BU CS 332 – Theory of Computation

https://forms.gle/HqPUHK2XkP9LjMh97

Lecture 9:

- Turing Machines

Reading:

Sipser Ch 3.1, 3.3

Mark Bun

October 5, 2021

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 $\{a^n b^n \mid n \geq 0\}$
- Can't recognize palindromes $\{w \mid w = w^R\}$

Somewhat more powerful (not in this course):

Pushdown Automata / Context-Free Grammars

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

Can recognize / generate

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.


LET'S SEE WHO THIS SUPERCOMPUTER REALLY IS..
Turing Machine
imgflip.com

h/t Islam

A Brief History

1900 – Hilbert's Tenth Problem

$p(x,y,z) = x^2 + yz^2 + 3z - 3$

$\exists ?\ x,y,z \in \mathbb{Z}^3$ s.t. $p(x,y,z) = 0$

$(x,y,z) = (0,0,1)$ is a solution

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.

an 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?

Input: mathematical statement

Output: Is this statement true or false?

David Hilbert 1862-1943

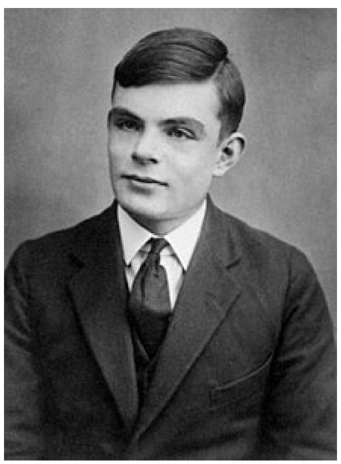
1936 – Solution to the *Entscheidungsproblem*

Alonzo Church 1903-1995

"An unsolvable problem of elementary number theory“

Model of computation: λ-calculus (CS 320)

"regular expression"



Alan Turing 1912-1954

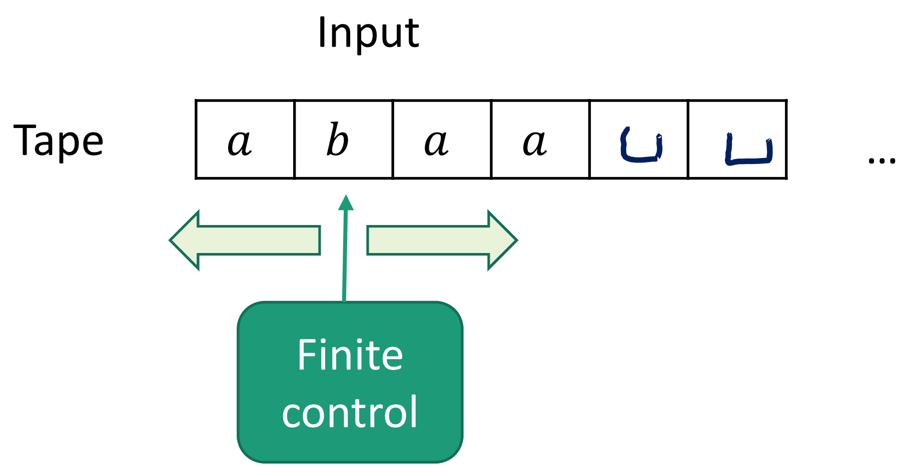
“On computable numbers, with an application to the *Entscheidungsproblem*”

Model of computation: Turing Machine

"finite automata"

