CMSC 28100

Introduction to Complexity Theory

Spring 2024
Instructor: William Hoza

The Church-Turing Thesis

- Let L be a language

Church-Turing Thesis:

There exists an "algorithm" / "procedure" for figuring out whether a given string is in L if and only if there exists a Turing machine that decides L.

← Intuitive notion

← Mathematically precise notion

Are Turing machines too powerful?

- **OBJECTION:** “The Turing machine’s infinite tape is unrealistic!”
- **RESPONSE 1:** If M decides some language, then on any particular input w, M only uses a finite amount of space
- **RESPONSE 2:** We are studying idealized computation
- **RESPONSE 3:** We’re especially focused on impossibility results, so it’s better to err on the side of making the model extra powerful

Are Turing machines powerful enough?

- **OBJECTION:** "To encompass all possible algorithms, we should add various bells and whistles to the Turing machine model."
- Example: Let's define a left-right-stationary Turing machine just like an ordinary Turing machine, except now the transition function has the form $\delta: Q \times \Gamma \to Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}, \mathrm{S}\}$
- S means the head does not move in this step (prohibited if we see $\lozenge$)
- (Exercise: Rigorously define NEXT, accepting, rejecting, etc.)

Left-right-stationary Turing machines

- The left-right-stationary Turing machine model poses a challenge to the Church-Turing thesis, because the model is still realistic, even though we added an extra feature
- Does the Church-Turing thesis survive this challenge?
- Yes, because the left-right-stationary Turing machine model is equivalent to the original Turing machine model, in the following sense:

Left-right-stationary Turing machines

- Let L be a language

Theorem: There exists a left-right-stationary TM that decides L if and only if there exists a TM that decides L

- **Proof:** The $(\Leftarrow)$ direction is trivial, because a TM can be considered a left-right-stationary TM that just happens to never use S

Left-right-stationary Turing machines

- Idea of the proof of $(\Rightarrow)$: Simulate S by doing L followed by R
- Details: Let $M = \left(Q, \Sigma, \Gamma, \diamondsuit, \sqcup, \delta, q_0, q_{\text{accept}}, q_{\text{reject}}\right)$ be a left-right-stationary TM that decides L
- New TM: $M' = \left(Q', \Sigma, \Gamma, \diamondsuit, \sqcup, \delta', q_0, q_{\text{accept}}, q_{\text{reject}}\right)$
- New set of states: $Q' = Q \cup \left\{\underline{q} : q \in Q\right\}$, i.e., two disjoint copies of Q

Left-right-stationary Turing machines

- New transition function $\delta': Q' \times \Gamma \to Q' \times \Gamma \times \{\mathrm{L}, \mathrm{R}\}$ given by:
  - If $\delta(q, b) = (q', b', \mathrm{L})$, then $\delta'(q, b) = \delta(q, b)$
  - If $\delta(q, b) = (q', b', \mathrm{R})$, then $\delta'(q, b) = \delta(q, b)$
  - If $\delta(q, b) = (q', b', \mathrm{S})$, then $\delta'(q, b) = \left(\underline{q'}, b', \mathrm{L}\right)$
  - For every q and b, we let $\delta'\left(\underline{q}, b\right) = (q, b, \mathrm{R})$
- Exercise: Rigorously prove that M' decides L

The Church-Turing Thesis

- Let L be a language

Church-Turing Thesis:

There exists an "algorithm" / "procedure" for figuring out whether a given string is in L if and only if there exists a Turing machine that decides L.

← Intuitive notion

← Mathematically precise notion

Multi-tape Turing

In each step, what determines the actions of head 1?

