CSE 431 Theory of Computation **Spring 2014**

Lecture 11:

Lecturer: James R. Lee *Scribe: Yueqi Sheng*

NP-completeness

11.1 P and NP

Review: definition

- $P = \bigcup_{k \geq 1} TIME(n^k)$;
- $NP = \bigcup_{k \geq 1} NTIME(n^k)$
- $NP \rightarrow$ languages whose YES instance can be verified in (deterministic) polynomial time.

Following are some examples of problem in NP

Theorem 11.1 *SUBSET-SUM* $\in NP$

Proof:

SUBSET-SUM $= \{< S, t >: S = \{x_1, x_2, x_3, \ldots x_k\}$ where $\exists N \subseteq S$, $N = \{n_1, n_2, n_3, \ldots, n_j\}$ such that $\sum n_i = t\}$

(Example: $S = \{1, 17, 4, 8, 3, 9\}$, $t = 22$, then $< S, t >$ is a YES instance of SUBSET-SUM.)

Certificate: Given a set of elements in S $\{y_1, y_2, \ldots y_j\}$ such that $\sum y_i = t$
Verifier : Check each y_i is from S
Check no duplicates
Check $\sum y_i = t$

$\Rightarrow$ SUBSET-SUM is in NP. ∎

Theorem 11.2 $L \in P \rightarrow L \in NP$

Proof: Suppose $L \in P$
Certificate: $-$
Verifier: Run the poly time decider for L ∎

Theorem 11.3 *SAT* $\in NP$

Proof:

$$\text{SAT} = \{< \phi >: \phi \text{ is satisfiable boolean formula}\}$$

First define satisfiable boolean formula:

Boolean Formula: A formula with variables $x_1, x_2, \ldots, x_n$ and their negation $\bar{x_i}$, $\vee$ and $\wedge$
(For example: $\phi = (x_1 \vee \bar{x_2}) \wedge (x_3 \vee x_4) \wedge (x_2 \wedge \bar{x_1}))$
A boolean formula is satisfiable: if $\exists$ assignments that makes ϕ true (evaluate to be 1).
(Taking the formula from obove, assign $x_1, x_2, x_3, x_4 = 1$, $\phi \to 1$, so $\phi \in$ SAT)

Certificate: Assignment to the variables
Verifier: Check if the assignment satisfies the formula.
$\Rightarrow$ SAT $\in NP$ ■

Theorem 11.4 *3SAT* $\in NP$

Proof:

$$\text{3SAT} = \{< \phi >: \phi \text{ is a 3CNF that's satisfiable}\}$$

3CNF:
Definition: $\phi = C_1 \wedge C_2 \wedge C_3 \cdots \wedge C_N$ where $C_i = x_i \vee y_i \vee z_i$. ($x_i, y_i$ and z_i are literals($\to x_i$ and $\bar{x_i}$))
(Example of 3CNF: $(x_2 \vee x_1 \vee \bar{x_4}) \wedge (x_2 \vee x_1 \vee \bar{x_3}) \wedge (x_1 \vee \bar{x_2} \vee x_3))$

From the fact that SAT $\in NP$,
$\Rightarrow$ 3SAT $\in NP$ ■

11.2 Reduction

Definition 11.5 computable
a function $f : \Sigma^* \to \Sigma^*$ is computable if there is a poly-time TM on input w write $f(w)$ on the tape then HALTS.

Definition 11.6 reduciable
a language A is poly time reducible to language B if $\exists$ poly time computable function $f : \Sigma^* \to \Sigma^*$ such that $\forall w \in \Sigma^*$, $w \in A \iff f(w) \in B$
Denoted by $A \leq_P B$

Lemma 11.7 *If $A \leq_P B$ and B has a poly-time algorithm then A has a poly-time algorithm*

Proof: For any input w, to show $w \overset{?}{\in} A$

1. use $A \leq_P B$, map w to $f(w) \in B$;
2. from the fact that $B \in P$, use the ploy time algorithm to compute $f(w) \overset{?}{\in} B$ and use the result;

Then A has a poly-time algorithm. ■

Theorem 11.8 *3SAT $\leq_P$ CLIQUE*

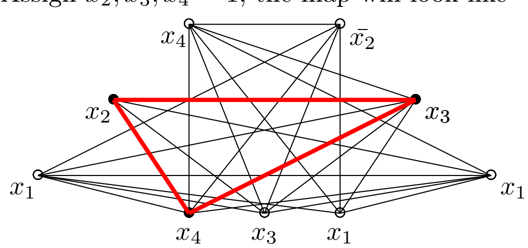
Proof:

Reminder:

$$\text{CLIQUE} = \{< G, k >: \text{if G has a clique of size k}\}$$

TO show that 3SAT $\leq_P$ CLIQUE,we want to show $\phi \underset{poly}{\iff} f(\phi)$ $\phi \underset{poly}{\iff} f(\phi)$ where $f(\phi)$ is some instance $< G, k >$ $\phi = C_1 \wedge C_2 \cdots \wedge C_m$ where $C_i = x_i \vee y_i \vee z_i$

$\phi \to f(\phi)$:
Map ϕ to $f(\phi)$ by add all possible edges excepts the ones between x_i and $\bar{x}_i$
Suppose ϕ is satisfiable, then each C_j has at lease 1 true literal. Pick one true literal x_i from C_i.
For all $j \neq i$, there must be a x_j from C_j that's true and since x_i and $\bar{x}_i$ can not both be true, $x_i \neq \bar{x}_i$. So for each x_i where $i \in [1 \dots m]$, there's a path between x_i and x_j.
Thus we obtain a clique
Example:
P Let $\phi = (x_1 \vee x_2 \vee x_4) \wedge (\bar{x_2} \vee x_3 \vee x_1) \wedge (x_4 \vee x_3 \vee x_1)$
Assign $x_2, x_3, x_4 = 1$, the map will look like



$f(\phi) \to \phi$:
Suppose G has a k clique.
To obtain ϕ, must choose one node for each C_j.
Decode each node to get a partial assignment then fill in the rest of the assignment arbitrarily.
Since there is no edge between x_i and $\bar{x}_i$, they must not be in the clique together, thus there's no inconsistency.
$\Rightarrow \phi$ is satisfied. ■

11.3 COOK-LEVIN THEOREM

Definition 11.9 NP-complete
A language A is NP-complete if:

1. *$A \in NP$;*

2. *$\forall$ problem $B \in NP$, $B \leq_P A$*

Theorem 11.10 Cook-Levin Thm
$P = NP \iff SAT \in P$

Fact If A is NP-complete and $A \in P$, then $P = NP$
Suppose $A \in P$, A is NP-complete, then $\forall B \in NP$, $B \leq_P A$, thus $B \in P$ then we have $P = NP$

Theorem 11.11 *If $NP \neq P$, $\exists L$ such that $L \in NP \wedge L \notin NP$-complete*

Theorem 11.12 Cook-Levin Thm *(restate)*
SAT is NP-complete

We will prove this theorem next time.