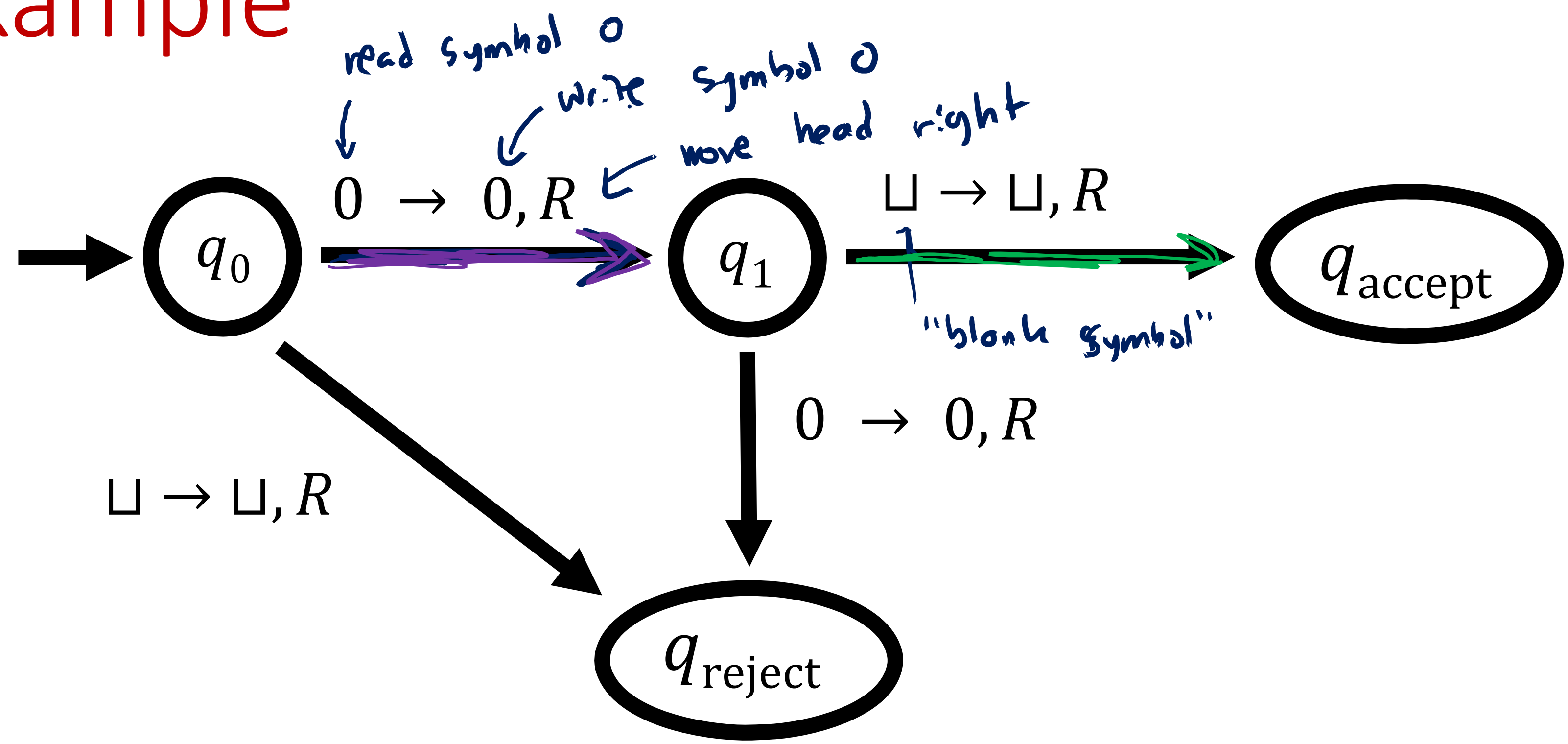
Turing Machines

The Basic Turing Machine (TM)

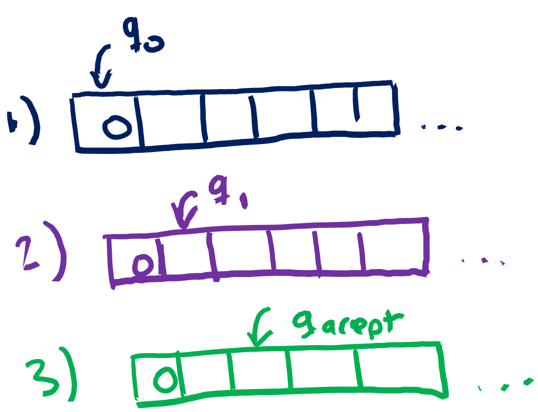


- Input is written on an infinitely long tape
- Head can both read and write, and move in both directions
- Computation halts as soon as control reaches "accept" or "reject" state

