CSCI 1590 Intro to Computational Complexity

Applications of and Limits on Diagonalization

John E. Savage

Brown University

February 23, 2009

Summary

1. Time Hierarchy Theorem
2. Oracle Turing Machines
3. Under Relativization Both $\mathbf{P} = \mathbf{NP}$ and $\mathbf{P} \neq \mathbf{NP}$

Time Hierarchy Theorem

If the TM M on input $\mathbf{x}$ runs in time $t(\mathbf{x})$, the Universal TM U defined previously can simulate this computation in time $O(t^2(\mathbf{x}))$.

To see this, observe that U may bounce back and forth between the description $\lfloor M \rfloor$ of M at the beginning of a tape to the head position, taking at most $O(t(\mathbf{x}))$ steps to simulate one step of M.

Theorem

Let $f : \mathcal{N} \mapsto \mathcal{N}$ and $g : \mathcal{N} \mapsto \mathcal{N}$ be proper resource functions and let $f(n) \log f(n) = o(g(n))$. Then,

$$DTIME(f(n)) \subsetneq DTIME(g(n))$$

Time Hierarchy Theorem

Proof

The result uses the fact that a universal TM U can simulate a TM M on input of length n in $O(n \log n)$ steps.

We prove weaker result, namely, that $\text{DTIME}(n) \subsetneq \text{DTIME}(n^{2.10})$.

Given an input string $\mathbf{x}$, let $M_{\mathbf{x}}$ be the TM with description $\lfloor M \rfloor = \mathbf{x}$. If $\mathbf{x}$ is not well-formed, let it represent the TM that has one state and accepts all inputs.

Let D be a DTM that simulates $M_{\mathbf{x}}$ with the universal TM U on input $\mathbf{x}$ for $|\mathbf{x}|^{2.1}$ steps. If $M_{\mathbf{x}}$ outputs an answer in $\{0,1\}$ (accept, reject), let D produce $D(\mathbf{x}) = 1 - M_{\mathbf{x}}(\mathbf{x})$. Otherwise, let $D(\mathbf{x}) = 0$.

D accepts a language $L \in \text{DTIME}(n^{2.10})$. We show that $L \notin \text{DTIME}(n)$.

Time Hierarchy Theorem

Proof (cont.)

Assume that there exists TM T that decides $\mathbf{x} \in L$ in time cn for some constant $c > 0$, $n = |\mathbf{x}|$. We show a contradiction.

Given T, for every $\mathbf{x} \in \Sigma^*$, $T(\mathbf{x}) = D(\mathbf{x})$. U simulates T on input $\mathbf{x}$ in time at most $d|\mathbf{x}|^2$ for some constant $d > 0$.

There exists n' such that for $n \geq n'$, $n^{2.1} > dn^2$. Because T is equivalent to an infinite set of TMs, there is a description $\mathbf{x}$ of a TM equivalent to T of length greater than n'. Given the definition of D, on input $\mathbf{x}$, $D(\mathbf{x}) = 1 - T(\mathbf{x}) \neq T(\mathbf{x})$. We have a contradiction and conclude that T does not exist.

Oracle Turing Machines

Diagonalization uses two facts, a) each TM can be represented by a computable string and b) a universal TM exists that simulates another TM on its input with a small (logarithmic) overhead. These properties apply to oracle TMs. We show that oracle TMs can't resolve whether or not $\mathbf{P} \ = \mathbf{NP}$.

Definition

An oracle Turing machine (OTM) M is TM and an oracle $O \subseteq \{0,1\}$. M has three special states, q_{oracle}, q_{yes}, and q_{no} and a special read/write tape. M writes a string on this tape. When it enters state q_{oracle}, the oracle determines whether or not this string is in O. If so, it moves M to state q_{yes} in one step. Otherwise, it moves M to q_{no} in one step. M can be deterministic or nondeterministic.

Oracle Turing Machines

Definition

For $O \subseteq \{0,1\}$, $\mathbf{P}^O$ is the class of languages recognized by a PTIME DTM with oracle O. Similarly, $\mathbf{NP}^O$ is the class recognized by a nondeterministic PTIME NTM with oracle O.

