

# BU CS 332 – Theory of Computation

<https://forms.gle/Xj7E7pXaXdJZEeBc6>



## Lecture 9:

- Turing Machines

Reading:

Sipser Ch 3.1-3.3

Mark Bun & Alexander Poremba

February 26, 2026

# Turing Machines – Motivation

We've seen finite automata as a restricted model of computation

## Finite Automata / Regular Expressions

- Can do simple pattern matching (e.g., substrings), check parity, addition
- Can't perform unbounded counting  $\{0^n 1^n \mid n \geq 0\}$  not regular
- Can't recognize palindromes  $\{w \mid w^R = w\}$  not regular

Somewhat more powerful (not in this course):

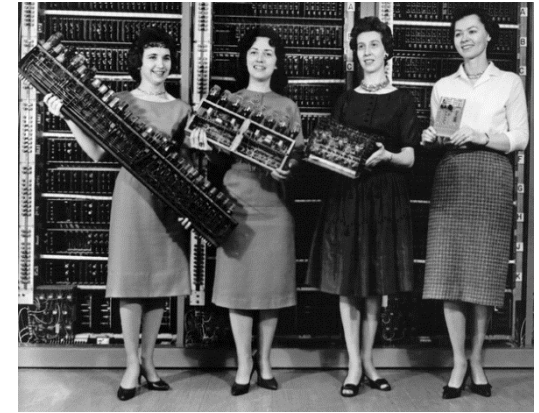
## Pushdown Automata / Context-Free Grammars

- Can count and compare, parse math expressions
- Can't recognize  $\{a^n b^n c^n \mid n \geq 0\}$

# Turing Machines – Motivation

## Goal:

Define a model of computation that is



- 1) **General purpose.** Captures all algorithms that can be implemented in any programming language.
- 2) **Mathematically simple.** We can hope to prove that things are not computable in this model.

# A Brief History

# 1900 – Hilbert's Tenth Problem

Ex: (Diophantine equation)

$$P(x, y, z) = 3xz - x^2y$$

Does there exist  $(x, y, z) \in \mathbb{Z}^3$  s.t.  
 $P(x, y, z) = 0$  ?

Solution:  $(x, y, z) = (3, 1, 1)$ .

*Given a Diophantine equation with any number of unknown quantities and with rational integral numerical coefficients: To devise a process according to which it can be determined in a finite number of operations whether the equation is solvable in rational integers.*

algorithm



David Hilbert 1862-1943

# 1928 – The *Entscheidungsproblem*



Wilhelm Ackermann 1896-1962

*The “Decision Problem”*

*Is there an algorithm which takes as input a formula (in first-order logic) and decides whether it's logically valid?*

Given: Mathematical statement

Output: Is the statement true or false?



David Hilbert 1862-1943

# 1936 – Solution to the *Entscheidungsproblem*



Alonzo Church 1903-1995

"An unsolvable problem of elementary number theory"

**Model of computation:**  $\lambda$ -calculus (CS 320)



Alan Turing 1912-1954

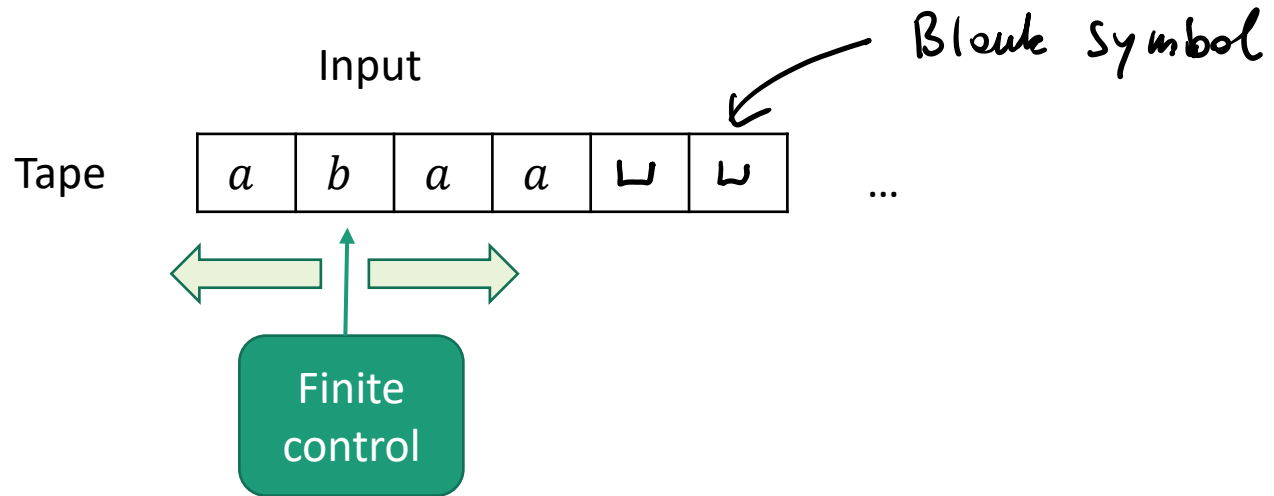
"On computable numbers, with an application to the *Entscheidungsproblem*"

