# CS151
# Complexity Theory

Lecture 18

May 27, 2021

## Approaches to open problems

- Almost all major open problems we have seen entail proving **lower bounds**
  - **$P \neq NP$**
  - **$L \neq P$**
  - **$P \neq PSPACE$**
  - **NC** proper
  - **$BPP \neq EXP$**
  - **PH** proper
  - **$EXP \subsetneq P/poly$**
  - **$P = BPP$ ***
  - **$NP = AM$ ***

• we know circuit lower bounds imply derandomization

• more difficult (and recent): derandomization implies circuit lower bounds!



## Approaches to open problems

- two natural approaches
  - simulation + diagonalization (uniform)
  - circuit lower bounds (non-uniform)
- no success for either approach as applied to date

Why?



## Approaches to open problems

in a precise, formal sense these approaches are **too powerful !**

- if they could be used to resolve major open problems, a side effect would be:
  - proving something that is false, or
  - proving something that is believed to be false



## Relativization

- Many proofs and techniques we have seen **relativize**:
  - they hold after replacing all TMs with oracle TMs that have access to an oracle A
  - e.g. $L^A \subseteq P^A$ for all oracles A
  - e.g. $P^A \neq EXP^A$ for all oracles A



## Relativization

- Idea: design an oracle A relative to which some statement is *false*
  - implies there can be no **relativizing proof** of that statement
  - e.g. design A for which $P^A = NP^A$
- Better: also design an oracle B relative to which statement is *true*
  - e.g. also design B for which $P^B \neq NP^B$
  - implies no **relativizing proof** can resolve truth of the statement either way !

# Relativization

- Oracles are known that falsify almost every major conjecture concerning complexity classes
  - for these conjectures, non-relativizing proofs are required
  - almost all known proofs in Complexity relativize (sometimes after some reformulation)
  - notable exceptions:
    - The PCP Theorem
    - IP = PSPACE
    - most circuit lower bounds (more on these later)



# Oracles for **P** vs. **NP**

- Goal:
  - oracle A for which $\mathbf{P}^A = \mathbf{NP}^A$
  - oracle B for which $\mathbf{P}^B \neq \mathbf{NP}^B$
- conclusion: resolving

  P vs. NP

  requires a non-relativizing proof



# Oracles for **P** vs. **NP**

- for $\mathbf{P}^A = \mathbf{NP}^A$ need A to be powerful
  - warning: intend to make **P** more powerful, but also make **NP** more powerful.
  - e.g. A = SAT doesn't work
  - however A = QSAT works:

  $$\mathbf{PSPACE} \subseteq \mathbf{P}^{QSAT} \subseteq \mathbf{NP}^{QSAT} \subseteq \mathbf{NPSPACE}$$

  and we know **NPSPACE** = **PSPACE**



# Oracles for **P** vs. **NP**

**Theorem**: there exists an oracle B for which $\mathbf{P}^B \neq \mathbf{NP}^B$.

- Proof:
  - define

  $$L = \{1^k : \exists\, x \in B \text{ s.t. } |x| = k\}$$

  - we will show $L \in \mathbf{NP}^B - \mathbf{P}^B$.
  - easy: $L \in \mathbf{NP}^B$ (no matter what B is)



# Oracles for **P** vs. **NP**

- design B by diagonalizing against all "$\mathbf{P}^B$ machines"
- $M_1, M_2, M_3, \ldots$ is an enumeration of deterministic OTMs
- each machine appears infinitely often
- $B_i$ will be those strings of length $\leq i$ in B
- we build $B_i$ after simulating machine $M_i$



# Oracles for **P** vs. **NP**

$$L = \{1^k : \exists\, x \in B \text{ s.t. } |x| = k\}$$

- Proof (continued):
  - maintain "exceptions" X that must not go in B
  - initially X = { }, $B_0$ = { }

  Stage i:
  - simulate $M_i(1^i)$ for $i^{\log i}$ steps
  - when $M_i$ makes an oracle query q:
    - if $|q| < i$, answer using $B_{i-1}$
    - if $|q| \geq i$, answer "no"; add q to X
  - if simulated $M_i$ accepts $1^i$ then $B_i = B_{i-1}$
  - if simulated $M_i$ rejects $1^i$, $B_i = B_{i-1} \cup \{x \in \{0,1\}^i : x \notin X\}$

## Oracles for **P** vs. **NP**

$L = \{1^k : \exists x \in B \text{ s.t. } |x| = k\}$

- Proof (continued):
  - if $M_i$ accepts, we ensure no strings of length i in B
  - therefore $1^i \notin L$, and so $M_i$ does not decide L
  - if $M_i$ rejects, we ensure *some* string of length i in B
  - Why?

