CS 121: Lecture 17
Efficient Computation: P

Madhu Sudan

https://madhu.seas.Harvard.edu/courses/Fall2020

Book: https://introtcs.org

How to contact us
- The whole staff (faster response): CS 121 Piazza
- Only the course heads (slower): cs121.fall2020.course.heads@gmail.com

Announcements:

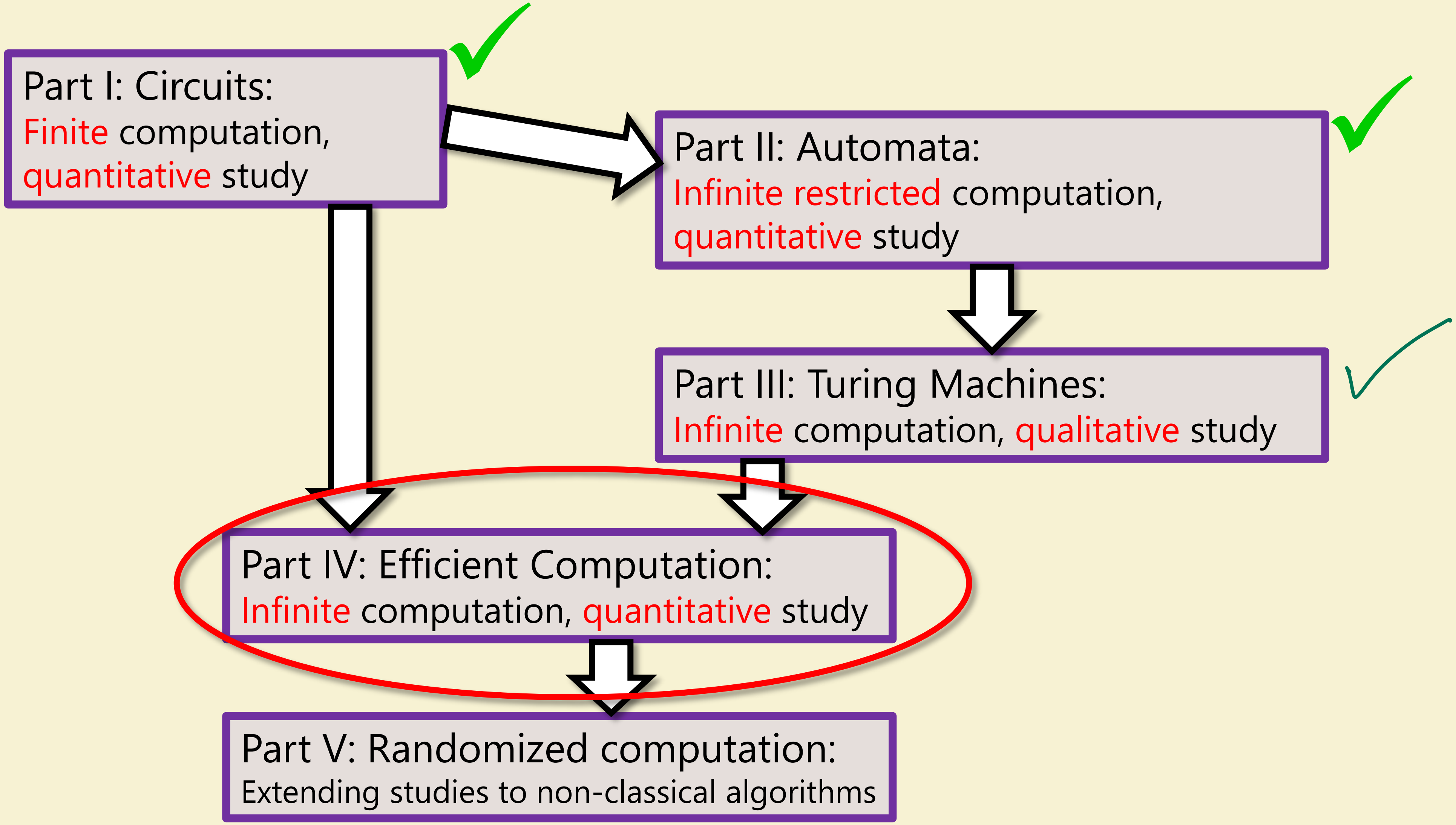
- 121.5: Bjorn Poonen: Uncomputability in Number Theory
  - Why is $x^3 + y^3 + z^3 = 33$ an unsolved equation (over $\mathbb{Z}$)?
- Sections: Week 8 cycle start, material on canvas (as usual).
- Homework 4 due today.
- Homework 5 out. Due in 14 days.