Example



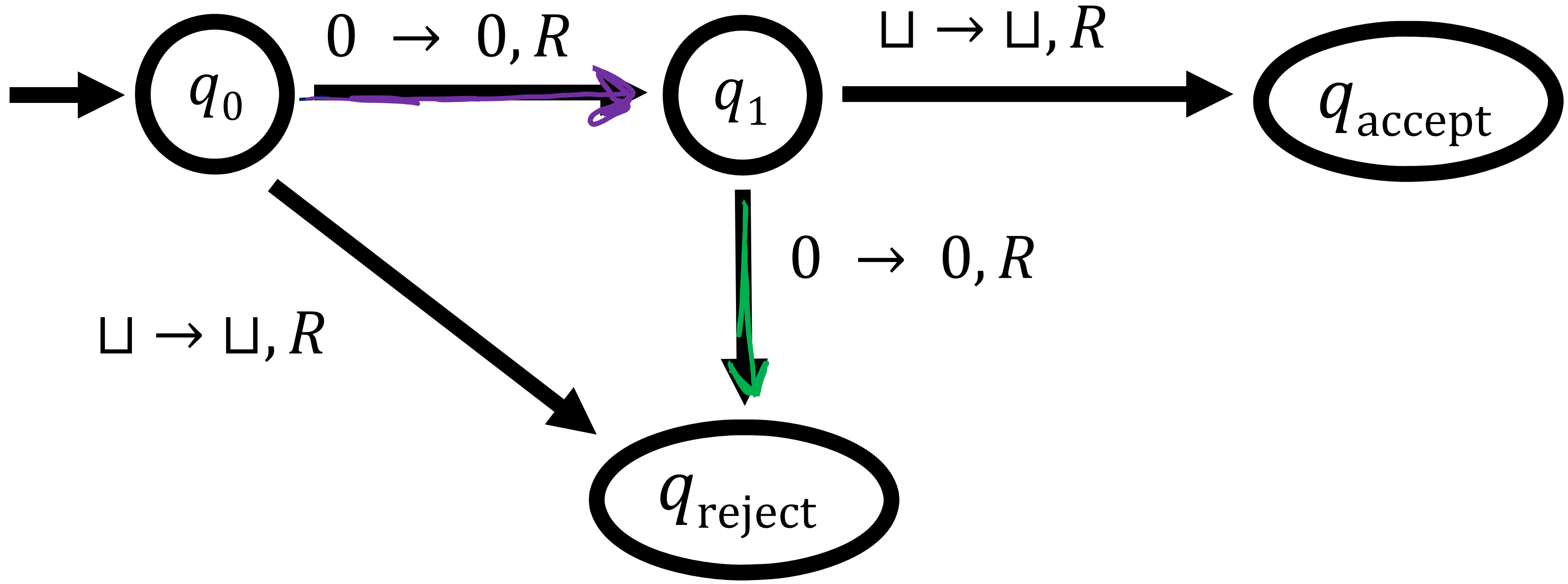
Ex: Input 0



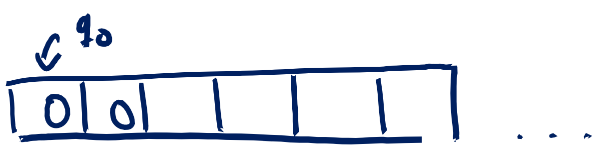
TM accepts input 0

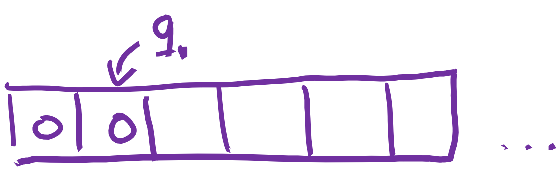
Example

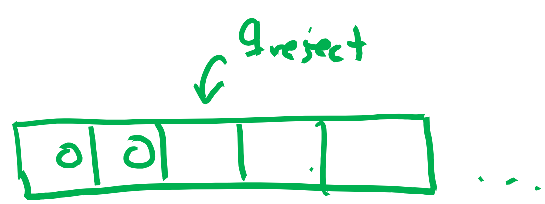
Language recognized by TM is $\{0\}$



Input: 00

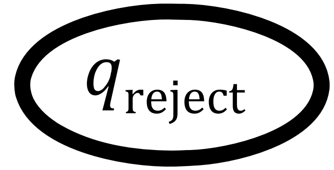
1) 

