CMSC 28100

Introduction to Complexity Theory

Autumn 2025

Instructor: William Hoza

The Church-Turing Thesis

- Let $Y \subseteq \{0,1\}^*$

Church-Turing Thesis:

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

← Intuitive notion

← Mathematically precise notion

The Church-Turing Thesis

- The Church-Turing thesis says:
  - The Turing machine model is a "correct" way of modeling arbitrary computation
  - The informal concept of an "algorithm" is successfully captured by the rigorous definition of a Turing machine
- Consequence: It is really, truly impossible to design an algorithm that decides $\mathrm{SELF\text{-}REJECTORS}$ or any other undecidable language!

Are Turing machines powerful enough?

- **OBJECTION:** “To encompass all possible algorithms, we should add various bells and whistles to the Turing machine model.”
- Example: Left-Right-Stationary Turing Machine: Like an ordinary Turing machine, except it has a transition function $\delta: Q \times \Sigma \to Q \times \Sigma \times \{\mathrm{L}, \mathrm{R}, \mathrm{S}\}$
- S means the head does not move in this step

Left-right-stationary Turing machines

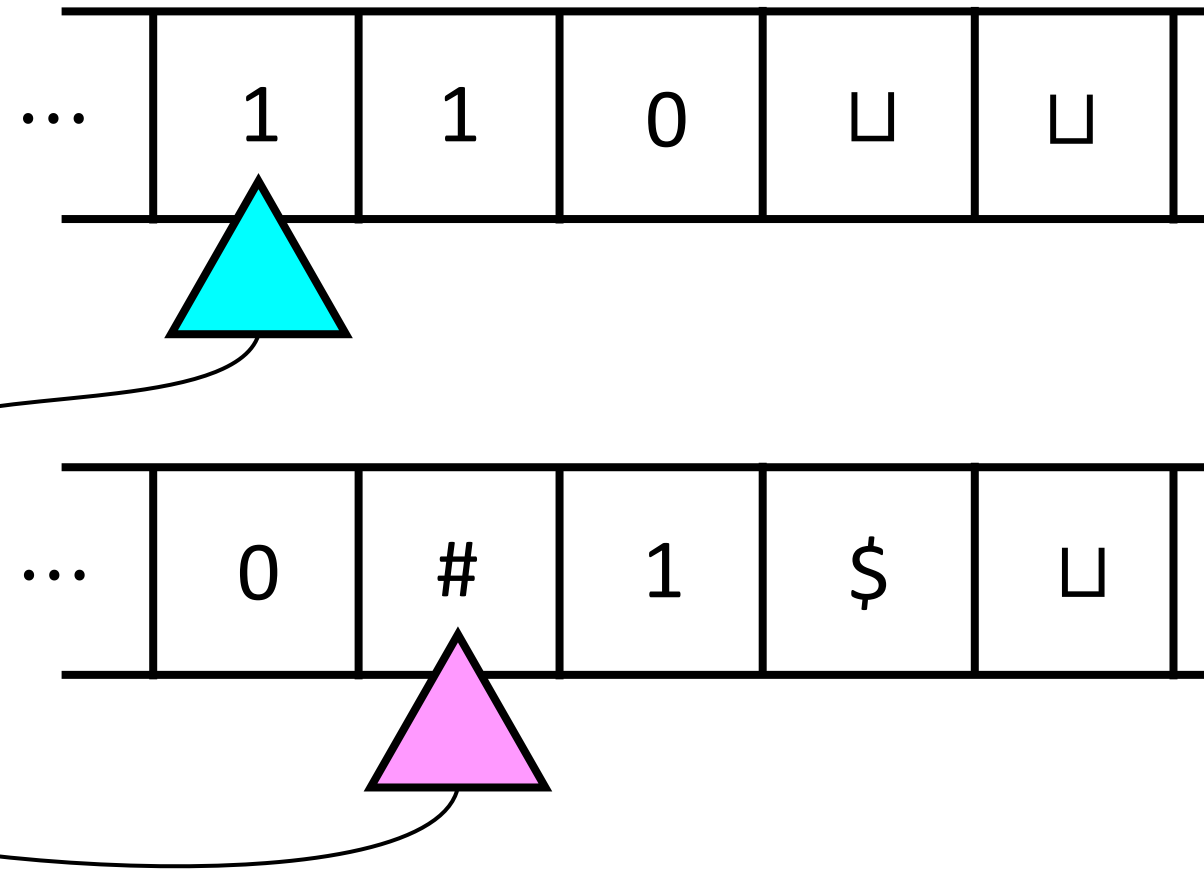

- Let Y be a language
- We proved:

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

Multi-tape Turing machines

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

  $\delta: Q \times \Sigma^k \to Q \times \Sigma^k \times \{\mathrm{L}, \mathrm{R}, \mathrm{S}\}^k$
- (Exercise: Rigorously define acceptance, rejection, etc.)



Multi-tape Turing machines

- Let k be any positive integer and let Y be a language

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

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

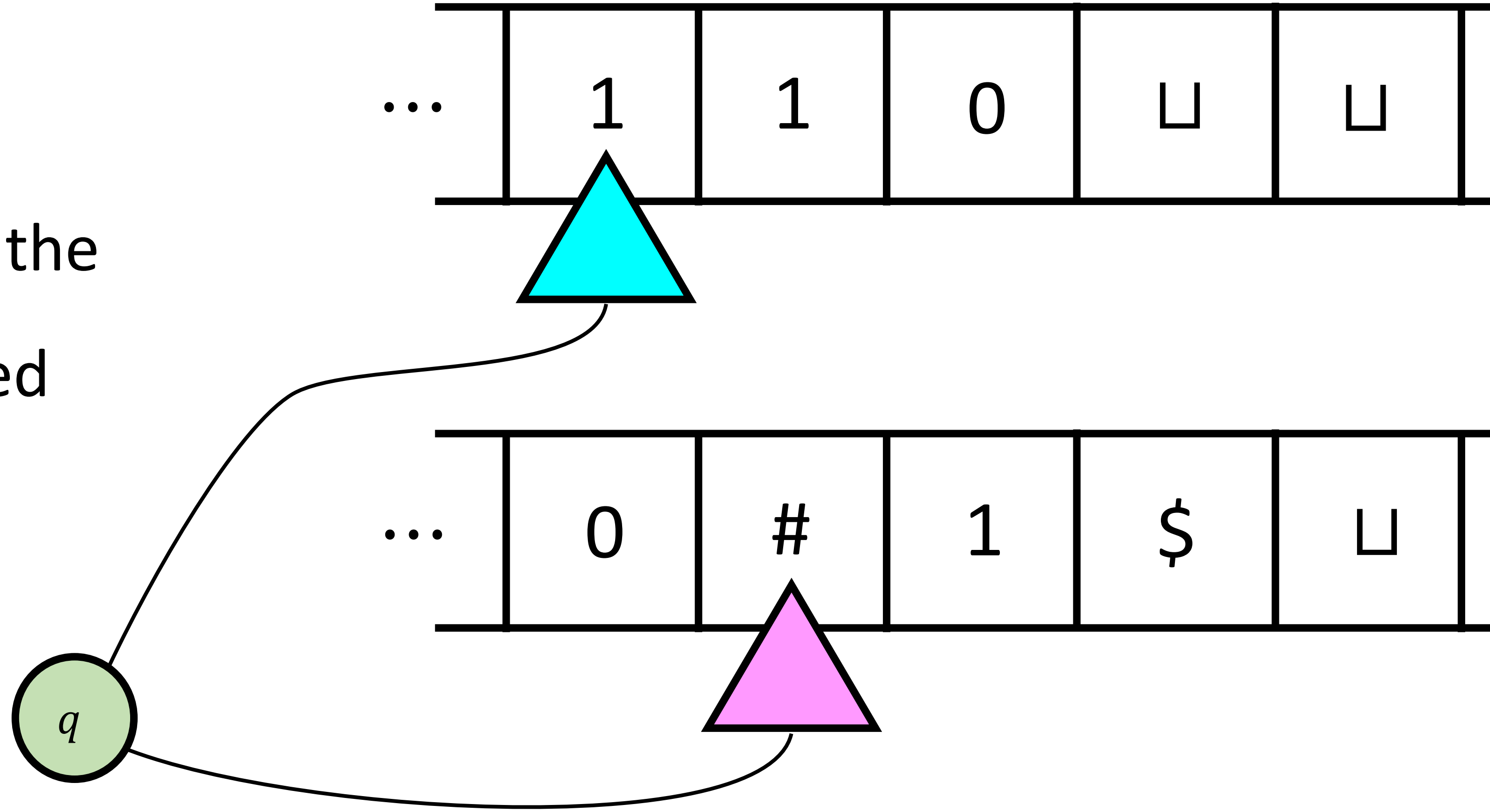
D: Store the location data in a single dedicated tape cell

