CBCS 5th [illegible]

NP- completeness

→ Allmost all algorithms considered so far in all the Lectures (till now 22/11/06) run in worst - case polynomial time

$$T(n) = O(n^k) \text{ for some constant } k$$
$$n = \text{input size}$$

→ {P class} -:

– The class of algorithms that run in polynomial time is called P

→ A problem Q is a binary relation on a set I of instances and a set S of solution

Example -: shortest path problem

instance -: graph G
vertices U and V

solution -: sequence of vertices
(shortest path)

Decision problems :-

→ A decision problem is a Question that has two possible answers yes/no. The Question is about some input

→ statement of decision problem.

- instance description
  [defines the information expected in the input]
- Question part
  [states the actual yes/no Question contains variables defined in the instance description]

Ex: instance : an undirected graph. $G = (V, E)$

Question : Does G contain a clique of K vertices.

instance : an undirected graph. $G = (V, E)$ and an integer k

Question : Does G contain a clique of k vertices

A language L_1 is poly-time reducible to language L_2, written

$L \leq_p L_2$ if $\exists$ a poly-time

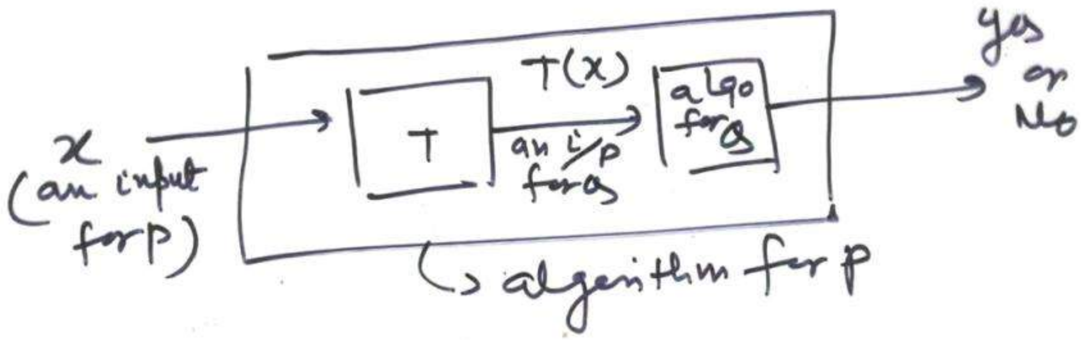
Computable function

$f : \{0,1\}^* \longrightarrow \{0,1\}^*$ such that

$\forall\, x \in \{0,1\}^*$

$$\boxed{x \in L_1 \text{ iff } \underbrace{f}_{\text{reduction fn.}}(x) \in L_2}$$

Note (It's a one way function Q' will not always reduce to Q)



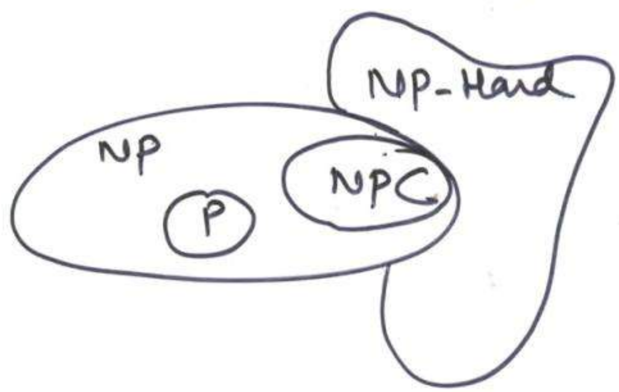
NP Completeness

NP complete problems are the hardest problems in NP i.e every problem in NP reduces to an NP complete problem

→ a language $L \subseteq \{0,1\}^*$ is NPC if $L \in NP$, and $L' \leq_p L$ for $\forall L' \in NP$

→ ~~The~~ a language $L \subseteq \{0,1\}^*$ is NP hard if $L' \leq_p L$ for every $L' \in NP$.