**Model of computation:** Turing Machine

# The Turing Machine Model

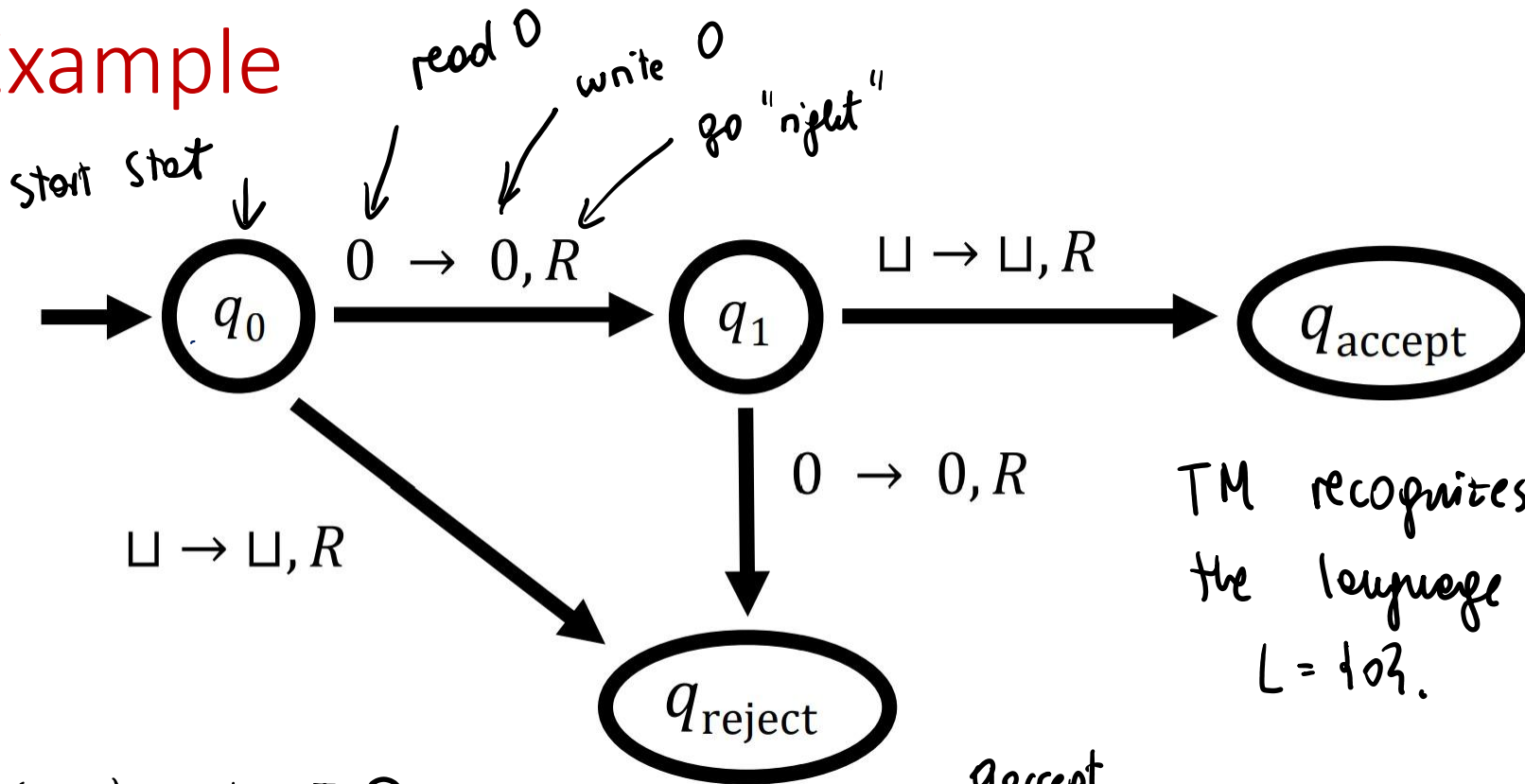
# The Basic Turing Machine (TM)