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

Proof on upcoming 12 slides

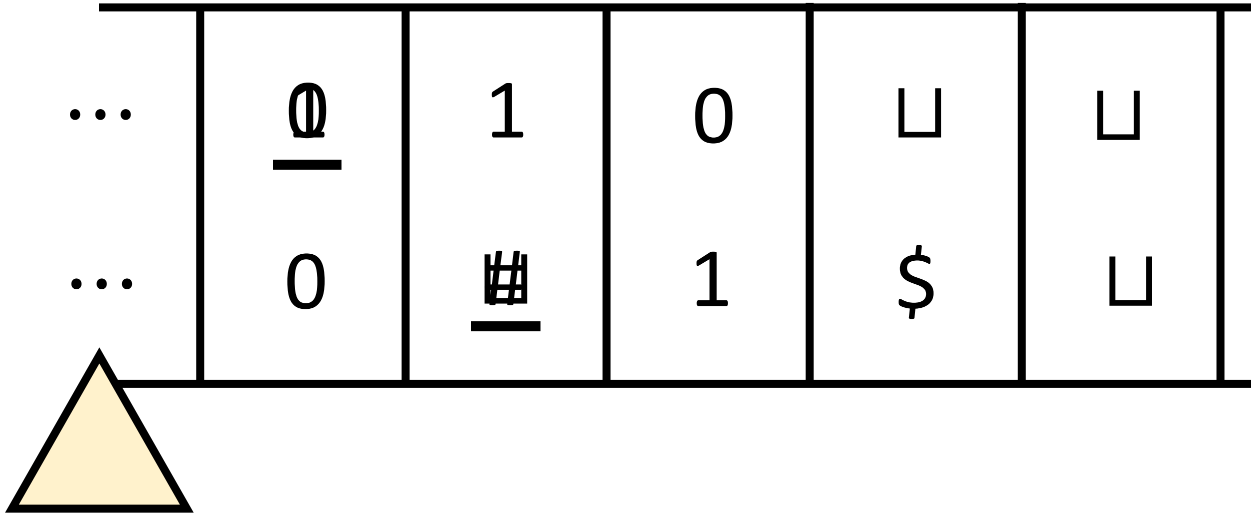
Simulating k tapes with 1 tape

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

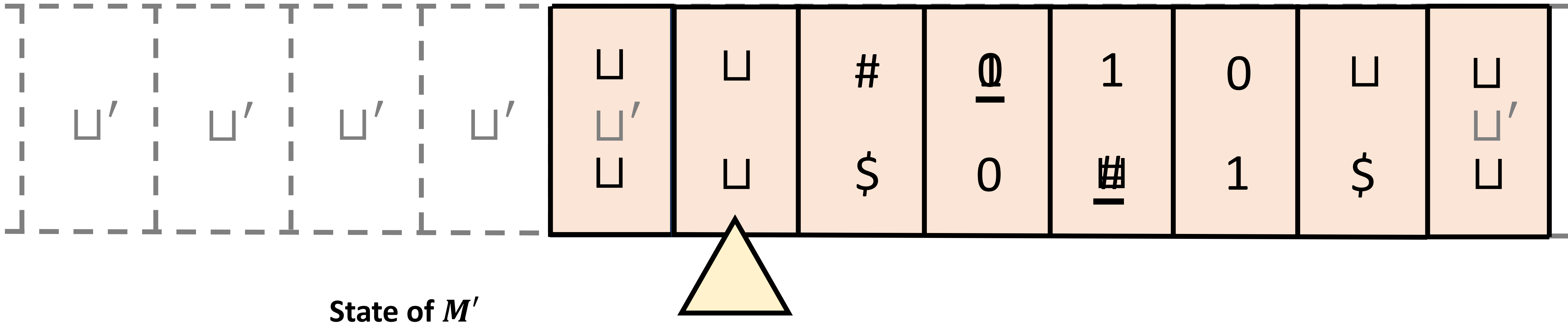
$$M' = (Q', q_0', q_{\text{accept}}', q_{\text{reject}}', \Sigma', \sqcup', \delta')$$

that also decides Y

Simulating k tapes with 1 tape: Alphabet

- Let $\Gamma = \Sigma \cup \{\underline{b} : b \in \Sigma\}$, i.e., two disjoint copies of Σ
  - An underline represents a simulated head
- New alphabet: $\Sigma' = \{\sqcup'\} \cup \left\{ \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix} : b_1, \dots, b_k \in \Gamma \right\}$
  - One symbol in Σ' is one "simulated column" of M
- Technicality: Encode input over the alphabet $\left\{ \begin{bmatrix} 0 \\ \sqcup \\ \vdots \\ \sqcup \end{bmatrix}, \begin{bmatrix} 1 \\ \sqcup \\ \vdots \\ \sqcup \end{bmatrix} \right\}$ instead of $\{0, 1\}$

Simulating 2 tapes with 1 tape: States



State of M'

State of M →	q'	R	← Direction of M' motion