→ A language that is NP hard is not necessarily in NP.



Strategy for proving $L \in NPC$

Step 1 :- Prove $L \in NP$ (poly-time verifiable)

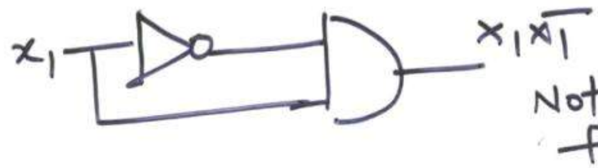
Step 2 :- Select $L' \in NPC$

Step 3 :- Describe a poly-time algorithm computing a function f that maps instances of L' to instance of L

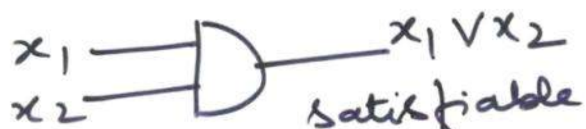
Step 4 :- Prove that $x \in L'$ iff $f(x) \in L \ \forall x \in \{0,1\}^*$

$L' \leq_p L$.

Circuit satisfiability



Not satisfiable for any $x_1 \in \{0,1\}$



satisfiable for $x_1 x_2$ = 01, 10, or 11 thus satisfiable

Given a boolean combination circuit composed of AND, OR and NOT gate is it satisfiable?

$$\text{CIRCUIT-SAT} = \{ \langle C \rangle \mid C \text{ is satisfiable boolean combinational circuit} \}$$

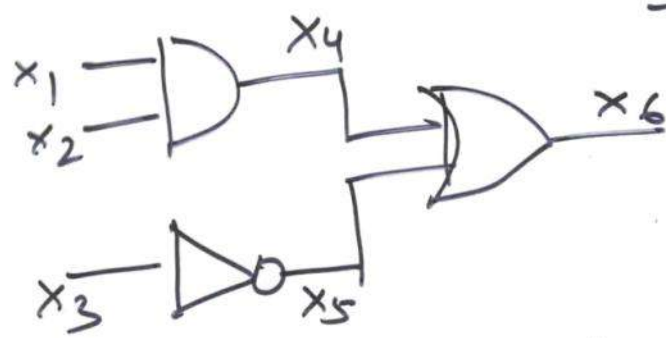
~~<C> is a binary string encodi~~

Determining membership in CIRCUIT-SAT would require checking of 2k possible binary assignment to the K input of a circuit

CIRCUIT SAT $\notin$ P

Reducibility

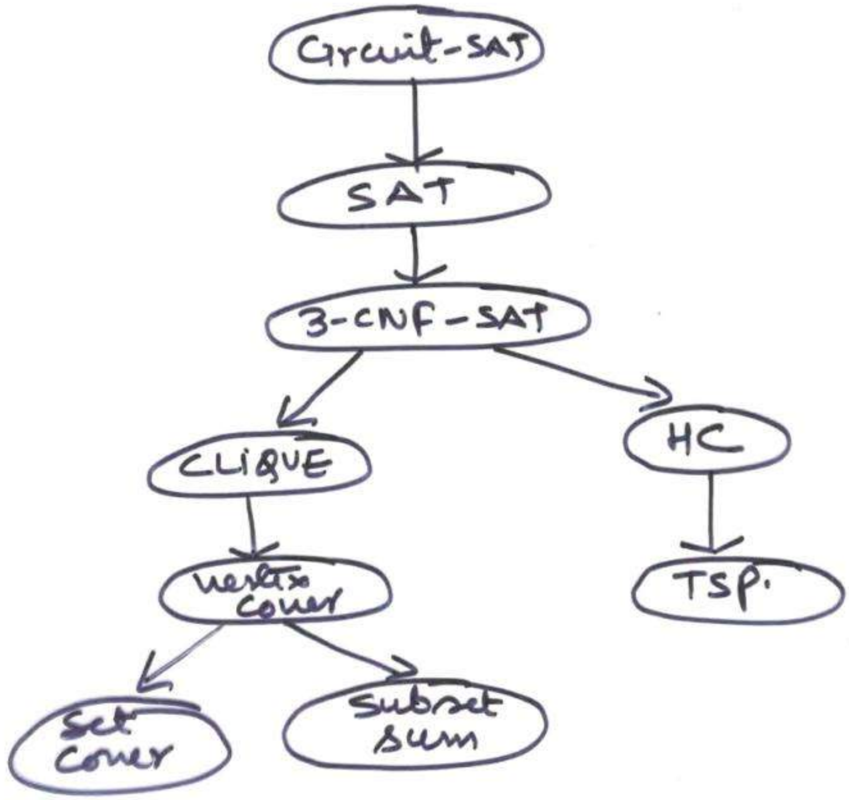
Circuit.