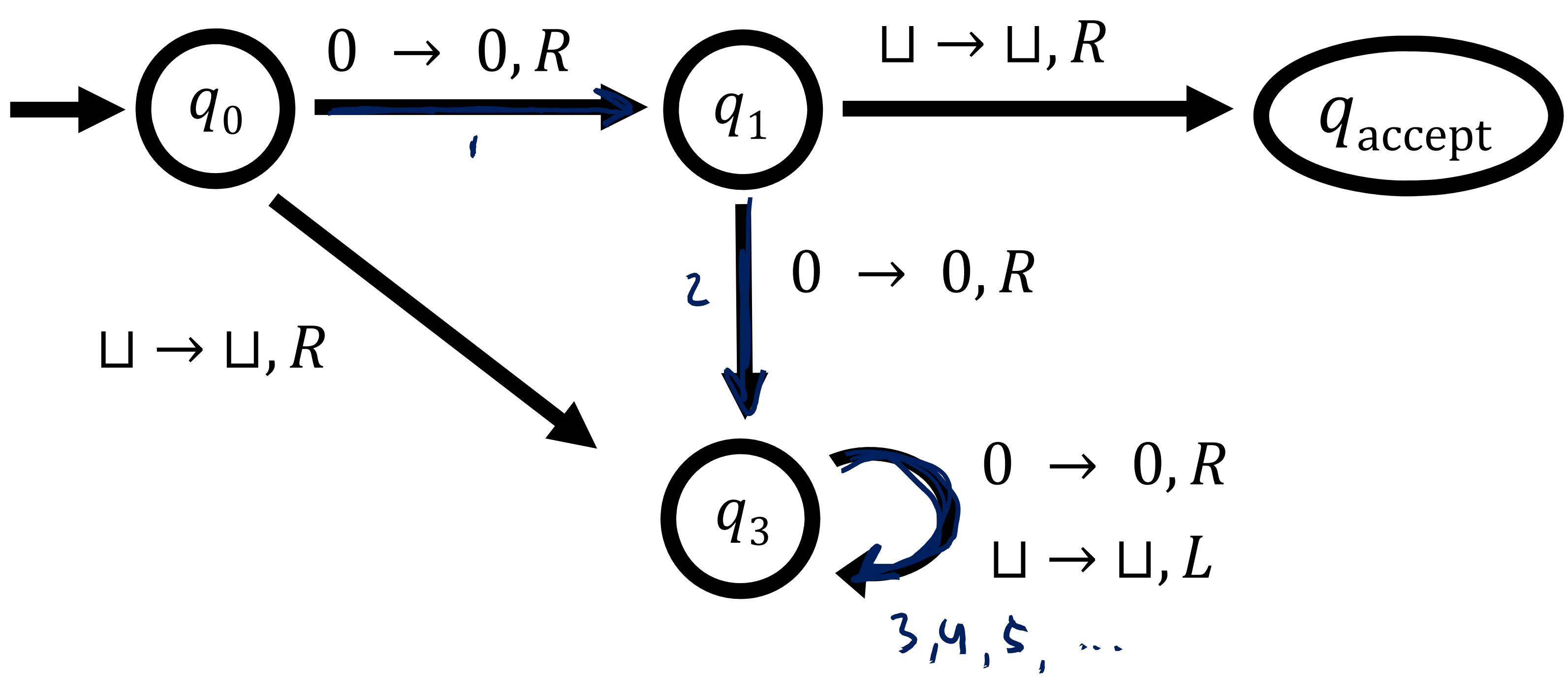
2) 

3) 

TM rejects 00

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

Program Analogy

High-Level Description

An algorithm (like CS 330)