Head 1 symbol →	1	(0, L)	← Head 1 instruction
Head 2 symbol →	0	(#, R)	← Head 2 instruction

Simulating k tapes with 1 tape: States

- New state set:

$$Q' = \left\{ \begin{array}{|c|c|} \hline q & D \\ \hline b_1 & \sigma_1 \\ \vdots & \vdots \\ b_k & \sigma_k \\ \hline \end{array} \; : \; \begin{array}{l} q \in Q \\ D \in \{\mathrm{L}, \mathrm{R}\} \\ b_1, \dots, b_k \in \Sigma \cup \{?\} \\ \sigma_1, \dots, \sigma_k \in (\Sigma \times \{\mathrm{L}, \mathrm{R}, \mathrm{S}\}) \cup \{?\} \end{array} \right\}$$

Simulating k tapes with 1 tape: Start state

- New start state:

$q_0' =$

q_0	L
?	?
⋮	⋮
?	?

Simulating k tapes with 1 tape: Transitions

$$\delta'\left(\begin{array}{|c|c|}\hline q & D \\ \hline b_1 & \sigma_1 \\ \vdots & \vdots \\ b_k & \sigma_k \\ \hline \end{array}, \begin{array}{|c|}\hline c_1 \\ \vdots \\ c_k \\ \hline \end{array}\right) = \left(\begin{array}{|c|c|}\hline q & D \\ \hline b_1' & \sigma_1' \\ \vdots & \vdots \\ b_k' & \sigma_k' \\ \hline \end{array}, \begin{array}{|c|}\hline c_1' \\ \vdots \\ c_k' \\ \hline \end{array}, D\right)$$