$AB A^{-1} B^{-1} ABCAA^{-1}$

$AB \to BA$

- Make sure to vote. (Lecture absence excused – but must catch up !!)

Where we are:


Part I: Circuits:
Finite computation,
quantitative study
Part II: Automata:
Infinite restricted computation,
quantitative study
Part III: Turing Machines:
Infinite computation, qualitative study
Part IV: Efficient Computation:
Infinite computation, quantitative study
Part V: Randomized computation:
Extending studies to non-classical algorithms

Review of course so far

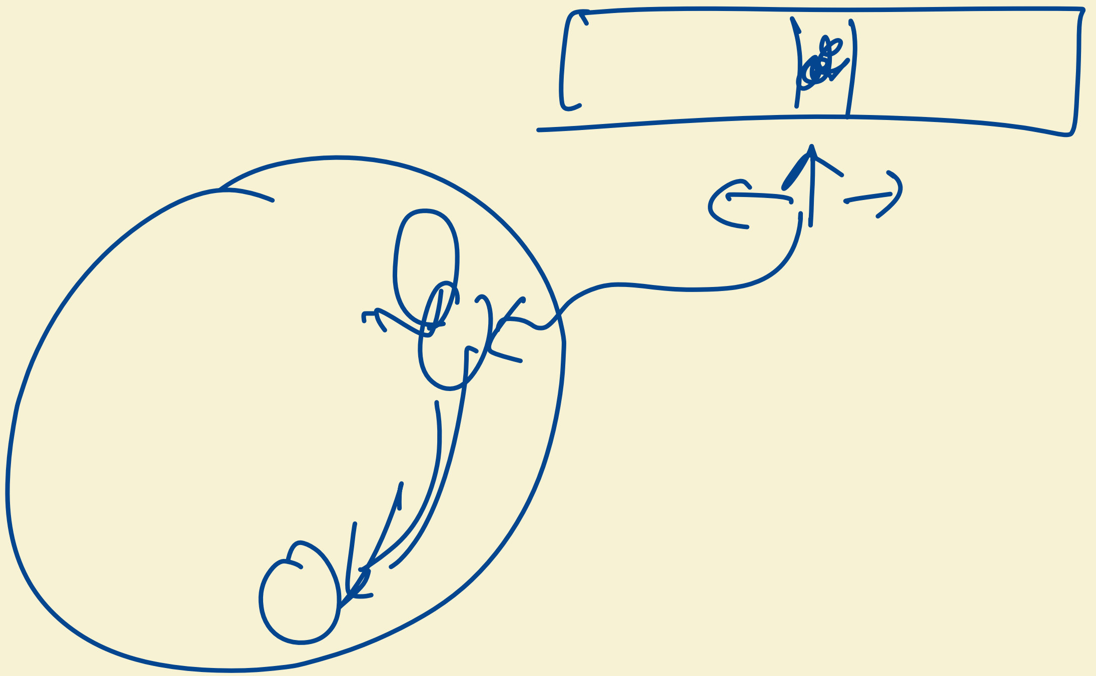
- Circuits:
  - Compute every finite function, with size $O\left(\frac{2^n}{n}\right)$. Some functions require this (by counting). Compute no infinite function.
- Finite Automata:
  - Compute some infinite functions. Do not compute a lot. "Pumping Lemma" (Pigeonhole Principle.)
- Turing Machines:
  - Compute everything computable! (By definition? By thesis? By lack of evidence to the contrary)
  - There exist uncomputable functions: HALT … Rice …

Today

- Defining Running Time
- Time Complexity Classes: P and EXP
- TM $\Leftrightarrow$ RAM time
- Time efficient Universal Simulation + Time Hierarchy Theorem
- Extended Turing-Church Thesis
- Efficiency for Circuits: $\mathrm{P}/_{\mathrm{poly}}$

