COL 352 Lecture 37 April 25

bounded - reduction

$L_1 \leq_f L_2$ iff there is a Turing computable many-one function f, s.t.

$$x \in L_1 \iff f(x) \in L_2$$

and f takes (i) polynomial-time

polynomial time redn

(ii) logarithmic space

... etc.

LOG space redn

LOG space $\Rightarrow$ polytime

If $L_1 \leq_{poly} L_2$ (L_1 is polynomial time reducible to L_2)

then

(i) If $L_2 \in P$ then $L_1 \in P$

(ii) If $L_2 \in NP$ then $L_1 \in NP$

(iii) If $L_1 \notin P$ then $L_2 \notin P$

TM for L_1

(i) Given an input x for L_1,
we compute $f(x_1)$

(ii) Run M_2 (for L_2) on $f(x_1)$
Report the answer ($x \in L_1 \Leftrightarrow f(x_1) \in L_2$)

Does this take overall polynomial-time?

$|x_1| = n$ suppose $f(x_1)$ can be computed in $p_1(n)$ steps p_1: polynomial

$$|f(x_1)| \leq p_1(n)$$

Suppose M_2 takes $p_2(n)$ steps p_2: poly

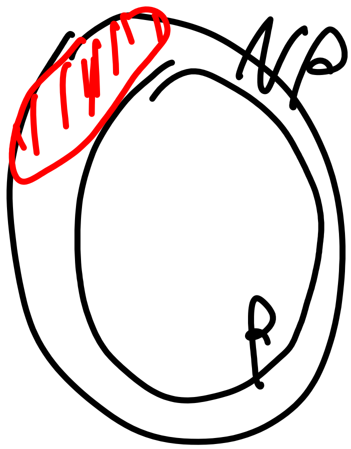
Total time to recognise x_1

$p_1(n) + p_2(p_1(n))$: $p_3(n)$
p_3: polynomial

<u>Claim</u> $L_1 \leq_{poly} L_2$ and $L_2 \leq_{poly} L_3$

$\Rightarrow L_1 \leq_{poly} L_3$

<u>Claim</u> : LOG space reducibility
(f is computable in $\log(|x|)$ space)
is also transitive



"<u>hardest</u>" problems in the class NP

If we could solve the "hardest" problems efficiently $\Rightarrow$ we can solve the rest

NP- complete languages (NPC)

(i) $L \in NP$

(ii) $\forall L', L' \in NP \quad L' \leq_{poly} L$

Suppose L is NPC and L has a polytime time TM $\Rightarrow P = NP$

Corollary: Given some $L_1 \in NP$ to show that L_1 is NPC

(i) $L_1 \in NP$

(ii) If L is NPC and $L \leq_{poly} L_1$ $\Rightarrow$ $L' \leq_{poly} L_1$ for $\forall L' \in NP$

Since L is NPC $\Rightarrow$ $L' \leq_{poly} L$

If $L \leq_{poly} L_1$ $\Rightarrow$ $L' \leq_{poly} L$

Corollary: If L_1, L_2 are both NPC $\Rightarrow$ $L_1 \leq_{poly} L_2$ and $L_2 \leq_{poly} L_1$

Thm (Cook-Levin thm)

The satisfiability problem of boolean variables is NPC

Satisfiability: Given a boolean expression over boolean variables $x_1, x_2 \dots x_n$ is there a truth assignment to x_i's such that the expression is true.

AND

OR

Eg. $$(x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee x_2)$$

$$x_1 \vee \bar{x}_2 \wedge x_1$$

$$(x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee x_2) \wedge (x_1 \vee x_2) \wedge (\bar{x}_1 \vee \bar{x}_2)$$

L_{SAT} = { all satisfiable boolean expression }

$\overline{L_{SAT}}$: all assignments lead to false value