

# BU CS 332 – Theory of Computation

## Lecture 11:

- TM Variants
- Closure Properties

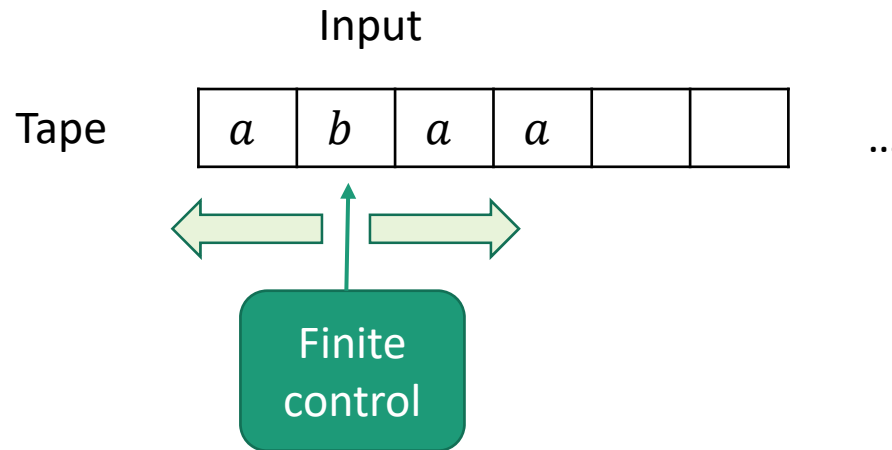
Reading:

Sipser Ch 3.2

HW 4 (on TMs)  
Due March 16th

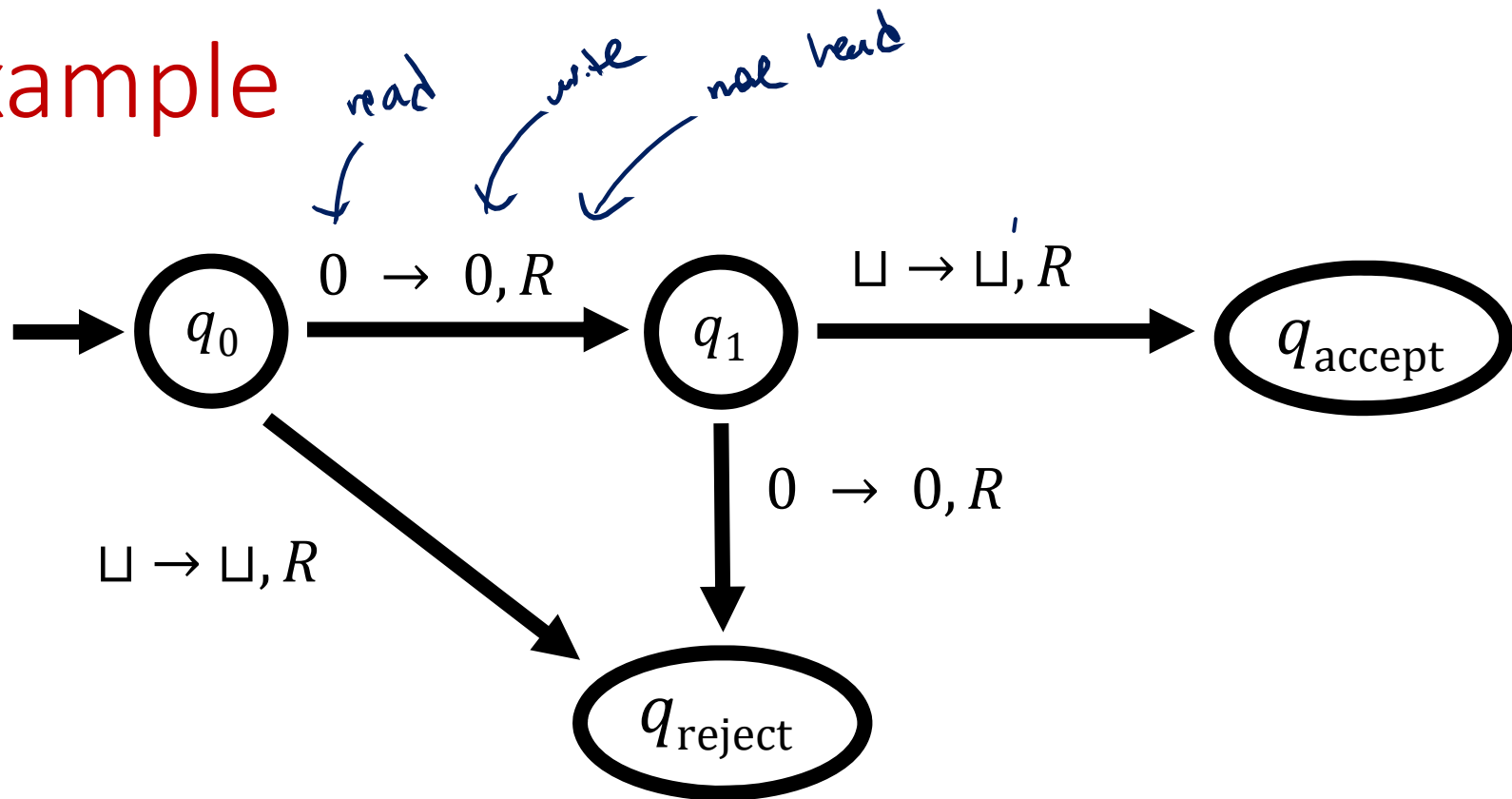
Mark Bun  
March 2, 2020

# The Basic Turing Machine (TM)



- Input is written on an infinitely long tape
- Head can both read and write, and move in both directions
- Computation halts when control reaches “accept” or “reject” state

# Example



# Formal Definition of a TM

deterministic

A TM is a 7-tuple  $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$

- $Q$  is a finite set of states
- $\Sigma$  is the input alphabet (does **not** include  $\sqcup$ )
- $\Gamma$  is the tape alphabet (contains  $\sqcup$  and  $\Sigma$ )  $\Sigma \subseteq \Gamma, \sqcup \in \Gamma$
- $\delta$  is the transition function

...more on this later

- $q_0 \in Q$  is the start state
- $q_{\text{accept}} \in Q$  is the accept state
- $q_{\text{reject}} \in Q$  is the reject state ( $q_{\text{reject}} \neq q_{\text{accept}}$ )

# TM Transition Function

$$\delta(q, a)$$

$$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$$

$L$  means “move left” and  $R$  means “move right”

$\delta(p, a) = (q, b, R)$  means:

- Replace  $a$  with  $b$  in current cell
- Transition from state  $p$  to state  $q$
- Move tape head right



$\delta(p, a) = (q, b, L)$  means:

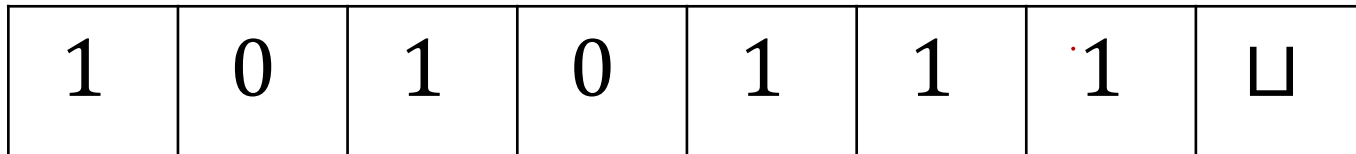
- Replace  $a$  with  $b$  in current cell
- Transition from state  $p$  to state  $q$
- Move tape head left UNLESS we are at left end of tape, in which case don't move

# Configuration of a TM

A string with captures the state of a TM together with the contents of the tape

*My convention:*

*1 0 1 q<sub>5</sub> 0 1 1 1*

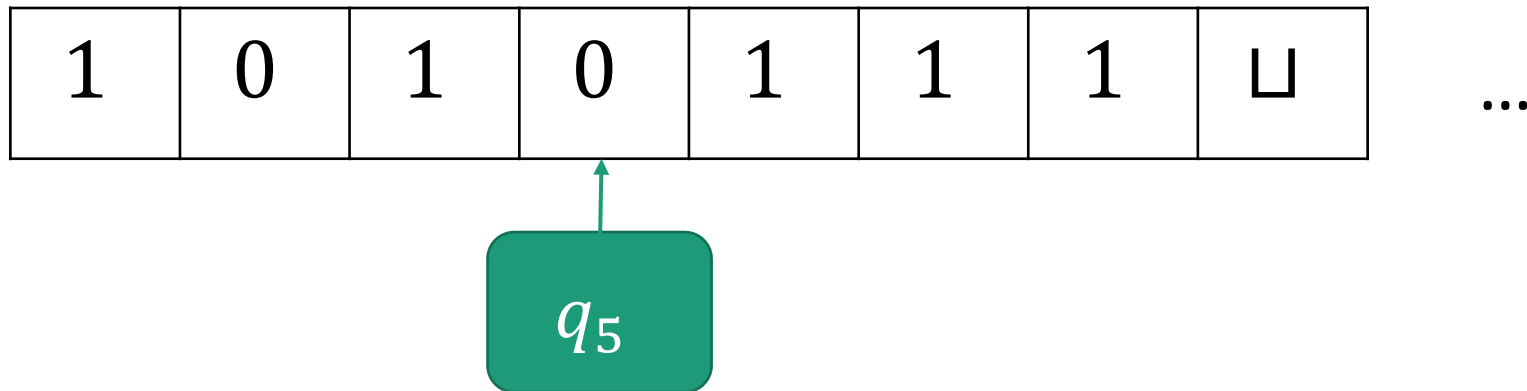


...