A: Head 1's state and the symbol observed by head 1

B: The machine's state and the symbols observed by all heads

C: Head 1's state and the symbols observed by all heads

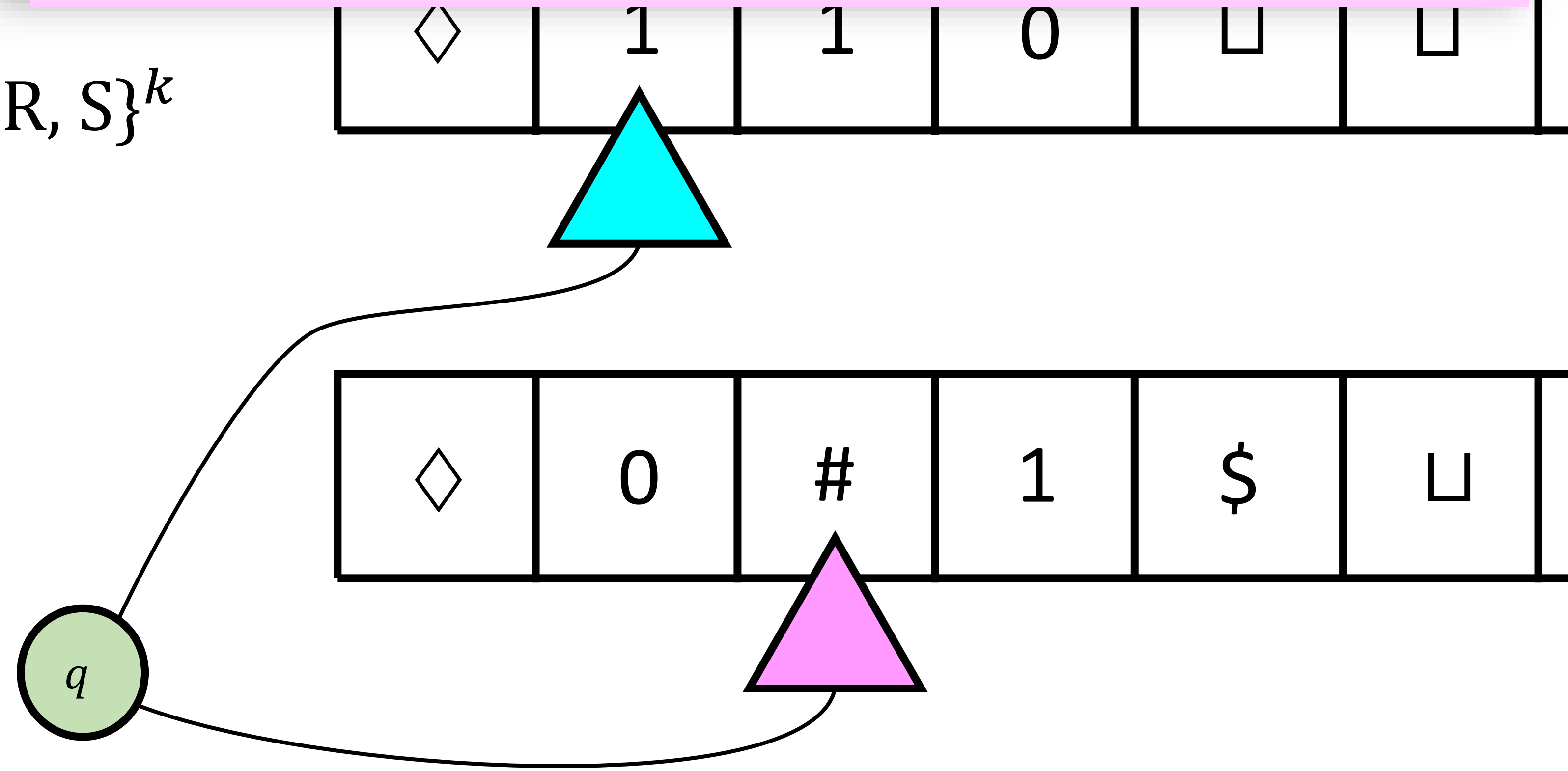
D: The machine's state and the symbol observed by head 1

Respond at PollEv.com/whoza or text "whoza" to 22333

- Another TM variant: "k-tape
- Transition function:

$$\delta: Q \times \Gamma^k \to Q \times \Gamma^k \times \{\mathrm{L}, \mathrm{R}, \mathrm{S}\}^k$$

- (Exercise: Rigorously define acceptance, rejection, etc.)



Multi-tape Turing

- Let k be any positive inte

How should we keep track of the locations of the simulated heads?