    $B_i = B_{i-1} \cup \{x \in \{0,1\}^i : x \notin X\}$

    and |X| is at most $\Sigma_{j \le i} j^{\log j} << 2^i$
  - therefore $1^i \in L$, and so $M_i$ does not decide L
  - Conclude: $L \notin \mathbf{P}^{\mathbf{B}}$



## Circuit lower bounds

- Relativizing techniques are out…
- but most circuit lower bound techniques do not relativize
- exponential circuit lower bounds known for weak models:
  - e.g. constant-depth poly-size circuits
- But, utter failure (so far) for more general models. Why?



## Natural Proofs

- Razborov and Rudich defined the following "natural" format for circuit lower bounds:
  - identify property **P** of functions $f:\{0,1\}^* \rightarrow \{0,1\}$
  - $\mathbf{P} = \cup_n \mathbf{P}_n$ is a natural property if:
    - (useful) $\forall n$ $f_n \in \mathbf{P}_n$ implies f does not have poly-size circuits [i.e. $f_n \in \mathbf{P}_n$ implies ckt size $\ge s(n) >> poly(n)$]
    - (constructive) can decide "$f_n \in \mathbf{P}_n$?" in poly time given the *truth table* of $f_n$
    - (large) at least $(½)^{O(n)}$ fraction of all $2^{2^n}$ functions on n bits are in $\mathbf{P}_n$
  - show some function family $g = \{g_n\}$ is in $\mathbf{P}_n$



## Natural Proofs

- *all known circuit lower bound techniques are natural* for a suitably parameterized version of the definition

**Theorem** (RR): if there is a $2^{n^\epsilon}$-OWF, then there is no natural property **P**.
  - factoring believed to be $2^{n^\epsilon}$-OWF
  - general version also rules out natural properties useful for proving many other separations, under similar cryptographic assumptions



## Natural Proofs

- Proof:
  - **main idea**: natural property $\mathbf{P}_n$ can efficiently distinguish

    pseudorandom functions

    from

    truly random functions
  - but cryptographic assumption implies existence of *pseudorandom functions* for which this is impossible



## Proof (continued)

- Recall: assuming One-Way-Permutations

  $f_k:\{0,1\}^k \rightarrow \{0,1\}^k$

  that are not invertible by $2^{k^\epsilon}$ size circuits
- we constructed PRG $G:\{0,1\}^k \rightarrow \{0,1\}^{2k}$
  - no circuit C of size $s = 2^{k^\delta}$ for which

    $|Pr_x[C(G(x)) = 1] - Pr_z[C(z) = 1]| > 1/s$

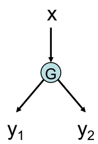
(BMY construction with slightly modified parameters)

# Proof (continued)

- Think of G as $G:\{0,1\}^k \rightarrow \{0,1\}^k \times \{0,1\}^k$

  $G(x) = (y_1, y_2)$

- Graphically:



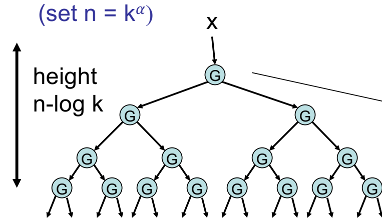



# Proof (continued)

- A function $F:\{0,1\}^k \rightarrow \{0,1\}^{2^n}$

  (set $n = k^\alpha$)

Given x, i, can compute i-th output bit in time n·poly(k)




each x, defines a poly-time computable function $f_x$



# Proof (continued)

(useful) $\forall n\ f_n \in \underline{\mathbf{P}}_n \Rightarrow f$ does not have poly-size circuits
(constructive) "$f_n \in \underline{\mathbf{P}}_n$?" in poly time given *truth table* of $f_n$
(large) at least $(½)^{O(n)}$ fraction of all $2^{2^n}$ fns. on n-bits in $\underline{\mathbf{P}}_n$

- $f_x$ in poly-time $\Rightarrow$ for all x: $f_x \notin \underline{\mathbf{P}}_n$ (useful)
- $\Pr_g[g \in \underline{\mathbf{P}}_n] \geq (1/2)^{O(n)}$ (large)
- constructive: exists circuit $T:\{0,1\}^{2^n} \rightarrow \{0,1\}$ of size $2^{O(n)}$ for which