$$\Sigma = \{a, b\}$$



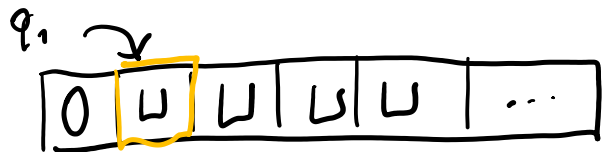
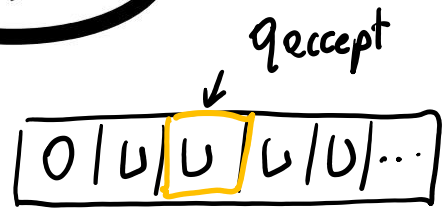
- Input is written on an infinitely long tape
- Head can both read and write, and move in both directions
- Computation halts as soon as control reaches “accept” or “reject” state

# Example



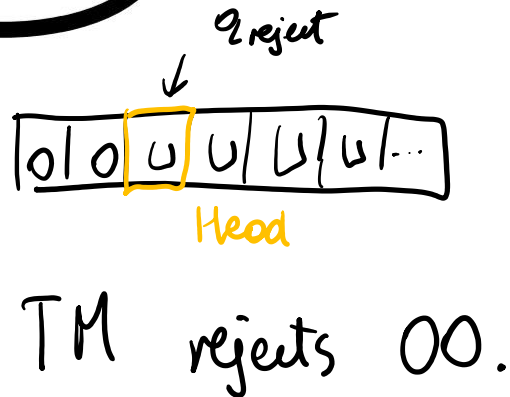
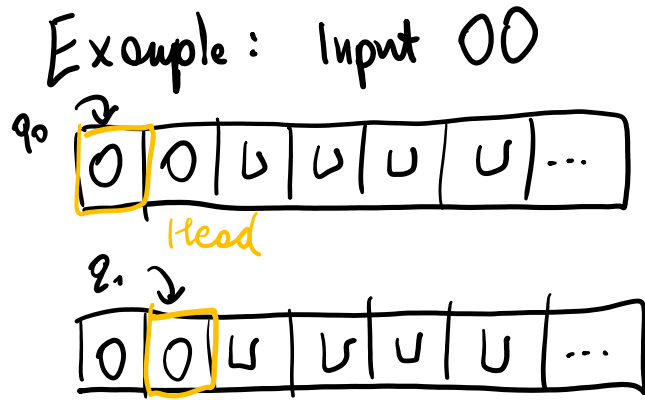
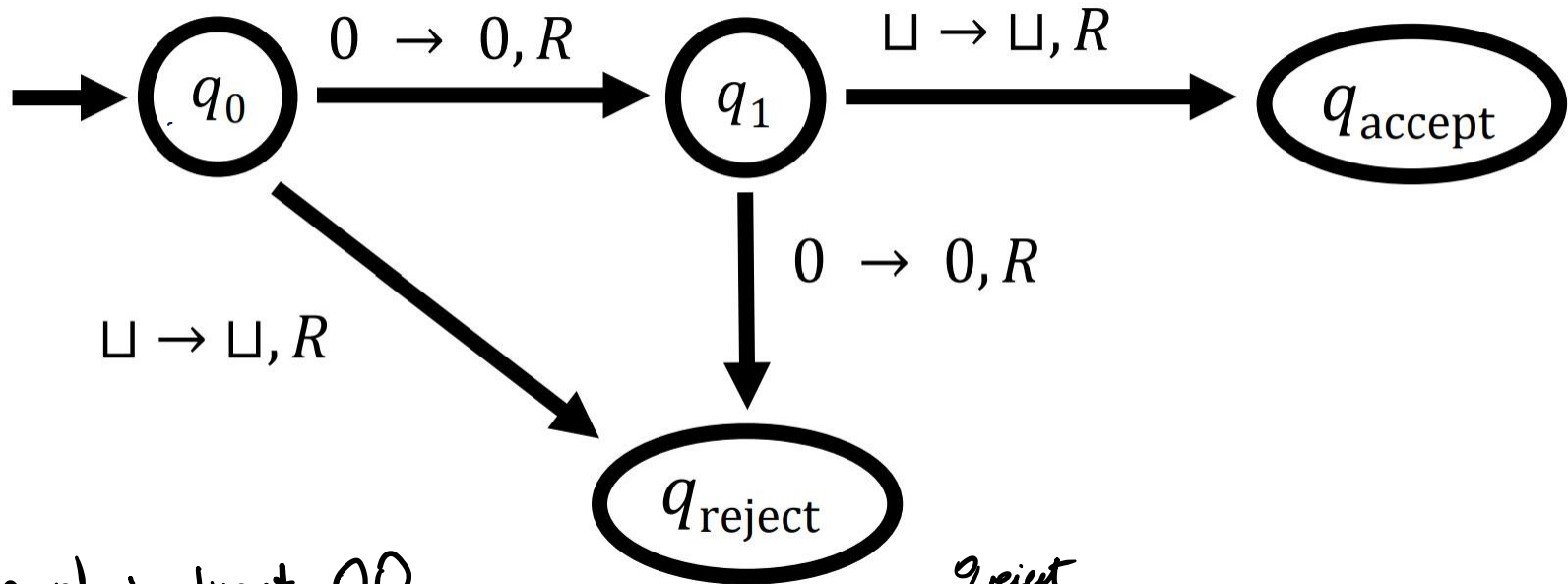
TM recognizes the language  $L = \{0\}$ .