Proposition

1. **co**SAT are "No" instances of SAT. Then, **co**SAT $\in \mathbf{P}^{\mathrm{SAT}}$.
2. If $O \in \mathbf{P}$, $\mathbf{P}^O = \mathbf{P}$.
3. Let EXPCOM be the language described below.

$$\{< M, \mathbf{x}, 1^n > \mid M \text{ outputs 1 on } \mathbf{x} \text{ in } 2^n \text{ steps}\}$$

Then, $\mathbf{P}^{\mathrm{EXPCOM}} = \mathbf{NP}^{\mathrm{EXPCOM}} = \mathbf{EXPTIME}$.

Oracle Turing Machines

Proof.

1. To decide the "No" instances of $\textsc{SAT}$, write an instance ϕ of $\textsc{SAT}$ on the oracle tape. Flip the response of the oracle.
2. Clearly $\mathbf{P} \subseteq \mathbf{P}^{O}$. If $O \in \mathbf{P}$, the oracle is redundant; we can simply incorporate its TM into a TM in $\mathbf{P}$. Thus, $\mathbf{P}^{O} \subseteq \mathbf{P}$.
3. Clearly, $\mathbf{EXPTIME} \subseteq \mathbf{P}^{\mathrm{EXPCOM}}$ – the oracle permits an exponential-time computation in one step. Let $M \in \mathbf{NP}^{\mathrm{EXPCOM}}$. In exponential time one can examine the exponentially many choices implied by a polynomial-length certificate and the polynomially many invocations of the $\textsc{EXPCOM}$ oracle. Thus, $\mathbf{NP}^{\mathrm{EXPCOM}} \subseteq \mathbf{EXPTIME}$. It follows that

$$\mathbf{EXPTIME} \subseteq \mathbf{P}^{\mathrm{EXPCOM}} \subseteq \mathbf{NP}^{\mathrm{EXPCOM}} \subseteq \mathbf{EXPTIME}$$

□

Under Relativization Both $\mathbf{P} = \mathbf{NP}$ and $\mathbf{P} \neq \mathbf{NP}$

Recall that diagonalization uses two facts, a) each TM can be represented by a computable string and b) a universal TM exists that simulates another TM on its input with a small (logarithmic) overhead. These properties apply to oracle TMs.

A universal oracle TM with oracle O, OU, exists that can simulate an arbitrary oracle TM using small (logarithmic) overhead using a computable description of an oracle TM.

Theorem (Baker, Gill, Solovay 1975)

There are oracles O_1 and O_2 such that $\mathbf{P}^{O_1} = \mathbf{NP}^{O_1}$ and $\mathbf{P}^{O_2} \neq \mathbf{NP}^{O_2}$.

Diagonalization alone does not suffice to separate **P** from **NP**!

Under Relativization Both $\mathbf{P} = \mathbf{NP}$ and $\mathbf{P} \neq \mathbf{NP}$

Proof

For the first statement, let $O_1 = \text{EXPCOM}$. For the second, we construct a language B. Let U_B be the following unary language:

$$U_B = \{1^n \mid \text{some string of length } n \text{ is in } B\}$$

For every oracle B, $U_B \in \mathbf{NP}^B$ because an NTM given 1^n can guess a string $\mathbf{x} \in B$ and then use the oracle to verify it. We construct a B such that $U_B \notin \mathbf{P}^B$.

B is constructed in stages. At ith stage, $1 \leq i$, strings are added based on oracle queries made by ith oracle TM M_i^B, M_i with oracle B, so that it cannot decide U_B in $\leq 2^n/5$ steps.

Under Relativization Both $\mathbf{P} = \mathbf{NP}$ and $\mathbf{P} \neq \mathbf{NP}$

Proof (cont.)

Initially B is empty. At ith stage choose n larger than the length of any string currently in B. Run M_i^B on input 1^n for $2^n/5$ steps. If M_i^B issues a query string whose status has been determined at earlier stage, give the same response to M_i^B.

If M_i^B halts in $2^n/5$ steps on input 1^n, we make sure that its answer is incorrect. Do this by not including any string of length n in B if M_i^B accepts (ensures that 1^n is rejected) and by including some string of length n in B that has not been queried (ensures that 1^n is accepted). (Such a string exists since at most $2^n/5$ queries have been issued.)

If M_i^B doesn't halt in $2^n/5$ steps on input 1^n, the language it accepts $L_i \notin \mathbf{P}^B$. If M_i^B halts in $2^n/5$ steps on input 1^n, it doesn't accept U_B. It follows that U_B is not in $\mathbf{P}^B$ or that $\mathbf{P}^B \neq \mathbf{NP}^B$.