Note Title 2/10/2007

6.841 LECTURE 02

TODAY

- DIAGONALIZATION : POWER + PROBLEMS
- $\mathrm{NTIME}(o(n^2)) \subsetneq \mathrm{NTIME}(n^2)$
- $P \neq NP \Rightarrow \exists L \in \underbrace{NP - (NP\text{-Complete}) - P}_{NP\text{-Intermediate}}$
- Relativization & Implications.

DIAGONALIZATION REVIEW

Say want $L \notin \{L_1, L_2, \ldots L_i, \ldots\}$

Notation: $L(x) = 1$ if $x \in L$
$= 0$ o.w.

Construct infinite matrix

$$
\begin{array}{c|ccccc}
 & L_1 & L_2 & \ldots & j & \ldots \\
\hline
x_1 & 0 & 1 & 0 & 1 & \ldots \\
x_2 & 1 & 1 & 0 & 1 & \ldots \\
x_3 & 0 & 1 & 1 & 0 & \ldots \\
\vdots & 1 & 1 & 0 & 1 & \ldots \\
i & & & & & \\
\end{array}
$$

$L_j(x_i)$

$L = \overline{\text{Diagonal}} = \{1, 0, 0, 0 \ldots\}$

To ensure L has low complexity must be able to <u>simulate</u> computations for

$$\underbrace{L_1, L_2, \ldots L_i, \ldots}$$

(all of them with one machine)

& must be able to <u>complement</u> answer.

<u>Diagonalization Summary</u>

1. ENUMERATION
2. SIMULATION
3. COMPLEMENTATION

Easy to see $\mathrm{TIME}(n^2) \subsetneq \mathrm{TIME}(n^{10}) \ldots$

but what about $\mathrm{NTIME}(n^2)$ vs. $\mathrm{NTIME}(n^{10})$?

Can't complement?

(Lazy Diagonalization).

$$NTIME(o(T(n))) \subsetneq NTIME(T(n+1))$$

Main idea:

To build L that is not recognized by NTM M on some input of length $\geq i$, will do the following

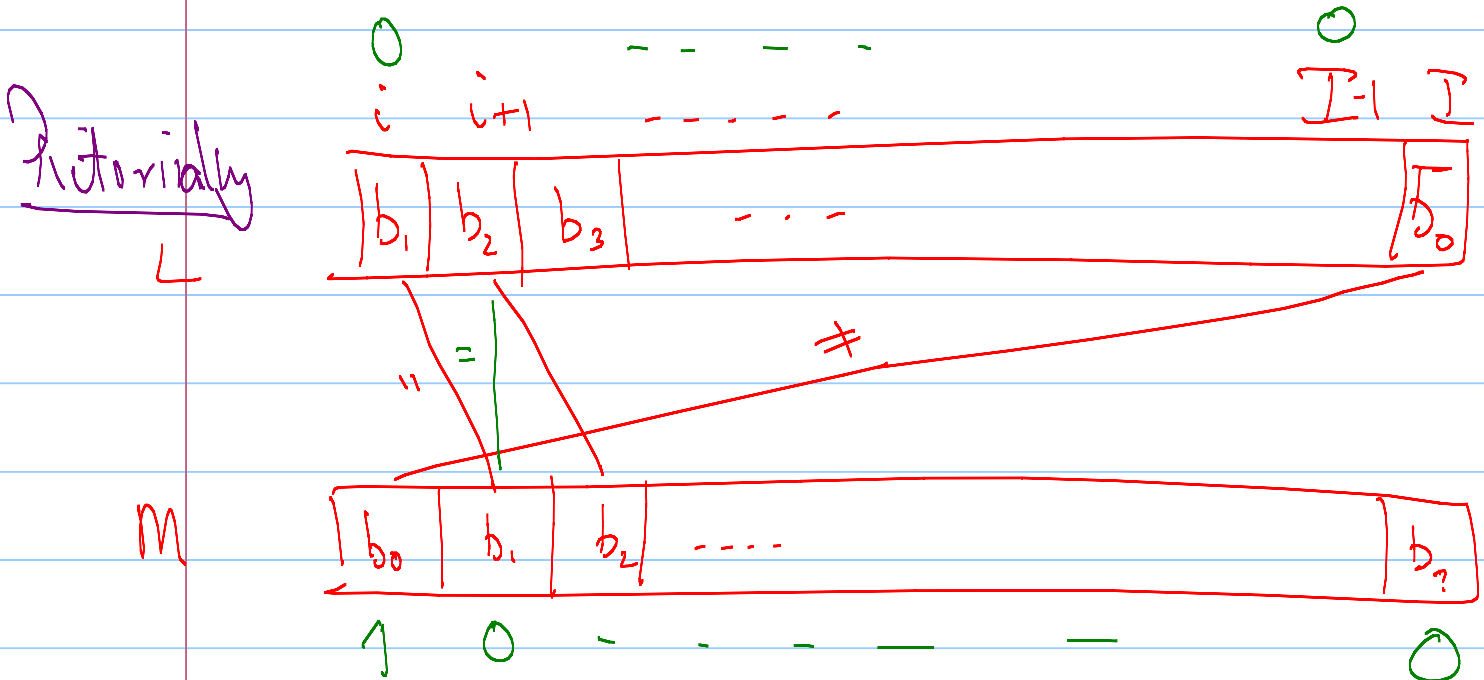
[Notation $L(x) = 1 \iff x \in L$]