ϕ Formula

$$\phi = x_6 \wedge (x_4 \Leftrightarrow (x_1 \wedge x_2)) \wedge (x_5 \Leftrightarrow \neg x_3) \wedge (x_6 \Leftrightarrow (x_4 \vee x_5))$$

→ constructing this formula takes polynomial time


Circuit-SAT
SAT
3-CNF-SAT
CLIQUE
HC
vertex cover
TSP.
Set cover
subset sum

Cook's theorem

→ Cook's theorem shows that the satisfiability problem is NP-complete

→ Without loss of generality, we assume that languages in NP are over the alphabet $\{0,1\}^*$

Lemma-1 :- if $L \in NP$, then L is accepted by a 1-Tape NTM N with alphabet $\{0,1\}$ such that for some polynomial $p(n)$, the following properties hold

* N's computation is composed of Two phases the guessing phase and checking phase
* In the guessing phase, N Nondeterministically writes a string y directly after the input string and in the checking phase, N behaves deterministically
* N uses at most $p(n)$ tape cells, never moves it's head to the left of w, and takes exactly $p(n)$ steps in the checking phase

Theorem :- CNFSAT is NP-Complete

Proof :- To prove that CNFSAT is NP-complete, we show that for any language $L \in NP$.

$$L \leq_p CNFSAT.$$

Note → Let $L \in NP$ and let N be a NTM accepting L that satisfied the properties earlier mentioned.

→ Transition fn = δ.

- states of N = $q_0 \ldots q_r$
- $s_0 = 0$, $s_1 = 1$, $s_2 = \sqcup$

Note → on input w of length n, how to construct a formula in CNF form f_w, which is satisfiable iff w is accepted by N.

→ The variables of f_{w} are

Variable	Range	Meaning
$Q[i,k]$	$0 \le i \le p(n)$ $0 \le k \le r$	At step i of the checking phase, the state of N is q_k
$H[i,j]$	$0 \le i \le p(n)$ $0 \le j \le p(n)$	At step i of the checking phase, the head of N is at ~~a~~ the tape square j
$S[i,j,l]$	$0 \le i \le p(n)$ $0 \le j \le p(n)$ $0 \le l \le 2$	At step i of the checking phase, the symbol in square j is s_l

→ **Goal** :- To construct f_w so that it's satisfied only by assignments to the variables that corresponds to accepting computations of N on w.

→ The clauses of f_w are constructed to ensure that the following conditions are satisfied

Note

1) At each step i of the checking phase, N is in exactly 1 state $O(p(n))$

$$Q[i,0] \vee Q[i,1] \vee \cdots \vee Q[i,1]$$

Note Need to ensure that N is not both in state q_j and $q_{j'}$. $O(p(n))$

$$\overline{Q[i,j] \vee Q[i,j']}$$

2) At each step i, the head is on exactly one tape square $O(p(n)^2) + O(p(n)^3)$

3) At each step i, there is exactly 1 symbol in each tape square $O(p(n)^2) + O(p(n)^2)$