Example: Input 0

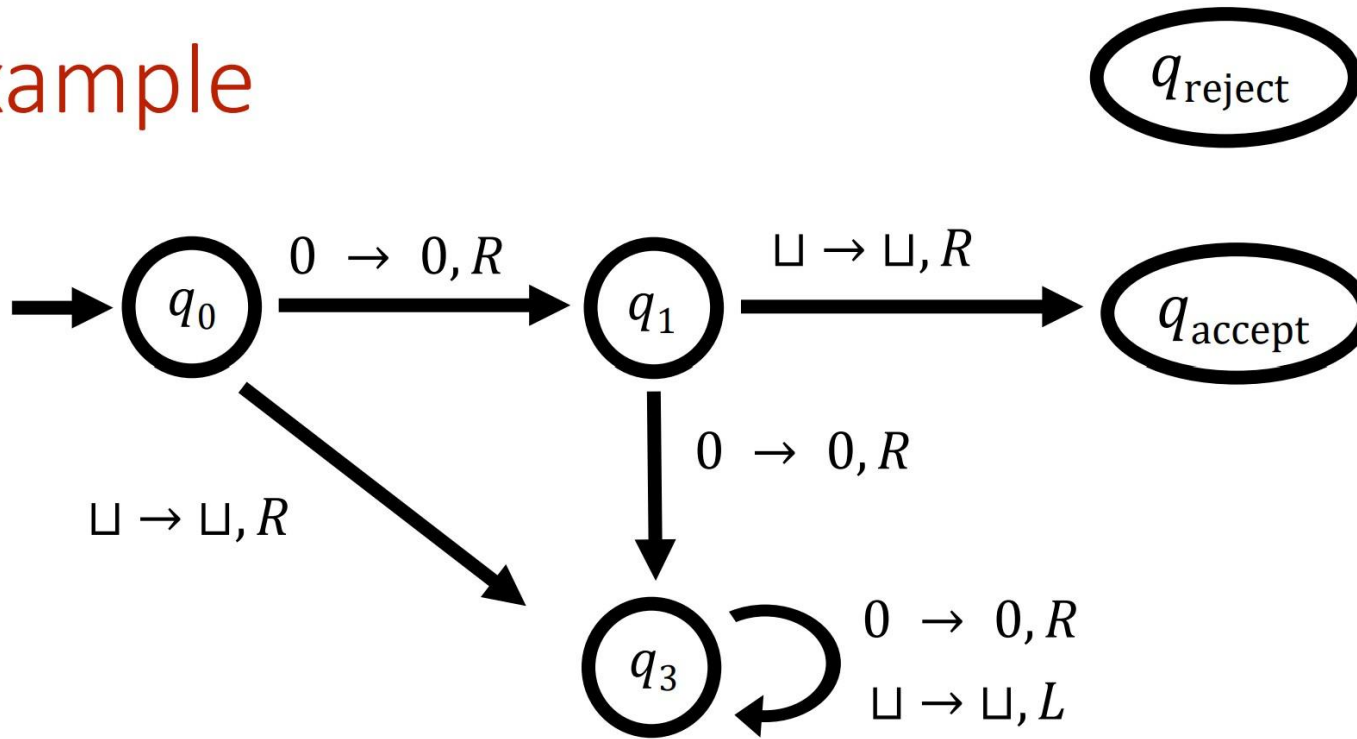


TM accepts input 0.