$$L(0^j) = M(0^{j+1}) \quad \text{for } j = i, i+1 \ldots I-1,$$

where I s.t. $T(I)$ suffices to decide

$L(0^i)$ deterministically.

Now ~~et~~ let $L(0^I) = \overline{m(0^i)}$.



Claim some $b_i \neq b_{i+1}$ or $L(0^i) \neq m(0^i)$.

Assume otherwise & say $b_1 = 0$

$\Rightarrow b_2 = b_1 = 0$

$\Rightarrow b_3 = b_2 = 0$

$\vdots$

$\Rightarrow b_{I-i} = 0 \Rightarrow b_0 = \overline{b_{I-i}} = 1$

$\Rightarrow b_0 \neq b_1 \Rightarrow L(0^i) \neq m(0^i)$!

Converting this to a real proof.

Let $M_1, M_2, \ldots M_i$ be enumeration of $NTIME(T(n))$ machines.

$L(0^n)$ defined as follows

1. Compute $j = j(n)$ = index of machine to be diagonalized;
& indices i, I to be used as above

$j \leftarrow 1$; $i = 1$;

repeat
$\quad I$ = integer such that
$\quad\quad NTIME(T(i)) \subseteq DTIME(T(I))$;
$\quad$ if $I \geq n$ then return (j, i, I);
$\quad$ else $j \leftarrow j+1$;
$\quad\quad i \leftarrow I+1$;
continue

2. If $n = I$ then $L(0^I) = \overline{M(0^i)}$;
else $L(0^i) = M(0^{i+1})$.

Eventually for every NTIME(T(n)) machine M_j, we find an input length i s.t. $L(0^k) \neq M(0^k)$ for $k \in \{i, \ldots I\}$.

(Theorem above proved by Cook;
Proof from Fortnow's Survey on Diagonalization.
Also see van Melkebeek's paper...)

Moving On: Understanding NP

- Would like to show $NP \neq P$ but have failed.
- Almost all problems we've seen are NP-Complete or in P. Why?

- Is this a theorem?

- Natural counterexamples (at the time): Factoring, LP, Graph Isomorphism.

(Just a handful!)

LADNER'S THEOREM: If $NP \neq P$ then there exist $L \in NP$

$L \notin P$

$L \notin$ NP-Complete

Notation: M^O = Machine M with access to oracle O (subroutine)

e.g. M^{SAT} = M with alg. for SAT etc.

Proof of LADNER's THEOREM

Main Idea:

Let $M_1^O, M_2^O, \ldots M_i^O, \ldots$

be enumeration of polytime reductions (algorithms with access to oracle for O);

Will make sure $L \in NP$, — 0

But $L \neq M_1^\phi, M_2^\phi, \ldots M_i^\phi$ — 1

ϕ = empty language $\in P$

& SAT $\neq M_1^L, M_2^L, \ldots M_i^L \ldots$ — 2

SAT = NP-Complete.

0, 1, 2 $\Rightarrow$ Theorem

Construction of L in stages

j^{th} stage has parameter i:

$j.1$ Will try to make sure

$$L \neq M_j^{\phi} \quad : \quad \text{How?}$$

Input X; unless we find small x s.t.