4) At step 0 of the checking, the state is the initial state $O(1)$

5) At step $p(n)$ of the checking phase, N is in accepting state $O(1)$

6: The configuration of N at the $(i+1)^{st}$ step follow from that at the i^{th} step, by applying the transition function

→ If at step i, the tape head of N is pointing to j^{th} tape cell, N is in q_k, s_l is the symbol under the tape head, $(q_k, s_l, q_{k'}, s_{l'}, X) \in \delta$, $X \in \{L, R\}$

$$\overline{Q[i,k]} \vee \overline{H[i,j]} \vee \overline{S[i,j,l]} \vee Q[i+1,k']$$

→ Then at step $i+1$, the tape head is pointing to $(j+y)$th tape cell, where $y=-1$ if $X=L$, $y=1$ if $X=R$ and ~~x~~ N is in the state $q_{k'}$ and the symbol in the cell j is $s_{l'}$

$O(p(n^2))$

$$\overline{Q[i,k]} \vee \overline{H[i,j]} \vee \overline{S[i,j,l]} \vee H[i+1, j+y]$$

$$\overline{Q[i,k]} \vee \overline{H[i,j]} \vee \overline{S[i,j,l]} \vee S[i+1, j, l']$$

→ All of the clauses for condition 1 to 6 can be computed in polynomial time

Final

→ w is accepted by N iff $f w$ is satisfiable

16

COOK'S THEOREM

CPSC506 | **Complexity of Computation** | **Term 2, 2005**

1 Cook's Theorem

Cook's theorem shows that the satisfiability problem is NP-complete. Without loss of generality, we assume that languages in NP are over the alphabet $\{0,1\}^*$. Lemma 1, useful for the proof, states that we can restrict the form of a computation of a NTM that accepts languages in NP.

Lemma 1 *If $L \in NP$, then L is accepted by a 1-tape NTM N with alphabet $\{0,1\}$ such that for some polynomial $p(n)$, the following properties hold.*

- *N's computation is composed of two phases, the guessing phase and the checking phase.*
- *In the guessing phase, N nondeterministically writes a string $\sqcup y$ directly after the input string, and in the checking phase, N behaves deterministically.*
- *N uses at most $p(n)$ tape cells, never moves its head to the left of w, and takes exactly $p(n)$ steps in the checking phase.*

A Boolean formula f over variable set V is in *conjunctive normal form (CNF)* if

p(n)

$$\boxed{f = \wedge_{i=1}^{m} \vee_{j=1}^{k_i} l_{i,j}}$$

for some values of m and k_i, $1 \leq i \leq m$, where literal $l_{i,j}$ is either x or $\bar{x}$ for some $x \in V$. For each i, the term $\vee_{j=1}^{k_i} l_{i,j}$ is called a clause of the formula. f is *satisfiable* if there exists a truth assignment to the variables in V that sets f to true. CNFSAT is the set of satisfiable Boolean formulas in CNF.

$(\vee \vee \vee) \wedge ($

Theorem 1 (Cook's Theorem) *CNFSAT is NP-complete.*

Proof sketch: It is not hard to show that CNFSAT $\in$ NP. To prove that CNFSAT is NP-complete, we show that for any language $L \in$ NP, $L <_m^p$ CNFSAT.

Let $L \in$ NP and let N be a NTM accepting L that satisfies the properties of Lemma 1. Let the transition function of N be δ. Let the states of N be $q_0, \ldots, q_r$. Let s_0, s_1, s_2 denote $0, 1, \sqcup$, respectively. Assume that the tape cells are numbered consecutively from the left end of the input, starting at 0. On input w of length n, we show how to construct a formula in CNF form f_w, which is satisfiable if and only if w is accepted by N. The variables of f_w are as follows:

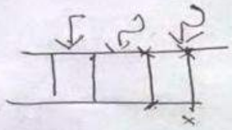
Variables	Range	Meaning
$Q[i,k]$	$0 \leq i \leq p(n)$ $0 \leq k \leq r$	At step i of the checking phase, the state of N is q_k.
$H[i,j]$	$0 \leq i \leq p(n)$ $0 \leq j \leq p(n)$	At step i of the checking phase, the head of N is on tape square j.
$S[i,j,l]$	$0 \leq i \leq p(n)$ $0 \leq j \leq p(n)$ $0 \leq l \leq 2$	At step i of the checking phase, the symbol in square j is s_l.