# Example



# Example



What does this TM do on input 000?

- a) Halt and accept
- b) Halt and reject
- c) Halt in state  $q_3$
- d) Loop forever without halting



# Three Levels of Abstraction

## High-Level Description

An algorithm (like CS 330)

Programming language:  
Python, Java, ...

## Implementation-Level Description

Describe (in English) the instructions for a TM

- How to move the head
- What to write on the tape

Assembly  
language

## Low-Level Description

State diagram or formal specification

machine  
language

# Example

$$\Sigma = \{0\}$$



Determine if a string  $w \in \{0\}^*$  is in the language

$$A = \{0^{2^n} \mid n \geq 0\}$$

Problem: Given a string  $w$  of all 0's, is its length a power of 2?

## High-Level Description

Repeat the following forever:

- If there is exactly one 0 in  $w$ , **accept**
- If there is an odd ( $> 1$ ) number of 0s in  $w$ , **reject**
- Delete half of the 0s in  $w$

# Example

Determine if a string  $w \in \{0\}^*$  is in the language

$$A = \{0^{2^n} \mid n \geq 0\}$$

Input: 

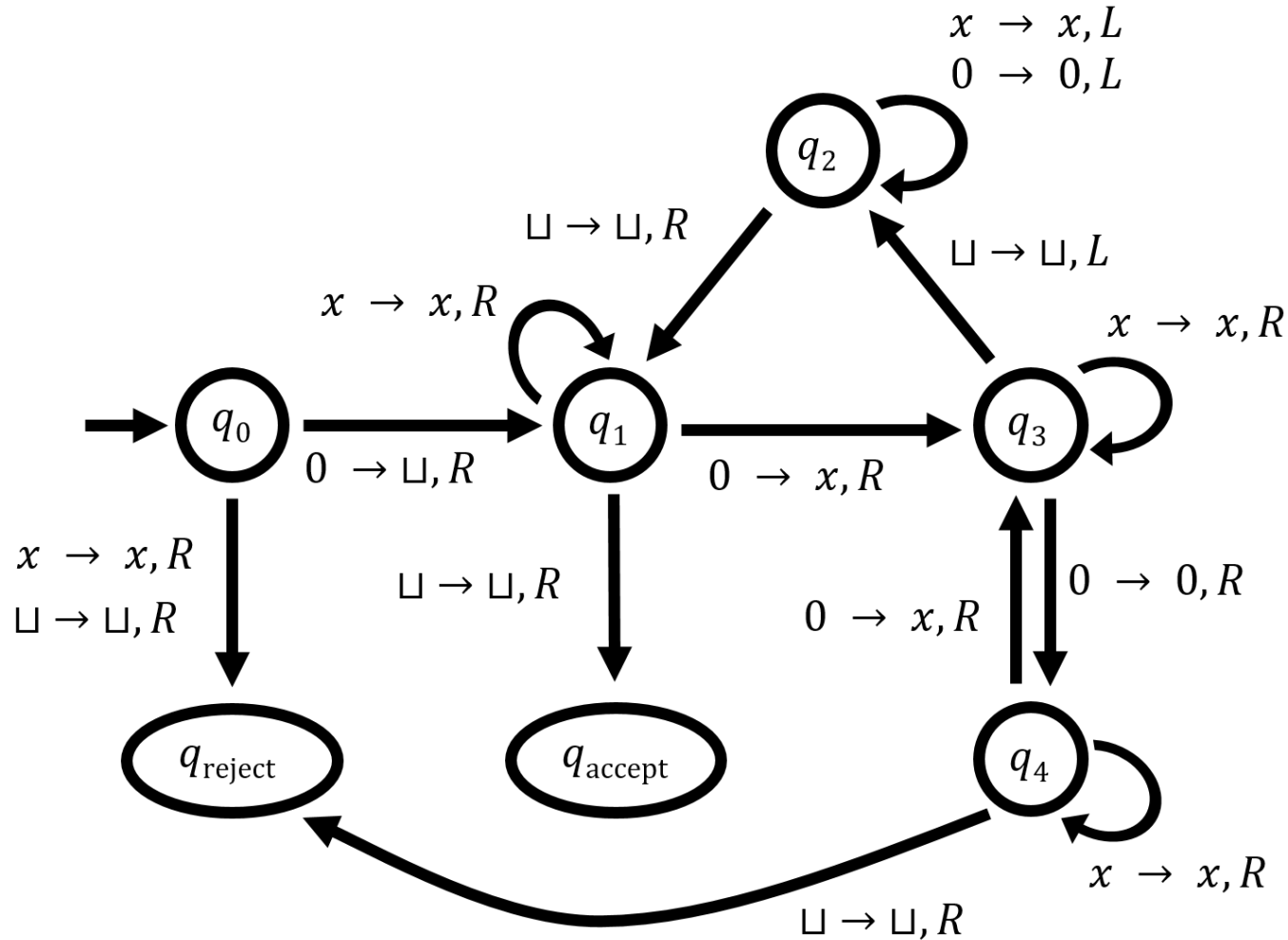
## Implementation-Level Description

1. While moving the tape head left-to-right:
  - a) Cross off every other 0
  - b) If there is exactly one 0 when we reach the first blank symbol, **accept**
  - c) If there is an odd ( $> 1$ ) number of 0s when we reach first blank symbol, **reject**
2. Return the head to the left end of the tape
3. Go back to step 1