A: Store the location data in the machine's state

B: Ensure that the real/simulated heads' locations are always equal

C: Use special symbols to mark the cells containing simulated heads

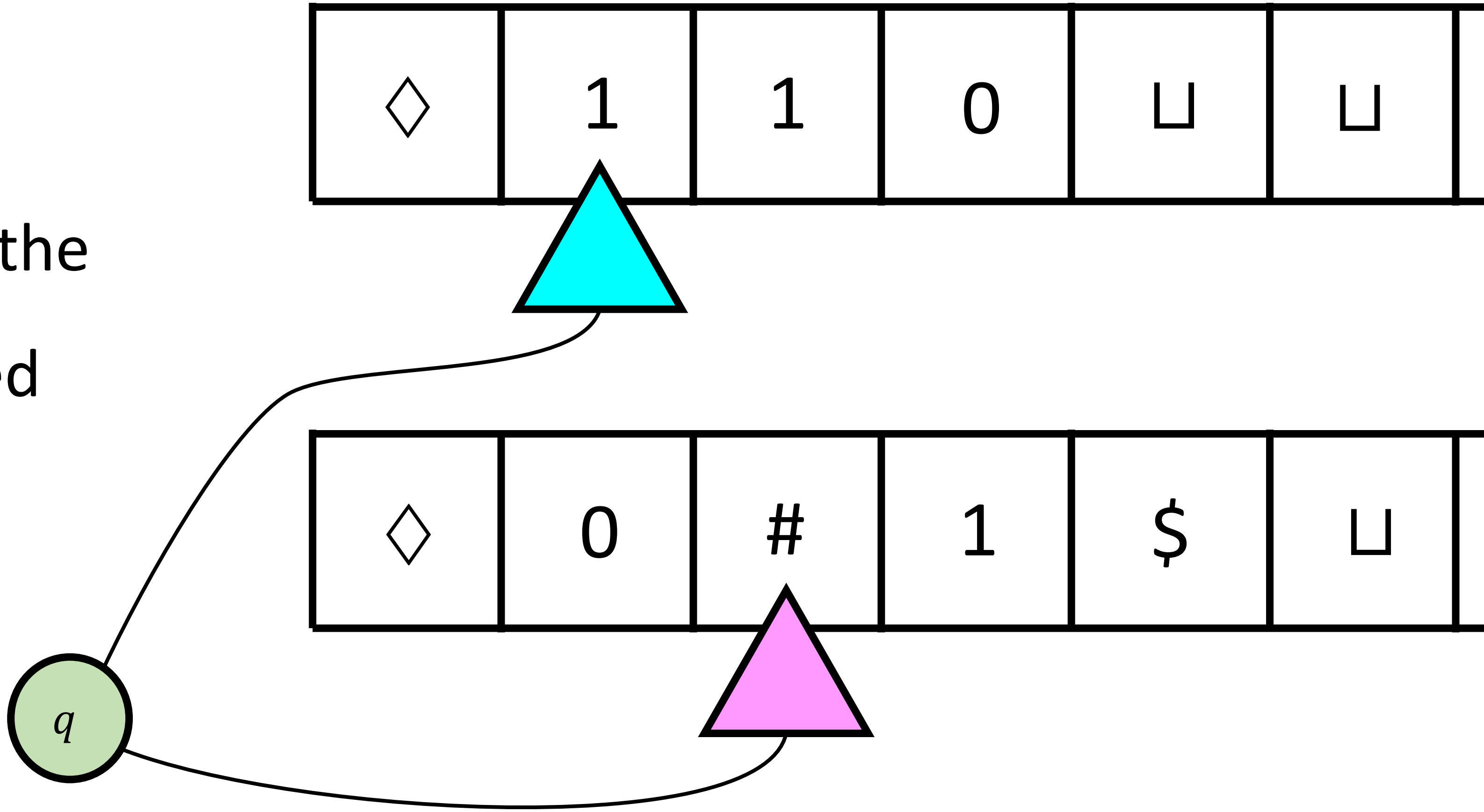
D: Store the location data in a single dedicated tape cell

Respond at PollEv.com/whoza or text "whoza" to 22333

Theorem: There exists a k-tape TM that decides L if and only if there exists a 1-tape TM that decides L