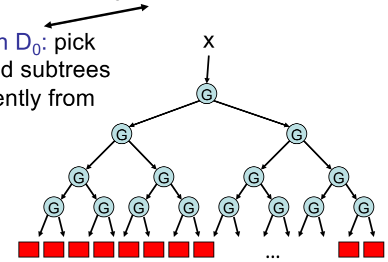
  $|\Pr_x[T(f_x) = 1] - \Pr_g[T(g) = 1]| \geq (1/2)^{O(n)}$



# Proof (continued)

- $|\Pr_x[T(f_x) = 1] - \Pr_g[T(g) = 1]| \geq (1/2)^{O(n)}$

distribution $D_0$: pick roots of red subtrees independently from $\{0,1\}^k$

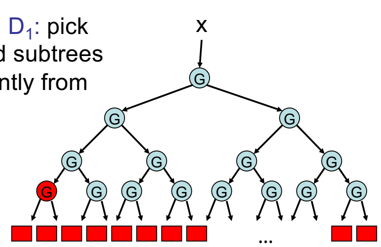





# Proof (continued)

- $|\Pr_x[T(f_x) = 1] - \Pr_g[T(g) = 1]| \geq (1/2)^{O(n)}$

distribution $D_1$: pick roots of red subtrees independently from $\{0,1\}^k$

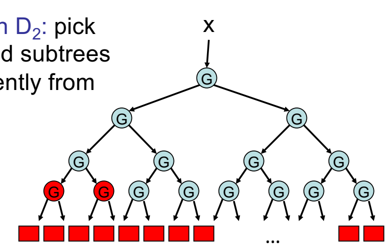





# Proof (continued)

- $|\Pr_x[T(f_x) = 1] - \Pr_g[T(g) = 1]| \geq (1/2)^{O(n)}$

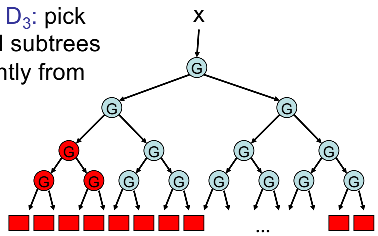
distribution $D_2$: pick roots of red subtrees independently from $\{0,1\}^k$