Running time

- Time = #TM State Transitions.
- Defn: $F: \{0,1\}^* \to \{0,1\}^*$ is computable in time $T(n)$ if there exists a TM M_F that on every input $x \in \{0,1\}^*$, halts after at most $T(|x|)$ transitions and with output $F(x)$ on tape.
- "Best algorithm" + "Worst input"



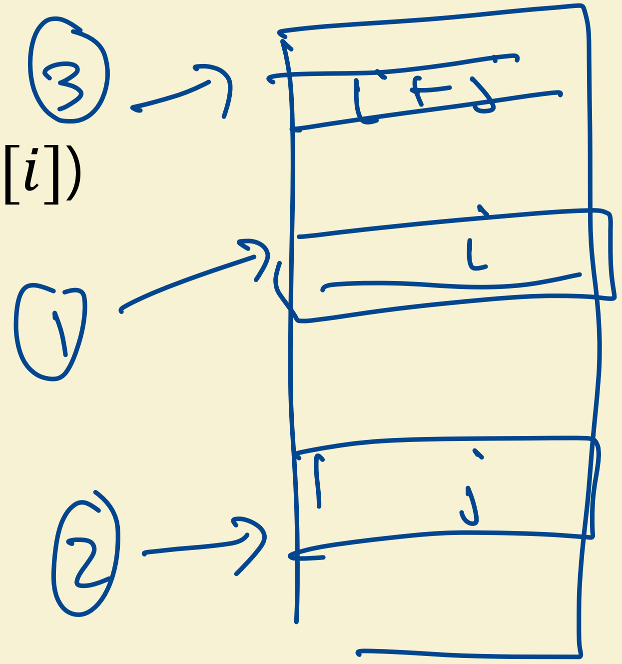
Running time

- Time = #TM State Transitions.
- Defn: $F: \{0,1\}^* \to \{0,1\}^*$ is computable in time $T(n)$ if there exists a TM M_F that on every input $x \in \{0,1\}^*$, halts after at most $T(|x|)$ transitions and with output $F(x)$ on tape.
- "Best algorithm" + "Worst input"
- Do conventions matter?
  - YES: E.g., F(x) = 0 : Time complexity depends on output convention
  - NO: Same up to additive factor of $O(|x| + |F(x)|)$
- Does TM type matter? #tapes? #heads?
  - YES: E.g., Palindrome?
  - NO: But only up to polynomial factors. F computable in time $T(n)$ with k-tape machine $\Rightarrow$ F computable in time $O(T(n)^2)$ with our (standard) model.

RAM Model + Time

RANDOM ACCESS MEMORY

- Common model for algorithm analysis: RAM model + Time.
- RAM Model:
  - Deals with "word"-sized integers in 1 step. $(i + j,\ i * j,\ A[i])$
  - Has built in arrays and allows "random access".
  - Run time "$T_{RAM}(n)$" measures # RAM operations
- Usual algorithm run times stated in this model
  - "Sorting n words takes $O(n \log n)$ time"
  - "Palindrome detection takes $O(n)$ time"
- Theorem: $\mathbf{TIME}(T(n)) \subseteq \mathbf{TIME}_{\text{RAM}}(O(T(n)) \subseteq \mathbf{TIME}(O(T(n)^4))$
- Food for thought: Is $\mathbf{P} = \mathbf{P}_{\text{RAM}}$? Is $\mathbf{EXP} = \mathbf{EXP}_{\text{RAM}}$?



Time Hierarchy Theorem

- Recall Size Hierarchy Theorem for circuits.
  - If $s_1(n)$ sufficiently smaller than $s_2(n)$ sufficiently smaller than $2^n/n$ …
  - Then $\mathbf{SIZE}\big(s_1(n)\big) \subsetneq \mathbf{SIZE}\big(s_2(n)\big)$
  - "More is more"
- **Theorem (13.9):** For nice functions $T(n)$,

$$\mathbf{TIME}_{\mathrm{RAM}}\big(T(n)\big) \subsetneq \mathbf{TIME}_{\mathrm{RAM}}(T(n)\log n)$$

- **Corollaries:**
  - $\mathbf{TIME}\big(T(n)\big) \subsetneq \mathbf{TIME}\big((T(n)\log n)^4\big)$
  - $\mathbf{P} \neq \mathbf{EXP}$

Proof of Time Hierarchy Theorem

- Two ingredients:
  - Timed Universal Turing Machine (Timed RAM Algorithm):
  - Diagonalization

Universal Machine computes EVAL
EVAL(M, x) = M(x) if M halts on x.