- If $\sigma_j = (a, D)$ and $c_j = \underline{b_j}$: Let $b_j' = ?$, $\sigma_j' = ?$, $c_j' = a$
- If $\sigma_j = (a, \mathrm{S})$ and $c_j = \underline{b_j}$: Let $b_j' = a$, $\sigma_j' = ?$, $c_j' = \underline{a}$
- If $\sigma_j = ?$ and $b_j = ?$: Let $b_j' = c_j$, $\sigma_j' = ?$, $c_j' = \underline{c_j}$
- In all other cases: Let $b_j' = b_j$, $\sigma_j' = \sigma_j$, $c_j' = c_j$

Simulating k tapes with 1 tape: Transitions

$$\delta'\left(\begin{array}{|c|c|}\hline q & \mathrm{R} \\ \hline b_1 & \sigma_1 \\ \vdots & \vdots \\ b_k & \sigma_k \\ \hline \end{array}\ ,\ \sqcup'\right) = \left(\begin{array}{|c|c|}\hline q & \mathrm{L} \\ \hline b_1' & \sigma_1' \\ \vdots & \vdots \\ b_k' & \sigma_k' \\ \hline \end{array}\ ,\ \begin{array}{|c|}\hline c_1' \\ \vdots \\ c_k' \\ \hline \end{array}\ ,\ \mathrm{L}\right)$$

- If $\sigma_j = ?$ and $b_j = ?$: Let $b_j' = \sqcup$, $\sigma_j' = ?$, $c_j' = \underline{\sqcup}$
- In all other cases: Let $b_j' = b_j$, $\sigma_j' = \sigma_j$, $c_j' = \sqcup$

Simulating k tapes with 1 tape: Transitions

$$\delta'\left(\begin{array}{|c|c|}\hline q & \mathrm{L} \\ \hline b_1 & \sigma_1 \\ \vdots & \vdots \\ b_k & \sigma_k \\ \hline \end{array}, \sqcup'\right) = \left(\begin{array}{|c|c|}\hline q' & \mathrm{R} \\ \hline b_1' & \sigma_1' \\ \vdots & \vdots \\ b_k' & \sigma_k' \\ \hline \end{array}, \begin{array}{|c|}\hline c_1' \\ \vdots \\ c_k' \\ \hline \end{array}, \mathrm{R}\right)$$

- Let $(q', a_1, \dots, a_k, D_1, \dots, D_k) = \delta(q, b_1, \dots, b_k)$, treating $b_j = ?$ as $b_j = \sqcup$
- If q' is a halting state: Let $b_j' = ?$, $\sigma_j' = ?$, $c_j' = \sqcup$
- If $\sigma_j = ?$ and $b_j = ?$: Let $b_j' = \sqcup$, $\sigma_j' = (a_j, D_j)$, $c_j' = \underline{\sqcup}$
- In all other cases: Let $b_j' = b_j$, $\sigma_j' = (a_j, D_j)$, $c_j' = \sqcup$

Simulating k tapes with 1 tape: Halting states

$q'_{\text{accept}} =$

q_{accept}	R
?	?
⋮	⋮
?	?

$q'_{\text{reject}} =$

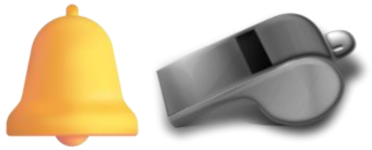
q_{reject}	R
?	?
⋮	⋮
?	?

Simulating k tapes with 1 tape

- That completes the definition of M'
- Exercise: Rigorously prove that M' decides the language Y

TMs can simulate all “reasonable” machines

- We could add various other bells and whistles to the basic TM model
  - The ability to observe the two neighboring cells
  - The ability to “teleport” back to the initial cell in a single step
  - A two-dimensional tape



- None of these changes has any effect on the power of the model

The Church-Turing Thesis

- Let $Y \subseteq \{0, 1\}^*$

Church-Turing Thesis:

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

← Intuitive notion

← Mathematically precise notion