# Example

Determine if a string  $w \in A = \{0^{2^n} \mid n \geq 0\}$

## Low-Level Description



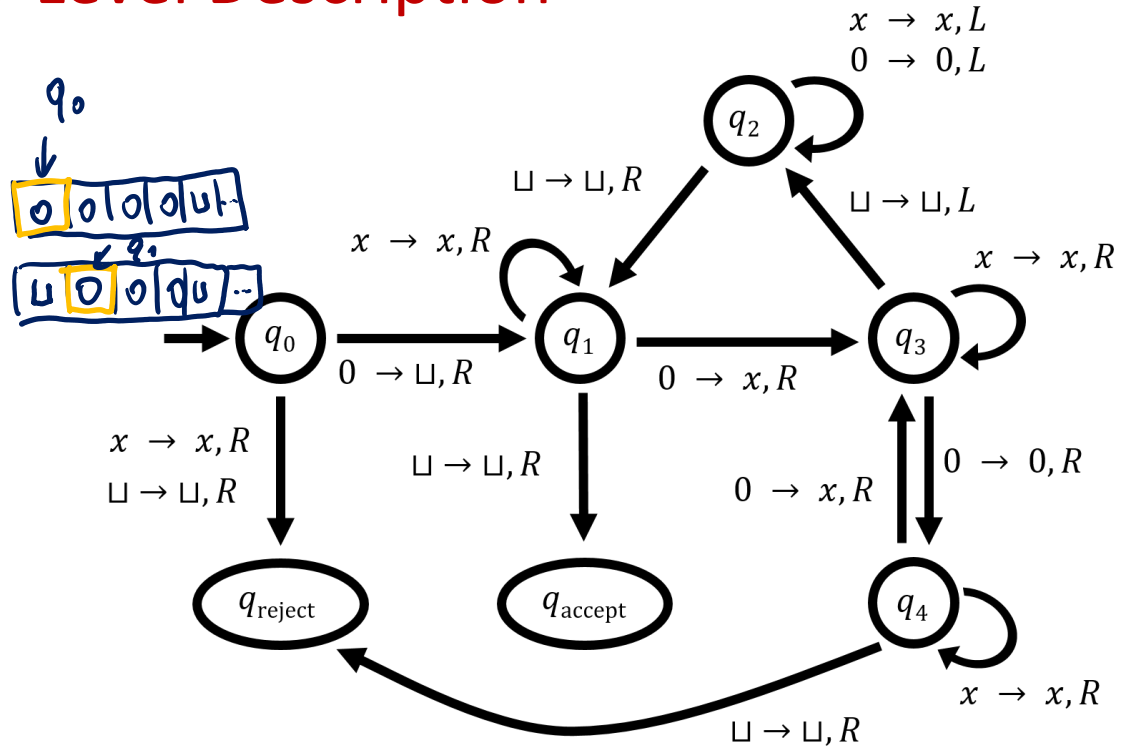
# Example

Determine if a string  $w \in A = \{0^{2^n} \mid n \geq 0\}$

## Low-Level Description

Input  $w = 0000$

$q_0 0 0 0 0 \sqcup \dots$   
 $\sqcup q_1 0 0 0 \sqcup \dots$  means



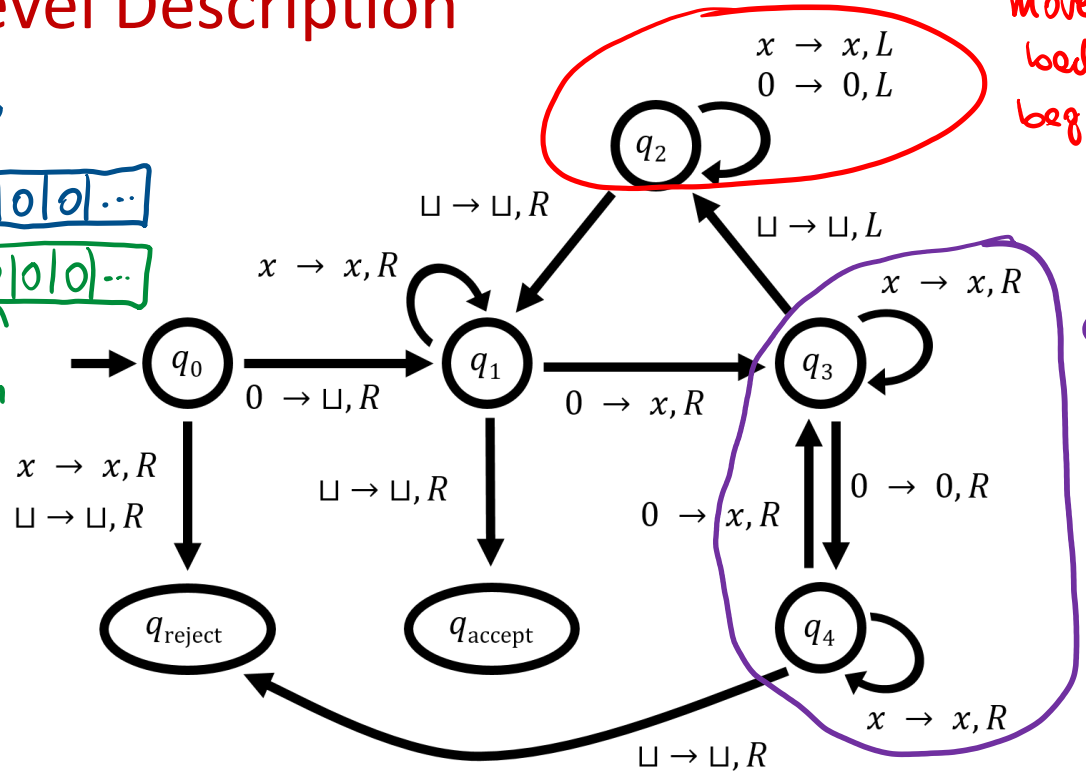
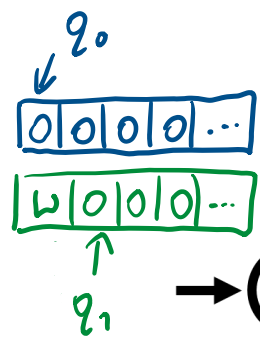
# Example

Determine if a string  $w \in A = \{0^{2^n} \mid n \geq 0\}$

## Low-Level Description

Input:  $w = 0000$

$q_0$  0 0 0 0  $\sqcup$ ... means  
 $\sqcup$   $q_1$  0 0 0  $\sqcup$ ...  
 $\sqcup$  x  $q_3$  0 0  $\sqcup$ ...  
 $\sqcup$  x 0  $q_4$  0  $\sqcup$ ...  
 $\sqcup$  x 0 x  $q_3$   $\sqcup$ ...  
 $\sqcup$  x 0  $q_2$  x  $\sqcup$ ...  
 $\sqcup$  x  $q_2$  0 x  $\sqcup$ ...  
 $\sqcup$   $q_2$  x 0 x  $\sqcup$ ...  
 $q_2$   $\sqcup$  x 0 x  $\sqcup$ ...  
 $\sqcup$   $q_1$  x 0 x  $\sqcup$ ...  
 $\sqcup$  x  $q_1$  0 x  $\sqcup$ ...  
 $\sqcup$  x x  $q_3$  x  $\sqcup$ ...  
 $\sqcup$  x x x  $q_3$   $\sqcup$ ...