Python, Java

Implementation-Level Description

Describe (in English) the instructions for a TM

C, assembly

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

Low-Level Description

State diagram or formal specification

Machine or byte

Example

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

$A = \{0^{2^n} \mid n \geq 0\}$ Not a regular language

Ex: $0\cancel{0}\cancel{0}\cancel{0}\cancel{0}\cancel{0}\cancel{0}\cancel{0}$

High-Level Description

Ex: $0\cancel{0}0\,\cancel{0}0\cancel{0}$

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

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

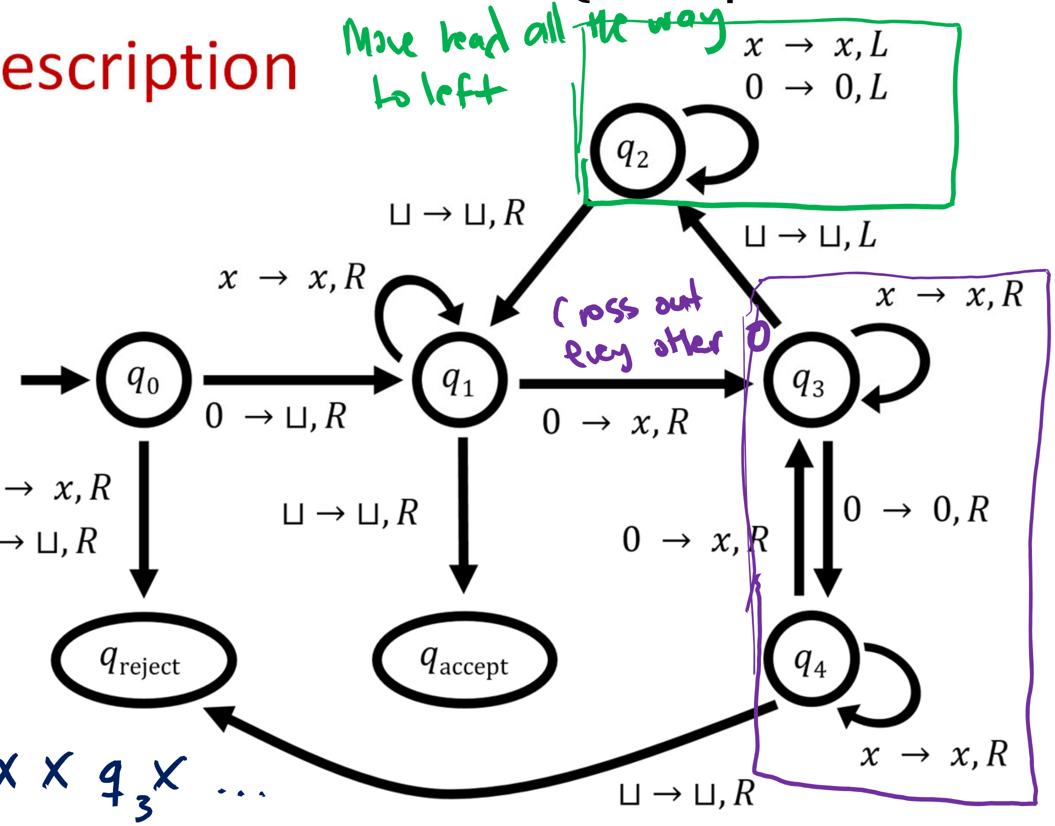
Implementation-Level Description

1. While moving the tape head left-to-right: Head movements
   a) Cross off every other 0 What TM is reading / writing
   b) If there is exactly one 0 when we reach the right end of the tape, accept
   c) If there is an odd (> 1) number of 0s when we reach the right end of the tape, 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



Ex: Input 0000

q_0 0000 ⊔⊔ ...

⊔ q_1 000 ...

⊔ x q_3 00 ...

⊔ x 0 q_4 0 ...

⊔ x 0 x q_3 ⊔ ...

⊔ x 0 q_2 x ...

⊔ x q_2 0 x ...

⊔ q_2 x 0 x ...

q_2 ⊔ x 0 x ...

⊔ q_1 x 0 x ...

⊔ x q_1 0 x ...

⊔ x x q_3 x ...

⊔ x x x q_3 ⊔ ...

⊔ x x q_2 x ⊔ ...

⊔ x q_2 x x ...

⊔ q_2 x x x ...

q_2 ⊔ x x x ...

⊔ q_1 x x x ...

⊔ x q_1 x x ...

⊔ x x q_1 x ...

⊔ x x x q_1 ...

