_j, b_j, q'_j, b'_j, d_j >$

$\varphi_8 = \bigwedge_i \bigwedge_j (I(j,i) \Rightarrow S(q_j, i))$ If instr j executed at time i then state must be correct to do j
$\bigwedge$
$\bigwedge_i \bigwedge_j (I(j,i) \Rightarrow S(q'_j, i+1))$ and at next time unit, state must be the proper next state for instr j

φ_8

φ_8 asserts that changes in tableau/tape correspond to transitions of M (as Lenny says, this is the big cookie).

Let jth instruction be $< q_j, b_j, q'_j, b'_j, d_j >$

$\varphi_8 = \bigwedge_i \bigwedge_j (I(j,i) \Rightarrow S(q_j, i))$ If instr j executed at time i then state must be correct to do j

$\bigwedge$

$\bigwedge_i \bigwedge_j (I(j,i) \Rightarrow S(q'_j, i+1))$ and at next time unit, state must be the proper next state for instr j

$\bigwedge$

$\bigwedge_i \bigwedge_h \bigwedge_j [(I(j,i) \wedge H(h,i)) \Rightarrow T(b_j, h, i)]$ if j was executed and head was at position h, then cell h has correct symbol for j

φ_8

φ_8 asserts that changes in tableau/tape correspond to transitions of M (as Lenny says, this is the big cookie).

Let jth instruction be $< q_j, b_j, q'_j, b'_j, d_j >$

$\varphi_8 = \bigwedge_i \bigwedge_j (I(j,i) \Rightarrow S(q_j, i))$ If instr j executed at time i then state must be correct to do j

$\bigwedge$

$\bigwedge_i \bigwedge_j (I(j,i) \Rightarrow S(q'_j, i+1))$ and at next time unit, state must be the proper next state for instr j

$\bigwedge$

$\bigwedge_i \bigwedge_h \bigwedge_j [(I(j,i) \wedge H(h,i)) \Rightarrow T(b_j, h, i)]$ if j was executed and head was at position h, then cell h has correct symbol for j $\bigwedge$

$\bigwedge_i \bigwedge_j \bigwedge_h [(I(j,i) \wedge H(h,i)) \Rightarrow T(b'_j, h, i+1)]$ if j was done then at time i with head at h then at next time step symbol b'_j was indeed written in position h

φ_8

φ_8 asserts that changes in tableau/tape correspond to transitions of M (as Lenny says, this is the big cookie).

Let jth instruction be $< q_j, b_j, q'_j, b'_j, d_j >$

$\varphi_8 = \bigwedge_i \bigwedge_j (I(j,i) \Rightarrow S(q_j, i))$ If instr j executed at time i then state must be correct to do j
$\bigwedge$
$\bigwedge_i \bigwedge_j (I(j,i) \Rightarrow S(q'_j, i+1))$ and at next time unit, state must be the proper next state for instr j
$\bigwedge$
$\bigwedge_i \bigwedge_h \bigwedge_j [(I(j,i) \bigwedge H(h,i)) \Rightarrow T(b_j, h, i)]$ if j was executed and head was at position h, then cell h has correct symbol for j $\bigwedge$
$\bigwedge_i \bigwedge_j \bigwedge_h [(I(j,i) \wedge H(h,i)) \Rightarrow T(b'_j, h, i+1)]$ if j was done then at time i with head at h then at next time step symbol b'_j was indeed written in position h $\bigwedge$
$\bigwedge_i \bigwedge_j \bigwedge_h [(I(j,i) \wedge H(h,i)) \Rightarrow H(h+d_j, i+1)]$ and head is moved properly according to instr j.

Clauses of $f_M(x)$

$f_M(x)$ is the conjunction of **8** clause groups:

$$f_M(x) = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \varphi_5 \wedge \varphi_6 \wedge \varphi_7 \wedge \varphi_8$$

where each φ_i is a CNF formula.

Clauses of $f_M(x)$

$f_M(x)$ is the conjunction of 8 clause groups:

$$f_M(x) = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \varphi_5 \wedge \varphi_6 \wedge \varphi_7 \wedge \varphi_8$$

where each φ_i is a CNF formula.

φ_1 asserts M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

Clauses of $f_M(x)$

$f_M(x)$ is the conjunction of **8** clause groups:

$$f_M(x) = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \varphi_5 \wedge \varphi_6 \wedge \varphi_7 \wedge \varphi_8$$

where each φ_i is a CNF formula.