$\sqcup$  x x  $q_2$  x  $\sqcup$  ...  
 $\sqcup$  x  $q_2$  x x  $\sqcup$  ...  
 $\sqcup$   $q_2$  x x x  $\sqcup$  ...  
 $q_2$   $\sqcup$  x x x  $\sqcup$  ...  
 $\sqcup$   $q_1$  x x x  $\sqcup$  ...

$\sqcup$  x  $q_1$  x x  $\sqcup$  ...  
 $\sqcup$  x x  $q_1$  x  $\sqcup$  ...  
 $\sqcup$  x x x  $q_1$   $\sqcup$  ...  
 $\sqcup$  x x x  $\sqcup$   $q_{accept}$   $\sqcup$  ...

ACCEPT!

# Differences between TMs and Finite Automata

TMs can write (not just read)

TMs can move left (L) or right (R)

TMs can immediately halt if accept or reject

TMs can enter infinite loops.

TMs can have a special "tape" alphabet.

# Formal Definition of a TM

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  $\Sigma$ )
- $\delta$  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

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

$L$  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:

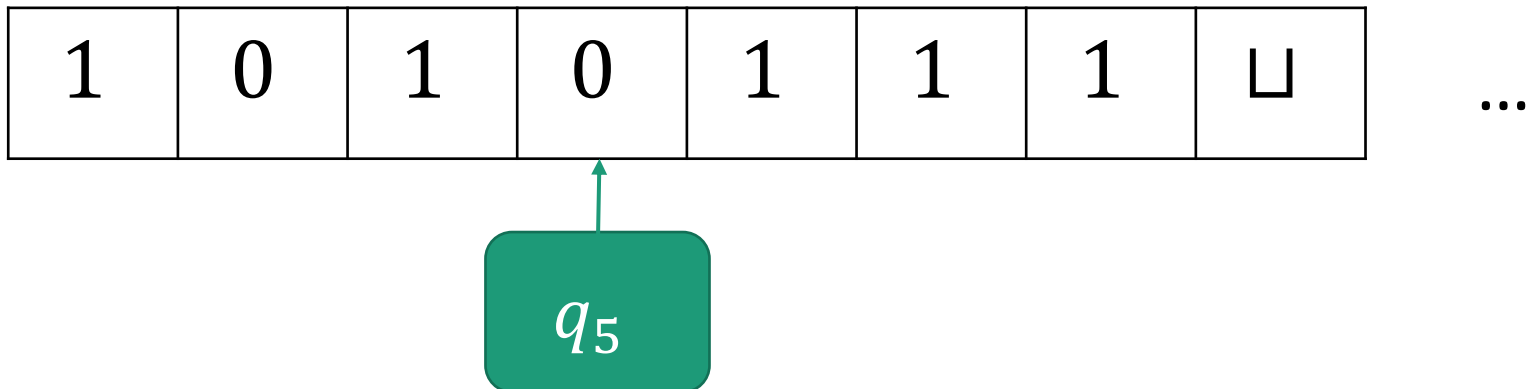
- Replace  $a$  with  $b$  in current cell
- Transition from state  $p$  to state  $q$
- Move tape head left UNLESS we are at left end of tape, in which case don't move

# Configuration of a TM: Formally

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

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

Example:  $101q_50111$



# How a TM Computes



**Start** configuration:  $q_0w$

In one step of computation:

- If  $\delta(q, b) = (q', c, R)$ , then  $uaqbv$  yields  $uacq'v$
- If  $\delta(q, b) = (q', c, L)$ , then  $uaqbv$  yields  $uq'acv$
- If we are at the left end of the tape in configuration  $qbv$ , what configuration do we reach if  $\delta(q, b) = (q', c, L)$ ?
  - a)  $cq'v$
  - b)  $q'cv$
  - c)  $q' \sqcup cv$
  - d)  $q'cbv$

# How a TM Computes

**Start** configuration:  $q_0w$

In one step of computation:

- If  $\delta(q, b) = (q', c, R)$ , then  $uaqbv$  yields  $uacq'v$
- If  $\delta(q, b) = (q', c, L)$ , then  $uaqbv$  yields  $uq'acv$
- If  $\delta(q, b) = (q', c, L)$ , then  $qbv$  yields  $q'cv$

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

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