## Proof (continued)

- $|Pr_x[T(f_x) = 1] – Pr_g[T(g) = 1]| \geq (1/2)^{O(n)}$

distribution $D_3$: pick roots of red subtrees independently from $\{0,1\}^k$

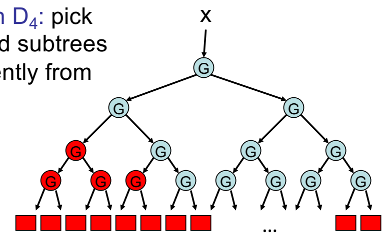





## Proof (continued)

- $|Pr_x[T(f_x) = 1] – Pr_g[T(g) = 1]| \geq (1/2)^{O(n)}$

distribution $D_4$: pick roots of red subtrees independently from $\{0,1\}^k$

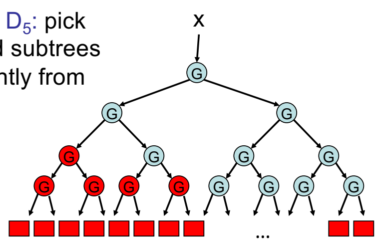





## Proof (continued)

- $|Pr_x[T(f_x) = 1] – Pr_g[T(g) = 1]| \geq (1/2)^{O(n)}$

distribution $D_5$: pick roots of red subtrees independently from $\{0,1\}^k$

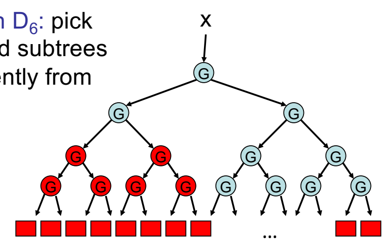





## Proof (continued)

- $|Pr_x[T(f_x) = 1] – Pr_g[T(g) = 1]| \geq (1/2)^{O(n)}$

distribution $D_6$: pick roots of red subtrees independently from $\{0,1\}^k$

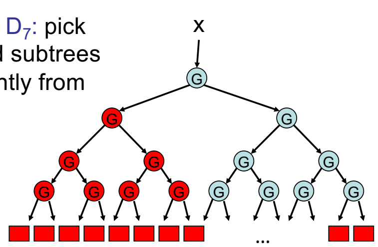





## Proof (continued)

- $|Pr_x[T(f_x) = 1] – Pr_g[T(g) = 1]| \geq (1/2)^{O(n)}$

distribution $D_7$: pick roots of red subtrees independently from $\{0,1\}^k$

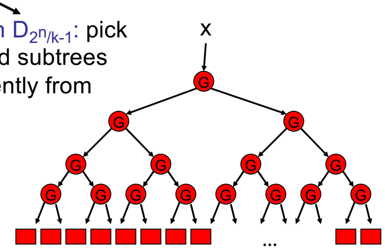





## Proof (continued)

- $|Pr_x[T(f_x) = 1] – Pr_g[T(g) = 1]| \geq (1/2)^{O(n)}$

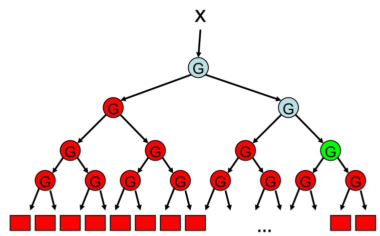
distribution $D_{2^{n/k}-1}$: pick roots of red subtrees independently from $\{0,1\}^k$



## Proof (continued)

– For some i:

$|\Pr[T(D_i) = 1] - \Pr[T(D_{i-1}) = 1]| \geq (1/2)^{O(n)}/2^n = (1/2)^{O(n)}$



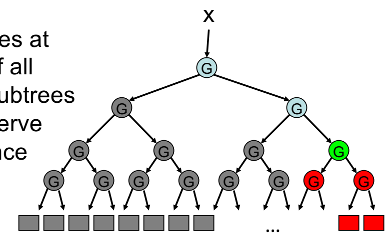



## Proof (continued)

– For some i:

$|\Pr[T(D_i) = 1] - \Pr[T(D_{i-1}) = 1]| \geq (1/2)^{O(n)}/2^n = (1/2)^{O(n)}$

fix values at roots of all other subtrees to preserve difference



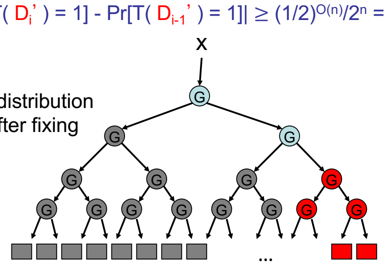



## Proof (continued)

– For some i:

$|\Pr[T( D_i' ) = 1] - \Pr[T( D_{i-1}' ) = 1]| \geq (1/2)^{O(n)}/2^n = (1/2)^{O(n)}$

$D_i'$: distribution $D_i$ after fixing



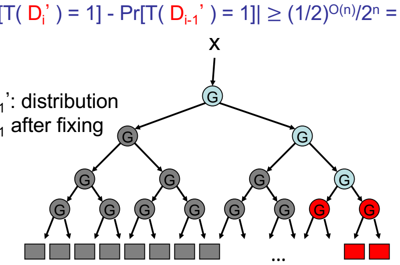



## Proof (continued)

– For some i:

$|\Pr[T( D_i' ) = 1] - \Pr[T( D_{i-1}' ) = 1]| \geq (1/2)^{O(n)}/2^n = (1/2)^{O(n)}$

$D_{i-1}'$: distribution $D_{i-1}$ after fixing

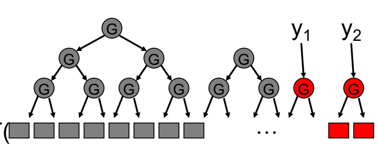





## Proof (continued)

$|\Pr[T( D_i' ) = 1] - \Pr[T( D_{i-1}' ) = 1]| \geq (1/2)^{O(n)}/2^n = (1/2)^{O(n)}$

– $C(y_1,y_2)=T($




$)$

$|\Pr_x[C(G(x)) = 1] - \Pr_{y_1, y_2}[C(y_1, y_2) = 1]| \geq (1/2)^{O(n)}$

T( $D_i'$ )     T( $D_{i-1}'$ )



## Proof (continued)

– recall: no circuit C of size $s = 2^{k^\delta}$ for which:

$|\Pr_x[C(G(x)) = 1] – \Pr_{y_1, y_2}[C(y_1, y_2) = 1]| > 1/s$

– we have C of size $2^{O(n)}$ for which:

$|\Pr_x[C(G(x)) = 1] - \Pr_{y_1, y_2}[C(y_1, y_2) = 1]| \geq (1/2)^{O(n)}$

– with $n = k^\alpha$, $\alpha$ arbitrary constant

– set $\alpha$ such that $2^{O(n)} < s$

– contradiction.

## Natural Proofs

- To prove circuit lower bounds, we must either:
  - Violate largeness: seize upon an incredibly specific feature of hard functions (one not possessed by a random function ! )
  - Violate constructivity: identify a feature of hard functions that cannot be computed efficiently from the **truth table**
- no "non-natural property" known for all but the very weakest models…



"We do not conclude that researchers should give up on proving serious lower bounds.