⊔ x x x ⊔ q_{accept} ⊔ ...

Accept!

TMs vs. Finite Automata

TM can move head in both directions (FA in one way)

TMs can write

TM explicitly enters accept or reject state, and computation then stops

TMs can do more! (solve problems requiring unbounded memory)

Specifically, can recognize non-regular languages

TMs can read or write symbols not in input alphabet

TMs can loop forever

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
- Σ is the input alphabet (does **not** include $\sqcup$)
- Γ is the tape alphabet (contains $\sqcup$ and Σ)
- δ is the transition function

$\Sigma \subseteq \Gamma$
$\sqcup \in \Gamma$
and Γ could have more in it, like X

...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

new symbol to write under head

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

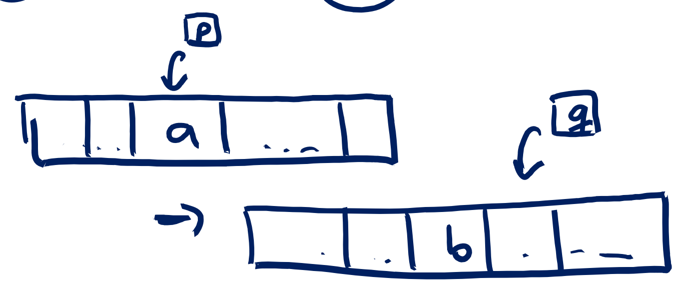
current state — symbol under head — next state — move head left or right

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

$$p \xrightarrow{a \to b,\ R} q$$



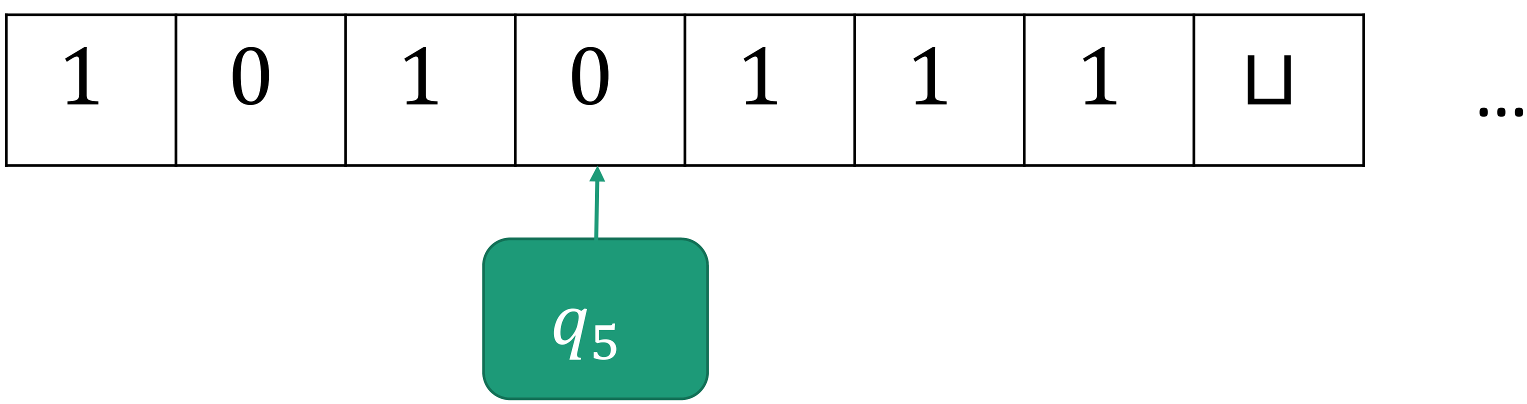
$\delta(p, a) = (q, b, L)$ means:

- 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 that captures the **state** of a TM together with the **contents of the tape**