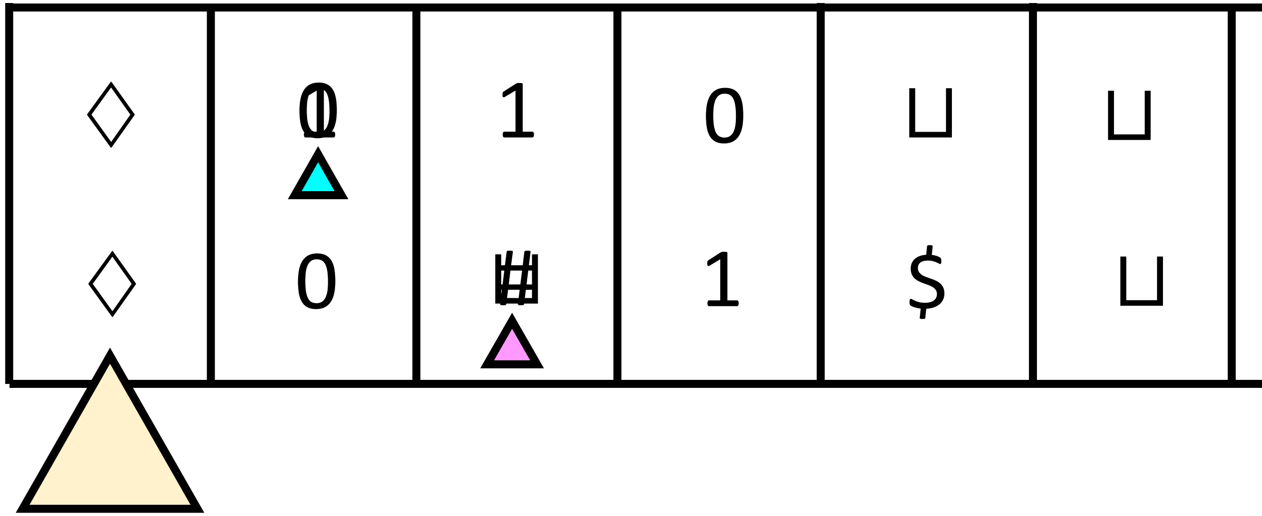
Simulating k tapes with 1 tape

- Idea: Pack a bunch of data into each cell
- Store "simulated heads" on the tape, along with k "simulated symbols" in each cell



Simulating k tapes with 1 tape

- Idea: Pack a bunch of data into each cell
- Store "simulated heads" on the tape, along with k "simulated symbols" in each cell



- The one "real head" will scan back and forth, updating the simulated heads' locations and the simulated tape contents. (Details on the next slides)

Simulating k tapes with 1 tape

- Let $M = (Q, \Sigma, \Gamma, \Diamond, \sqcup, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ be a k-tape Turing machine that decides L
- We will define a 1-tape Turing machine

$$M' = (Q', \Sigma, \Gamma', \Diamond, \sqcup, \delta', q_0', q_{\text{accept}}, q_{\text{reject}})$$

that also decides L

Simulating k tapes with 1 tape: Alphabet

- Let $\Lambda = \Gamma \cup \{\underline{b} : b \in \Gamma\}$, i.e., two disjoint copies of Γ
  - Interpretation: An underline indicates the presence of a simulated head
- New alphabet: $\Gamma' = \{\Diamond, \sqcup\} \cup \left\{ \begin{pmatrix} b_1 \\ \vdots \\ b_k \end{pmatrix} : b_1, \ldots, b_k \in \Lambda \right\}$
  - Interpretation: One symbol in Γ' is one "simulated column" of M
- Identify each input symbol $b \in \Sigma$ with the new symbol $\begin{pmatrix} b \\ \sqcup \\ \vdots \\ \sqcup \end{pmatrix}$, so $\Sigma \subseteq \Gamma'$

Simulating k tapes with 1 tape: Head statuses

- At each moment, each simulated head will have one of the following statuses:
  - "$\rightarrow b, D$" where $b \in \Gamma$ and $D \in \{\mathrm{L}, \mathrm{R}, \mathrm{S}\}$
    - Interpretation: The simulated head needs to write b and move in direction D
  - "👻"
    - Interpretation: The simulated head is not currently depicted on the real tape; the simulated head's location is currently the same as the real head's location
  - "$b \rightarrow$" where $b \in \Gamma$
    - Interpretation: In the next simulated step, the simulated head will read b