- Timed Universal Turing Machine:
  - Let $\mathrm{TIMEDEVAL}(M, x, 1^T) = 1 \Leftrightarrow M$ halts in $\leq T$ steps on x and outputs 1
  - Theorem: TIMEDEVAL computable in time $O(|M|^c \cdot T)$ on RAM.
    - Proof omitted.
  - Corollary: TIMEDEVAL computable in time $O(T^4)$ on some TM!
    - This is the "Timed Universal TM".

|M| = length of encoding of M.

$O(|M|^{4c} T^4)$

Proof of Time Hierarchy Theorem

- Timed Universal Turing Machine:
- Let $\mathrm{TIMEDEVAL}(M, x, 1^T) = 1 \Leftrightarrow$ M halts in $\leq$ T steps on x and outputs 1
- Theorem: TIMEDEVAL computable in time $O(M^c \cdot T)$ on RAM.
- Corollary: TIMEDEVAL computable in time $O(T^4)$ on some TM!

- Two ingredients:
  - Timed Universal Turing Machine (Timed RAM Algorithm):
  - Diagonalization
- Diagonalization:
  - $\mathrm{CANTOR}_T(M, x) = \overline{\mathrm{TIMEDEVAL}(M, (M, x), 1^{T \cdot \log\log x})}$ if $|M| \leq \log\log\log|x|$
  - Claim 1: CANTOR_T computable in time $O(T \log|x|)$ on RAM
  - Claim 2: CANTOR_T not computable in time $O(T)$ on RAM
    - Proof: Suppose M_{CANTOR} computes it in $O(T)$ time. Then for sufficiently long $|x|$

$M_{CANTOR}(M_{CANTOR}, x) = \overline{\mathrm{TIMEDEVAL}(M_{CANTOR}, (M_{CANTOR}, x), 1^{T \cdot \log\log x})} = \overline{M_{CANTOR}(M_{CANTOR}, x)}$

- In the text: $\mathrm{HALT}_T(M, x) = 1$ iff M halts in T steps on input x

Break: Think about CANTOR

- $\text{CANTOR}_T(M,x) = \overline{\text{TIMEDEVAL}(M,(M,x),1^{T\cdot\log\log x})}$ if $|M| \le \log\log\log|x|$
- Claim 1: CANTOR_T computable in time $O(T\log|x|)$ on RAM
- Claim 2: CANTOR_T not computable in time $O(T)$ on RAM
- Proof: Suppose M_{CANTOR} computes it in $O(T)$ time. Then for sufficiently long $|x|$

$M_{CANTOR}(M_{CANTOR},x) = \overline{\text{TIMEDEVAL}(M_{CANTOR},(M_{CANTOR},x),1^{T\cdot\log\log x})} = \overline{M_{CANTOR}(M_{CANTOR},x)}$

- What is x doing?
- Why do we have the T.log log x?
- Why $|M| \le \log\log\log x$?

Solution to "Break: Think about CANTOR"

- $\mathrm{CANTOR}_T(M,x) = \overline{\mathrm{TIMEDEVAL}(M,(M,x),1^{T\cdot\log\log x})}$ if $|M| \leq \log\log\log|x|$
- Claim 1: CANTOR_T computable in time $O(T\log|x|)$ on RAM
- Claim 2: CANTOR_T not computable in time $O(T)$ on RAM
- Proof: Suppose M_{CANTOR} computes it in $O(T)$ time. Then for sufficiently long $|x|$

$$M_{CANTOR}(M_{CANTOR},x) = \overline{\mathrm{TIMEDEVAL}(M_{CANTOR},(M_{CANTOR},x),1^{T\cdot\log\log x})} = \overline{M_{CANTOR}(M_{CANTOR},x)}$$

- What is x doing? (Need long inputs to make algorithms fail!)
- Why do we have the T.log log x?
  - Need to give TIMEDEVAL C.T time for arbitrarily large C. (or else final equality need not hold).
  - Do it by giving it T.log log x time!
- Why $|M| \leq \log\log\log x$?
  - May need $O(|M|^c \cdot T)$ time to universally simulate M for T steps – so needed for Claim 1.