$p(n) \times p(n)$

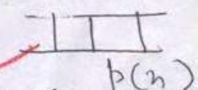
$\leq$ W

A computation of N naturally corresponds to an assignment of truth values to the variables. Other assignments to the variables may be meaningless. For example, an assignment with $Q[i,k] = Q[i,k'] = $ true, $k \neq k'$, would imply that N is in two different states at step i, which is impossible.

Our goal is to construct f_w so that it is satisfied only by assignments to the variables that correspond to accepting computations of N on w. The clauses of f_w are constructed to ensure that the following conditions are satisfied:



1. At each step i of the checking phase, N is in exactly 1 state.
2. At each step i, the head is on exactly one tape square.
3. At each step i, there is exactly 1 symbol in each tape square.
4. At step 0 of the checking phase, the state is the initial state of N in its checking phase, and the tape contents are $w \sqcup y$ for some y.
5. At step $p(n)$ of the checking phase, N is in an accepting state.



6. The configuration of N at the $(i+1)$st step follows from that at the i^{th} step, by applying the transition function of N.

Consider condition 1. For each i, we have the following clause:

$$Q[i,0] \vee Q[i,1] \vee \ldots \vee Q[i,r].$$

This clause ensures that the machine is in at least 1 state at step i. We also need clauses to ensure that N is not both in state q_j and $q_{j'}$:

$$\overline{Q[i,j]} \vee \overline{Q[i,j']} \text{ for each } j \neq j', 0 \leq j, j' \leq r.$$

Conditions 2 and 3 are handled similarly. Conditions 4 and 5 are quite easy. Finally, consider condition 6. For each (i,j,k,l) we add clauses that ensure the following: If at step i, the tape head of N is pointing to the j^{th} tape cell, N is in state q_k, s_l is the symbol under the tape head, and $(q_k, s_l, q_{k'}, s_{l'}, X) \in \delta$, where $X \in \{L, R\}$ then at step $i+1$, the tape head is pointing to the $(j+y)^{th}$ tape cell where $y = 1$ if $X = R$ and $y = -1$ if $X = L$, N is in state $q_{k'}$ and the symbol in cell j is $s_{l'}$. The following clauses ensure this:

$$\overline{Q[i,k]} \vee \overline{H[i,j]} \vee \overline{S[i,j,l]} \vee Q[i+1,k']$$

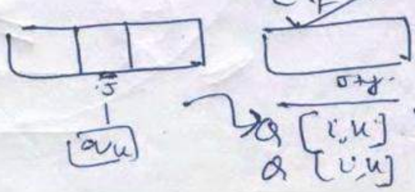
$$\overline{Q[i,k]} \vee \overline{H[i,j]} \vee \overline{S[i,j,l]} \vee H[i+1,j+y]$$

$$\overline{Q[i,k]} \vee \overline{H[i,j]} \vee \overline{S[i,j,l]} \vee S[i+1,j,l']$$

All of the clauses for condition 1 to 6 can be computed in polynomial time (how many clases are there?). Moreover, w is accepted by N if and only if f_w is satisfiable. □

$Q[i,0] \vee Q[i,1] \vee \ldots \vee Q[i,r]$

$\overline{Q[i,J]} \vee \overline{Q[i,J']}$
F T



$\overline{Q[i,u]} \vee \overline{H[i,J]} \vee \overline{[S,i,J,l]} \vee [\longrightarrow$

$Q[i,u] \vee$