$$L(x) \neq M_j^{\phi}(x) \quad \text{will let}$$

$$L(X) = SAT(X);$$

$j.2$ Will try to make sure

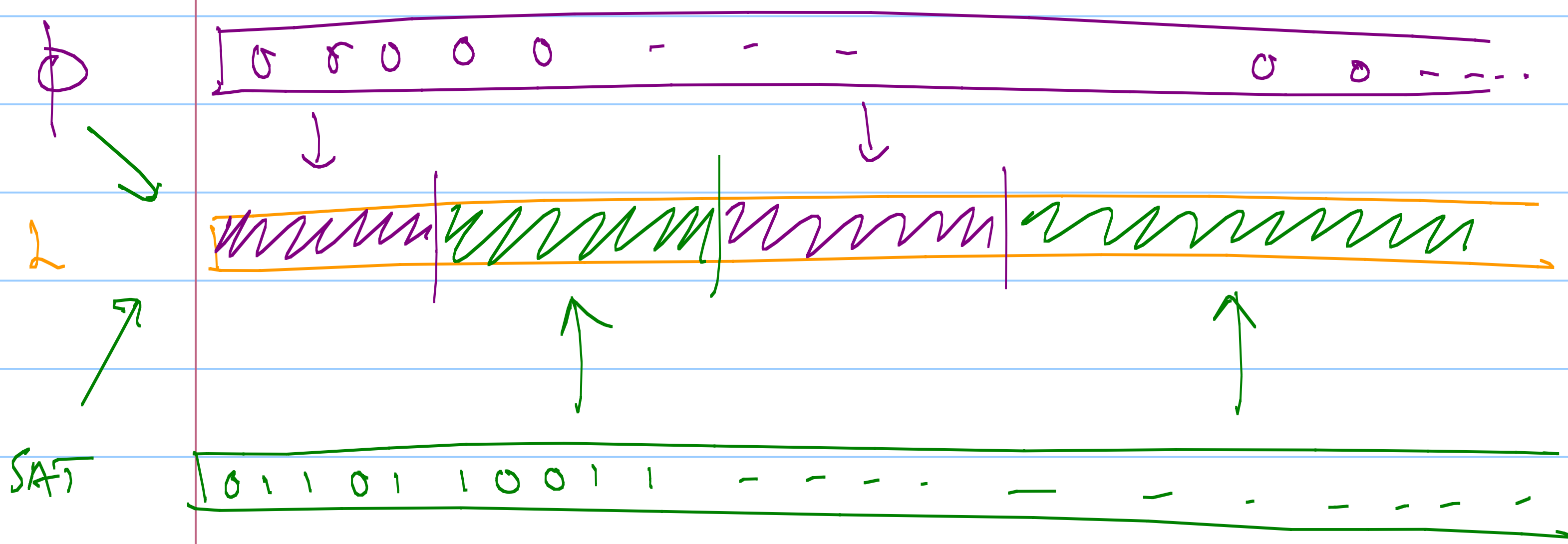
$$SAT \neq M_j^{L} \quad : \quad \text{How?}$$

Input X: unless we find small x s.t.

$$SAT(x) \neq M_j^{L}(x) \quad \text{will let}$$

$$L(X) = 0;$$

Pictorially



No explicit complementation!

So why is $L \notin P$?

Can only happen if we get in stage j.1 (or j.2).

But if so then $\Rightarrow$ $\forall$ sufficiently large x have $L(x) = SAT(x)$; $\Leftarrow$ NP

But also $L(x) = M_j^{\Phi}(x)$ $\Leftarrow$ P !

Converting to formal proof : Good Exercise (on PS1);

So .... Can Diagonalization Prove $NP \neq P$?

Correct Answer : Don't know!

Functional Answer : [Baker Gill Solovay]

Not directly ....

Feature of diagonalization ... relativizes.

i.e. if Diagonalization proves that

$$C = \{L(M_1), L(M_2) \ldots L(M_i), \ldots \}$$

does not contain $L(M)$

then if also "proves" that for every O

$$C^{O} = \{ L(M_1^{O}), L(M_2^{O}), \dots L(M_i^{O}) \dots \}$$

does not contain $L(M^{O})$

M_i^{O} = M_i with oracle access to oracle O.

As stated above doesn't make sense.

To make sense should define $M_i \triangleq M_i^{\phi}$

trivial oracle.

& $M = M^{\phi}$

Assertion 1: if Diagonalization proves $NP \neq P$ then for every O it proves $NP^O \neq P^O$

(By fiat)

Proposition 1: $NP^{TQBF} = P^{TQBF} = PSPACE$

$(\exists O \text{ s.t. } NP^O = P^O)$

Proposition 2: $\exists O$ s.t. $NP^O \neq P^O$

Proof: Construct O by Diagonalization!

$L^O = \{x \mid \text{if } \exists y \;\; |y| = |x| \;\&\; O(y) = 1\}$.

$L^O \in NP^O \quad \forall O$.

Need O s.t. $L^O \notin P^O$

To ensure $L^O \neq M_j^O$ on inputs of length $\geq i$

consider queries that $M_j(0^i)$ makes to O; for $O(x)$ $|x| < i$ answer based on whatever was decided.

for $|x| \geq i$ say $\underline{O(x) = 0}$;

can only ask $p(i)$ such queries; if $p(i) < 2^i$ still have other y $|y| = i$;

fix their value to negate the answer $M_i^O(0^i)$.

$\boxtimes$

Poetic Justice

Diagonalization proves its own futility!