"We do not conclude that researchers should give up on proving serious lower bounds. Quite the contrary, by classifying a large number of techniques that are unable to do the job, we hope to focus research in a more fruitful direction.



"We do not conclude that researchers should give up on proving serious lower bounds. Quite the contrary, by classifying a large number of techniques that are unable to do the job, we hope to focus research in a more fruitful direction. Pessimism will only be warranted if a *long period of time* passes without the discovery of a non-naturalizing lower bound proof."

**Rudich and Razborov**
**1994**



## Moral

- To resolve central questions:
  - avoid relativizing arguments
    - use PCP theorem and related results
    - focus on circuits, etc…
  - avoid constructive arguments
  - avoid arguments that yield lower bounds for random functions



## Course Summary

## Course summary

- <u>Time and space</u>
  - hierarchy theorems
  - FVAL in **L**
  - CVAL **P**-complete
  - QSAT **PSPACE**-complete
  - succinct CVAL **EXP**-complete



## Course summary

- <u>Non-determinism</u>
  - NTIME hierarchy theorem
  - "NP-intermediate" problems (Ladner's Theorem)
  - unary languages (likely) not NP-complete
  - Savitch's Theorem
  - Immerman-Szelepcsényi Theorem

  Problem sets:
  - *sparse* languages (likely) not NP-complete



## Course summary

- <u>Non-uniformity</u>
  - formula lower bound (Andreev, Hastad)
  - monotone circuit lower bound (Razborov)

  Problem sets:
  - Barrington's Theorem
  - formula lower bound for parity



## Course summary

- <u>Randomness</u>
  - polynomial identity testing + Schwartz-Zippel
  - unique-SAT (Valiant-Vazirani Theorem)
  - Blum-Micali-Yao PRG
  - Nisan-Wigderson PRG
  - worst-case hardness ) average-case hardness
  - Trevisan extractor

  Problem sets:
  - Goldreich-Levin hard bit



## Course summary

- <u>Alternation</u>
  - $\text{QSAT}_i$ complete for levels of the **PH**
  - Karp-Lipton theorem
  - **BPP** in **PH**

  Problem sets:
  - approximate counting + sampling with an **NP**-oracle
  - **VC**-dimension is $\Sigma_3$-complete
  - the class $S_2^P$ (final)



## Course summary

- <u>Counting</u>
  - #matching is **#P**-complete

  Problem sets:
  - permanent is **#P**-complete
  - Toda's theorem: $\mathbf{PH} \subseteq \mathbf{P}^{\#\mathbf{P}}$

## Course summary

- Interaction
  - **IP** = **PSPACE**
  - GI in **NP** ∩ **coAM**
  - using NW PRG for **MA**, variant for **AM**
  - hardness of approximation , PCPs
  - elements of the PCP theorem

  Problem sets:
  - BLR linearity test
  - Clique hard to approximate to within $N^{\epsilon}$



## Course summary

- Barriers to progress
  - oracles rule out relativizing proofs
  - "natural proofs" rule out many circuit lower bound techniques



## Course summary

| | |
|---|---|
| • Time and space | **L, P, PSPACE, EXP** |
| • Non-determinism | **NL, NP, coNP, NEXP** |
| • Non-uniformity | **NC, P/poly** |
| • Randomness | **RL, ZPP, RP, coRP, BPP** |
| • Alternation | **PH, PSPACE** |
| • Counting | **#P** |
| • Interaction | **IP, MA, AM, PCP[log n, 1]** |



## The big picture

- All classes on previous slide are probably distinct, except:
  - **P, ZPP, RP, coRP, BPP** (probably all equal)
  - **L, RL** (probably all equal; **NL**?)
  - **NP, MA, AM** (probably all equal)
  - **IP = PSPACE**
  - **PCP[log n, 1] = NP**
- Only real separations we know separate classes delimiting same resource:
  - e.g. **L** ≠ **PSPACE, NP** ≠ **NEXP**



## The big picture

Remember:

possible explanation for failure to prove conjectured separations…

…is that they are false



## The big picture

- Important techniques/ideas:
  - simulation and diagonalization
  - reductions and completeness
  - self-reducibility
  - encoding information using low-degree polynomials
  - randomness
  - others…

## The big picture

- I hope you take away:
  - an ability to extract the essential features of a problem that make it hard/easy…
  - knowledge and tools to connect computational problems you encounter with larger questions in complexity
  - background needed to understand current research in this area



## The big picture

- background to *contribute* to current research in this area
  - many open problems
  - young field
  - try your hand…