# Configuration of a TM: Formally

A **configuration** is a string  $uqv$  where  $q \in Q$  and  $u, v \in \Gamma^*$

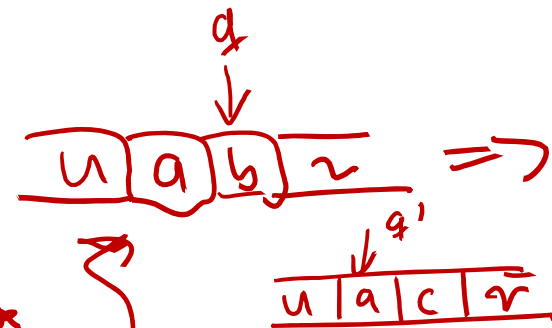
- Tape contents =  $uv$  (followed by blanks  $\sqcup$ )
- Current state =  $q$
- Tape head on first symbol of  $v$



# How a TM Computes

Input:  $w$

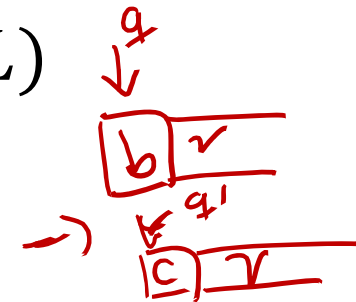
Start configuration:  $q_0 w$



One step of computation:

$u, v \in \Gamma^*$   
 $a, b \in \Gamma$

- $u a q b v$  yields  $u a c q' v$  if  $\delta(q, b) = (q', c, R)$
- $u a q b v$  yields  $u q' a c v$  if  $\delta(q, b) = (q', c, L)$
- $q b v$  yields  $q' c v$  if  $\delta(q, b) = (q', c, L)$   
At left end of tape



Accepting configuration:  $q = q_{\text{accept}}$

Rejecting configuration:  $q = q_{\text{reject}}$



# How a TM Computes

$M$  **accepts** input  $w$  if there is a sequence of configurations  $C_1, \dots, C_k$  such that:

- $C_1 = q_0w$  *input configuration*
- $C_i$  yields  $C_{i+1}$  for every  $i$  *can get from  $C_i$  to  $C_{i+1}$  in one step of TM computation*
- $C_k$  is an accepting configuration

$L(M)$  = the set of all strings  $w$  which  $M$  accepts

$A$  is **Turing-recognizable** if  $A = L(M)$  for some TM  $M$ :

- $w \in A \implies M$  halts on  $w$  in state  $q_{\text{accept}}$
- $w \notin A \implies M$  halts on  $w$  in state  $q_{\text{reject}}$  **OR**  
 $M$  runs forever

*TM's can loop forever*

# Recognizers vs. Deciders

$A$  decidable  $\Rightarrow A$  recognizable

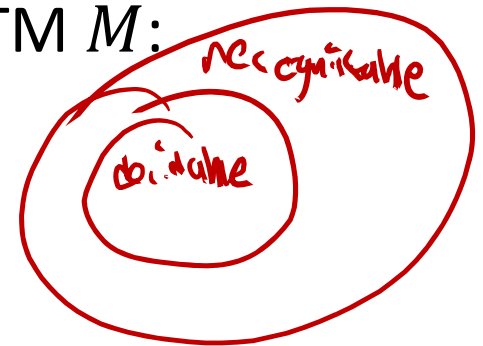
$\{\text{decidable languages}\} \subseteq$

$\{\text{recognizable languages}\}$

$L(M)$  = the set of all strings  $w$  which  $M$  accepts

$A$  is **Turing-recognizable** if  $A = L(M)$  for some TM  $M$ :

- $w \in A \Rightarrow M$  halts on  $w$  in state  $q_{\text{accept}}$
- $w \notin A \Rightarrow M$  halts on  $w$  in state  $q_{\text{reject}}$  **OR**  
 $M$  runs forever