Complexity Classes: **P** and **EXP**

Are all functions BF in EXP? No! Time Hierarchy theorem $T(n) = 2^{2^n}$

- Important: Classes always focus on Boolean Problems!!!!
- **Definition:** $BF: \{0,1\}^* \to \{0,1\}$ is in **P** if BF computable in time $O(n^c)$ for some constant c.
- **Definition:** $BF: \{0,1\}^* \to \{0,1\}$ is in **EXP** if BF computable in time $2^{O(n^c)}$ for some constant c
- **Definition:** $\mathbf{TIME}\big(T(n)\big) =$

$$\{BF: \{0,1\}^* \to \{0,1\} \mid BF \text{ computable in time } T(n)\}$$

- Note: Conventions+Models don't matter for **P**, **EXP**!
- $\mathbf{P} \neq \mathbf{EXP}$ (why?)

Boolean Problems

- Recall: May want to compute $F: \{0,1\}^* \to \{0,1\}^*$
- But complexity captured by $BF: \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$
  - $BF(x,i) \stackrel{\text{def}}{=} F(x)_i$
  - F computable in time $T(n) \Rightarrow BF$ computable in time $O\big(T(n)\big)$
  - BF computable in time $T'(n)$

    $\Rightarrow$ F computable in time $O\big(m \cdot T(n)\big)$ (m = output length)

    $\Rightarrow$ F computable in time $O(T(n)^2)$
  - F polynomial time computable $\Leftrightarrow BF \in \mathbf{P}$
  - F exponential time computable $\Leftrightarrow BF \in \mathbf{EXP}$

- Exercise: Define the Factoring problem. What does BFactoring look like?

Time Hierarchy Theorem

- Recall Size Hierarchy Theorem for circuits.
  - If $s_1(n)$ sufficiently smaller than $s_2(n)$ sufficiently smaller than $2^n/n$ …
  - Then $\mathbf{SIZE}(s_1(n)) \subsetneq \mathbf{SIZE}(s_2(n))$
  - “More is more”

- **Theorem (13.9):** For nice functions $T(n)$,
$$\mathbf{TIME}_{\mathrm{RAM}}(T(n)) \subsetneq \mathbf{TIME}_{\mathrm{RAM}}(T(n)\log n)$$
- **Corollaries:**
  - $\mathbf{TIME}(T(n)) \subsetneq \mathbf{TIME}((T(n)\log n)^4)$
  - $\mathbf{P} \neq \mathbf{EXP}$

Proof of Time Hierarchy Theorem

- Two ingredients:
  - Timed Universal Turing Machine (Timed RAM Algorithm):
  - Diagonalization

- Timed Universal Turing Machine:
  - Let $\mathrm{TIMEDEVAL}(M, x, 1^T) = 1 \Leftrightarrow M$ halts in $\leq T$ steps on x and outputs 1
  - Theorem: TIMEDEVAL computable in time $O(|M|^c \cdot T)$ on RAM.
    - Proof omitted.
  - Corollary: TIMEDEVAL computable in time $O(T^4)$ on some TM!
    - This is the “Timed Universal TM”.