Simulating k tapes with 1 tape: Head statuses

- Let Ω be the set of all possible statuses for a single simulated head:

$$\Omega = \{\text{“} \rightarrow b, D\text{”} : b \in \Gamma, D \in \{\mathrm{L}, \mathrm{R}, \mathrm{S}\}\}$$

$$\cup \{\text{“👻”}\}$$

$$\cup \{\text{“}b \rightarrow \text{ ”} : b \in \Gamma\}$$

Simulating k tapes with 1 tape: States

- New state set:

$$Q' = \{q_{\text{accept}}, q_{\text{reject}}\} \cup \left\{ \begin{pmatrix} s_1 \\ \vdots \\ s_k \end{pmatrix}_{q,D} : s_1, \dots, s_k \in \Omega; \quad q \in Q; \quad D \in \{\mathrm{L}, \mathrm{R}\} \right\}$$

- Interpretation:
  - Simulated head j has status s_j
  - The simulated machine is in state q
  - The one real head is making a pass over the tape in direction D

Simulating k tapes with 1 tape: Start state

- New start state:

$$q_0' = \begin{pmatrix} \text{“👻”} \\ \vdots \\ \text{“👻”} \end{pmatrix}_{q_0,\mathrm{R}}$$

Simulating k tapes with 1 tape: Transitions

- The new transition function will have the form

$$\delta' : Q' \times \Gamma' \to Q' \times \Gamma' \times \{\mathrm{L}, \mathrm{R}\}$$

Simulating k tapes with 1 tape: Transitions

- Let $\delta'\left(\begin{pmatrix} s_1 \\ \vdots \\ s_k \end{pmatrix}_{q,D}, \begin{pmatrix} b_1 \\ \vdots \\ b_k \end{pmatrix}\right) = \left(\begin{pmatrix} s_1' \\ \vdots \\ s_k' \end{pmatrix}_{q,D}, \begin{pmatrix} b_1' \\ \vdots \\ b_k' \end{pmatrix}, D\right)$ where s_j', b_j' are defined by:

- If $s_j =$ "👻": Let $b_j' = \underline{b_j}$ and $s_j' =$ "$b_j \rightarrow$"
- If $s_j =$ "$\rightarrow c_j, S$" and b_j has an underline: Let $b_j' = \underline{c_j}$ and $s_j' =$ "$c_j \rightarrow$"
- If $s_j =$ "$\rightarrow c_j, D$" and b_j has an underline: Let $b_j' = c_j$ and $s_j' =$ "👻"
- In all other cases: Let $b_j' = b_j$ and $s_j' = s_j$

Simulating k tapes with 1 tape: Transitions

- Let $\delta'\left(\begin{pmatrix} s_1 \\ \vdots \\ s_k \end{pmatrix}_{q,\mathrm{R}}, \ \sqcup\right) = \left(\begin{pmatrix} s_1' \\ \vdots \\ s_k' \end{pmatrix}_{q,\mathrm{L}}, \begin{pmatrix} b_1' \\ \vdots \\ b_k' \end{pmatrix}, \mathrm{L}\right)$ where s_j', b_j' are defined by:

- If $s_j =$ "👻": Let $b_j' = \underline{\sqcup}$ and $s_j' =$ " $\sqcup \rightarrow$ "

- In all other cases: Let $b_j' = \sqcup$ and $s_j' = s_j$

Simulating k tapes with 1 tape: Transitions

- What do we do when we see $\Diamond$? Let $s_1, \ldots, s_k \in \Omega$ (head statuses) and let $q \in Q$
- Assume that $\forall j$, either $s_j =$ "$b_j \rightarrow$ " or $s_j =$ "👻". In the latter case, let $b_j = \Diamond$
- Let $(q', c_1, \ldots, c_k, D_1, \ldots, D_k) = \delta(q, b_1, \ldots, b_k)$
- If $s_j =$ "$b_j \rightarrow$ ", let $s_j' =$ " $\rightarrow c_j, D_j$". If $s_j =$ "👻", let $s_j' =$ "👻"
- Let $\delta'\left(\begin{pmatrix} s_1 \\ \vdots \\ s_k \end{pmatrix}_{q,\mathrm{L}}, \Diamond\right) = q'$ if q' is a halting state and $\left(\begin{pmatrix} s_1' \\ \vdots \\ s_k' \end{pmatrix}_{q',\mathrm{R}}, \Diamond, \mathrm{R}\right)$ otherwise