$A$  is **(Turing-)decidable** if  $A = L(M)$  for some TM  $M$

which halts on every input

- $w \in A \Rightarrow M$  halts on  $w$  in state  $q_{\text{accept}}$
- $w \notin A \Rightarrow M$  halts on  $w$  in state  $q_{\text{reject}}$



# Back to Hilbert's Tenth Problem

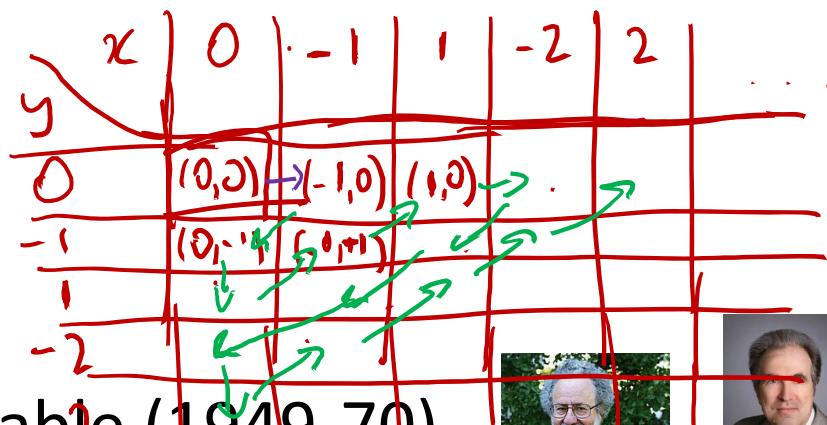
**Computational Problem:** Given a Diophantine equation, does it have a solution over the integers?

$$L = \{ p \mid p \text{ a polynomial w/ integer coeffs} \mid \exists z_1, \dots, z_n \in \mathbb{Z}^n \text{ s.t. } p(z_1, \dots, z_n) = 0 \}$$

- $L$  is Turing-recognizable

Design alg. accepts all  $p \in L$   
loop forever on all  $p \notin L$

Special case  
 $p(x, y)$



check  $p(0,0) \stackrel{?}{=} 0$   
if equal, accept,  
else move on

check  $p(-1,0) \stackrel{?}{=} 0$

check  $p(0,-1) \stackrel{?}{=} 0$

- $L$  is **not** decidable (1949-70)



# TM Variants

# How Robust is the TM Model?

Does changing the model result in different languages being recognizable / decidable?

So far we've seen...

- We can require that FAs/PDAs have a single accept state
- (CFGs can always be put in Chomsky Normal Form)
- Adding nondeterminism does not change the languages recognized by finite automata

Turing machines have an **astonishing** level of robustness

## Extensions that do not increase the power of the TM model

- TMs that are allowed to “stay put” instead of moving left or right

$$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, S\}$$

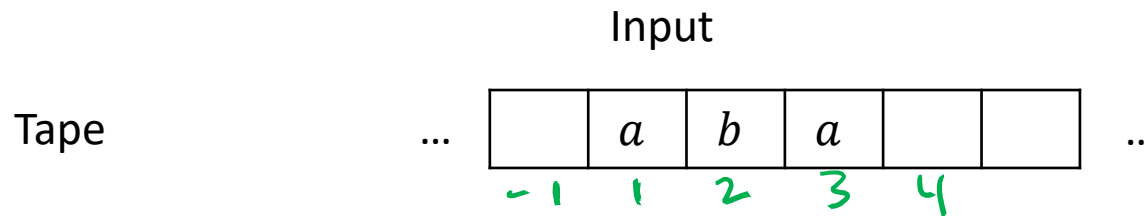
**Proof** that TMs with “stay put” are no more powerful:

**Simulation:** Convert any TM  $M$  with “stay put” into an equivalent TM  $M'$  without

Replace every “stay put” instruction in  $M$  with a move right instruction, followed by a move left instruction in  $M'$