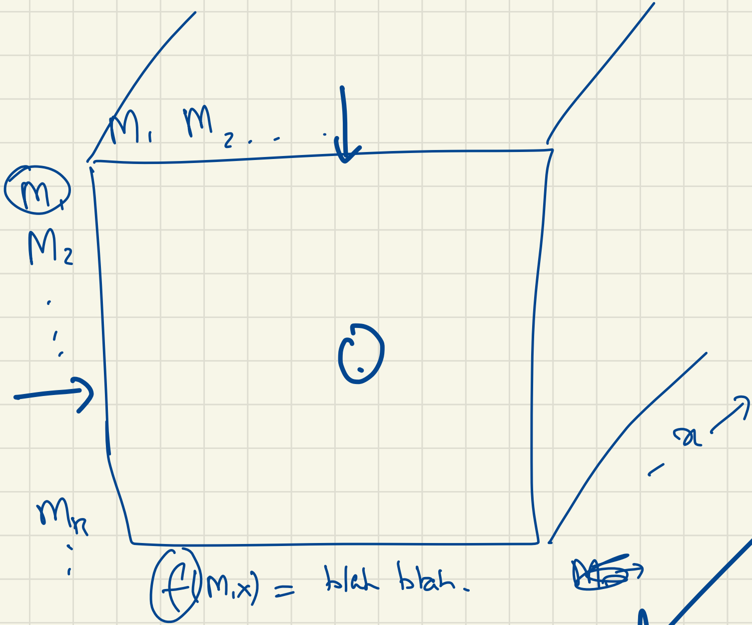
Proof of Time Hierarchy Theorem

- Timed Universal Turing Machine:
- Let $\mathrm{TIMEDEVAL}(M, x, 1^T) = 1 \Leftrightarrow$ M halts in $\leq$ T steps on x and outputs 1
- Theorem: TIMEDEVAL computable in time $O(M^c \cdot T)$ on RAM.
- Corollary: TIMEDEVAL computable in time $O(T^4)$ on some TM!

- Two ingredients:
  - Timed Universal Turing Machine (Timed RAM Algorithm):
  - Diagonalization
- Diagonalization:
  - $\mathrm{CANTOR}_T(M, x) = \overline{\mathrm{TIMEDEVAL}(M, (M, x), 1^{T \cdot \log\log x})}$ if $|M| \leq \log\log\log|x|$
  - Claim 1: CANTOR_T computable in time $O(T\log|x|)$ on RAM
  - Claim 1: CANTOR_T not computable in time $O(T)$ on RAM
    - Proof: Suppose M_{CANTOR} computes it in $O(T)$ time. Then for sufficiently long $|x|$

$$M_{CANTOR}(M_{CANTOR}, x) = \overline{\mathrm{TIMEDEVAL}(M_{CANTOR}, (M_{CANTOR}, x), 1^{T \cdot \log\log x})} = \overline{M_{CANTOR}(M_{CANTOR}, x)}$$

- In the text: $\mathrm{HALT}_T(M, x) = 1$ iff M halts in T steps on input x



$$\mathrm{Cantor}(M,x) = \overline{M(M,x)}$$

$$\mathrm{Cantor}_T(M,x) = \begin{cases} \overline{M_T(M,x)} & \text{if } M(M,x) \text{ halts in } \leq T \text{ steps} \\ 0 & \text{o.w.} \end{cases}$$

$TIME$ vs. $SIZE$

- Given $F:\{0,1\}^* \to \{0,1\}$ can get

$$\{F_n:\{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}'} \quad \text{where} \quad F_n(x) = F(x) \quad \forall x \in \{0,1\}^n$$

- Definition: $F \in \mathbf{P}/_{\mathbf{poly}}$ if $\exists c$ s.t. $\forall n \quad F_n \in SIZE(cn^c)$
- Theorem (13.12): $\mathbf{P} \subseteq \mathbf{P}/_{\mathbf{poly}}$
  - Fast algorithms $\Rightarrow$ small circuits.

" $P/_{Poly}$ " = Poly sized circuits

Extended Turing-Church Thesis

- Vanilla Thesis: Everything computable by physical means is computable by Turing Machine.
- Extended Thesis: Everything computable by physical means **in T time** is computable by Turing Machine **in $O(T^c)$ time**

- Mostly uncontested: Two live challengers:
  - Randomized computation (believed not stronger)
  - Quantum computation (believed stronger?)

Philosophical aside: Importance of $\mathbf{P}$

- Mathematically nice: Robust to models.
- Captures "intuitive" sense of "solving by understanding" (as opposed to "brute force")
  - Problem is in P iff we understand the problem?
  - Seems to hold for most problems we study
- Captures "feasibility" fairly well in practice
  - Is n^{100} practical?
  - But are there practical problems for which we have an n^{100} solution!

Summary of Lecture:

- Introduced time complexity (RAM and TM).
  - Should know both exist and are closely related. No need to know proofs.
- TIME Hierarchy theorem:
  - Uses Universal TM. (No need to know construction)
  - Diagonalization (Must know proof!)
- Complexity classes $\mathbf{P}$ and $\mathbf{EXP}$
- Mentions: No (need to know) proofs
  - SIZE vs TIME: $\mathbf{P}/_{\mathbf{poly}}$
  - Extended Turing-Church Thesis
  - Importance of $\mathbf{P}$