1 0 1 q_5 0 1 1 1

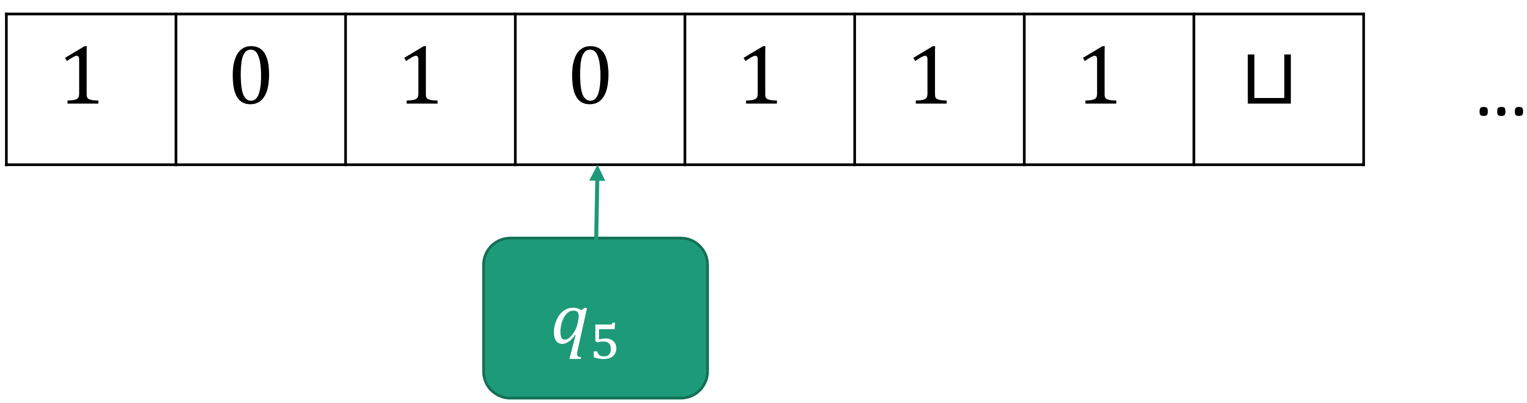


Configuration of a TM: Formally

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

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

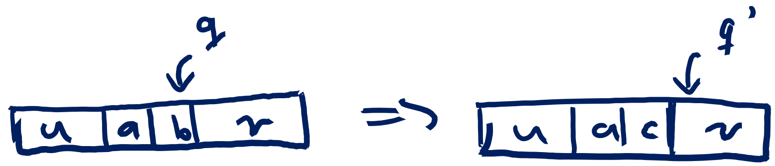
Example: $\overbrace{101}^{u}\overset{q}{q_5}\overbrace{0111}^{v}$



How a TM Computes

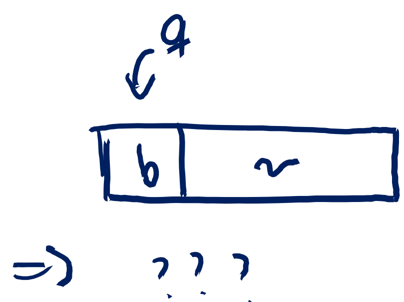
Start configuration: $q_0 w$

One step of computation:



- If $\delta(q, b) = (q', c, R)$, then $ua\ q\ bv$ yields $uac\ q'\ v$
- If $\delta(q, b) = (q', c, L)$, then $ua\ q\ bv$ yields $u\ q'\ acv$
- If we are at the left end of the tape in configuration $q\ bv$, 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: $q_0 w$

One step of computation:

- If $\delta(q, b) = (q', c, R)$, then $ua\ q\ bv$ yields $uac\ q'\ v$
- If $\delta(q, b) = (q', c, L)$, then $ua\ q\ bv$ yields $u\ q'\ acv$
- If $\delta(q, b) = (q', c, L)$, then $q\ bv$ yields $q'\ cv$

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

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

How a TM Computes

M accepts input w if there exists a sequence of configurations $C_1, \ldots, C_k$ such that:

- $C_1 = q_0 w$
- C_i yields C_{i+1} for every i
- 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_{accept}
- $w \notin A \implies$ M halts on w in state q_{reject} OR
  M runs forever

Recognizers vs. Deciders

$L(M)$ $=$ 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_{accept}
- $w \notin A \implies M$ halts on w in state q_{reject} OR
  M runs forever

A is (Turing-)decidable if $A = L(M)$ for some TM M
which halts on every input

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

Back to Hilbert's Tenth Problem

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

$L =$

- L is Turing-recognizable

- L is not decidable (1949-70)