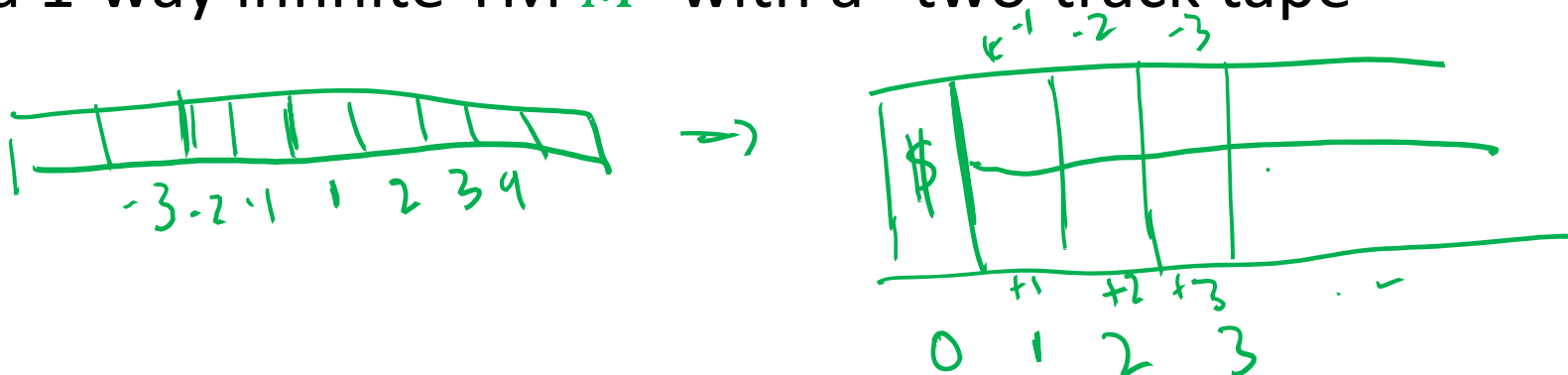
# Extensions that do not increase the power of the TM model

- TMs with a 2-way infinite tape, unbounded left to right



**Proof** that TMs with 2-way infinite tapes are no more powerful:

**Simulation:** Convert any TM  $M$  with 2-way infinite tape into a 1-way infinite TM  $M'$  with a “two-track tape”



# Formalizing the Simulation

$$M' = (Q', \Sigma, \Gamma', \delta', q'_0, q'_{\text{accept}}, q'_{\text{reject}})$$

**New tape alphabet:**  $\Gamma' = (\Gamma \times \Gamma) \cup \{\$\}$

**New state set:**  $Q' = Q \times \{+, -\}$

$(q, -)$  means “ $q$ , working on upper track”

$(q, +)$  means “ $q$ , working on lower track”

**New transitions:**

If  $\delta(p, a_-) = (q, b, L)$ , let  $\delta'((p, -), (a_-, a_+)) = ((q, -), (b, a_+), R)$

Also need new transitions for moving right, lower track, hitting \$,  
initializing input into 2-track format



# TMs are equivalent to...

- TMs with “stay put”
- TMs with 2-way infinite tapes
- Multi-tape TMs
- Nondeterministic TMs
- Random access TMs
- Enumerators
- Finite automata with access to an unbounded queue = 2-stack PDAs
- Primitive recursive functions
- Cellular automata
- ...

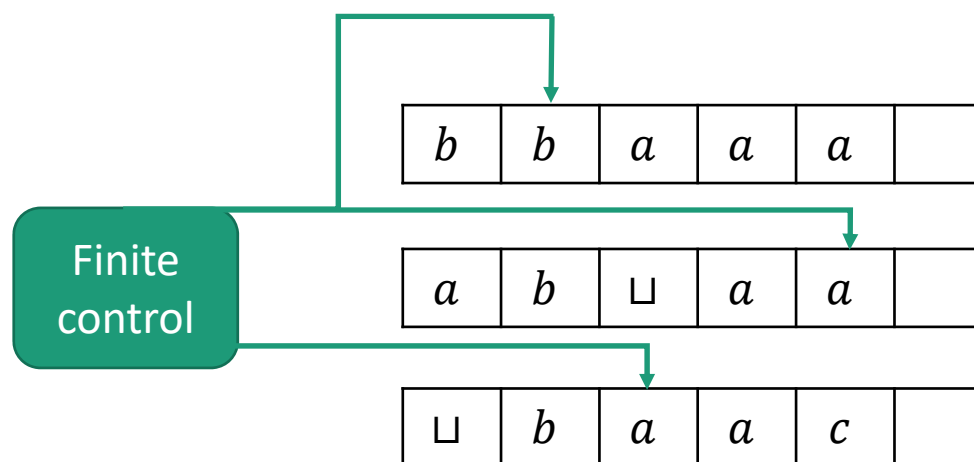
# Church-Turing Thesis

The equivalence of these models is a **mathematical theorem**

**Church-Turing *Thesis***: Each of these models captures our intuitive notion of algorithms

The Church-Turing Thesis is **not** a mathematical statement!

# Multi-Tape TMs



Fixed number of tapes  $k$  (can't change during computation)

Transition function  $\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R, S\}^k$

# Multi-Tape TMs are Equivalent to Single-Tape TMs

**Theorem:** Every  $k$ -tape TM  $M$  can be simulated by an equivalent single-tape TM  $M'$