φ_1 asserts M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

φ_2 asserts M in exactly one state at any time.

Clauses of $f_M(x)$

$f_M(x)$ is the conjunction of 8 clause groups:

$$f_M(x) = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \varphi_5 \wedge \varphi_6 \wedge \varphi_7 \wedge \varphi_8$$

where each φ_i is a CNF formula.

φ_1 asserts M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

φ_2 asserts M in exactly one state at any time.

φ_3 asserts that each tape cell holds a unique symbol at any time.

Clauses of $f_M(x)$

$f_M(x)$ is the conjunction of **8** clause groups:

$$f_M(x) = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \varphi_5 \wedge \varphi_6 \wedge \varphi_7 \wedge \varphi_8$$

where each φ_i is a CNF formula.

φ_1 asserts M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

φ_2 asserts M in exactly one state at any time.

φ_3 asserts that each tape cell holds a unique symbol at any time.

φ_4 asserts that the head of M is in exactly one position at any time.

Clauses of $f_M(x)$

$f_M(x)$ is the conjunction of 8 clause groups:

$$f_M(x) = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \varphi_5 \wedge \varphi_6 \wedge \varphi_7 \wedge \varphi_8$$

where each φ_i is a CNF formula.

φ_1 asserts M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

φ_2 asserts M in exactly one state at any time.

φ_3 asserts that each tape cell holds a unique symbol at any time.

φ_4 asserts that the head of M is in exactly one position at any time.

φ_5 asserts that M accepts.

Clauses of $f_M(x)$

$f_M(x)$ is the conjunction of 8 clause groups:

$$f_M(x) = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \varphi_5 \wedge \varphi_6 \wedge \varphi_7 \wedge \varphi_8$$

where each φ_i is a CNF formula.

φ_1 asserts M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

φ_2 asserts M in exactly one state at any time.

φ_3 asserts that each tape cell holds a unique symbol at any time.

φ_4 asserts that the head of M is in exactly one position at any time.

φ_5 asserts that M accepts.

φ_6 asserts that M executes a unique instruction at each time.

Clauses of $f_M(x)$

$f_M(x)$ is the conjunction of 8 clause groups:

$$f_M(x) = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \varphi_5 \wedge \varphi_6 \wedge \varphi_7 \wedge \varphi_8$$

where each φ_i is a CNF formula.

φ_1 asserts M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

φ_2 asserts M in exactly one state at any time.

φ_3 asserts that each tape cell holds a unique symbol at any time.

φ_4 asserts that the head of M is in exactly one position at any time.

φ_5 asserts that M accepts.

φ_6 asserts that M executes a unique instruction at each time.

φ_7 ensures that variables don't allow tape to change from one moment to next if the read/write head was not there.

Clauses of $f_M(x)$

$f_M(x)$ is the conjunction of 8 clause groups:

$$f_M(x) = \varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge \varphi_4 \wedge \varphi_5 \wedge \varphi_6 \wedge \varphi_7 \wedge \varphi_8$$

where each φ_i is a CNF formula.

φ_1 asserts M starts in state q_0 at time 0 with tape contents containing x followed by blanks.

φ_2 asserts M in exactly one state at any time.

φ_3 asserts that each tape cell holds a unique symbol at any time.

φ_4 asserts that the head of M is in exactly one position at any time.

φ_5 asserts that M accepts.

φ_6 asserts that M executes a unique instruction at each time.

φ_7 ensures that variables don't allow tape to change from one moment to next if the read/write head was not there.

φ_8 asserts that changes in tableau/tape correspond to transitions of M.

Proof of Correctness

(Sketch)

- Given M, x, poly-time algorithm to construct $f_M(x)$
- if $f_M(x)$ is satisfiable then the truth assignment completely specifies an accepting computation of M on x
- if M accepts x then the accepting computation leads to an "obvious" truth assignment to $f_M(x)$. Simply assign the variables according to the state of M and cells at each time i.

Thus M accepts x if and only if $f_M(x)$ is satisfiable

List of NP-Complete Problems to Remember

Problems

1. **SAT**
2. **3SAT**
3. **CircuitSAT**
4. **Independent Set**
5. **Clique**
6. **Vertex Cover**
7. **Hamilton Cycle** and **Hamilton Path** in both directed and undirected graphs
8. **3Color** and **Color**