BU CS 332 – Theory of Computation

https://forms.gle/44vcjAzahbobkuAQ8

Lecture 10:

- Turing Machines
- TM Variants and Closure Properties

Reading:

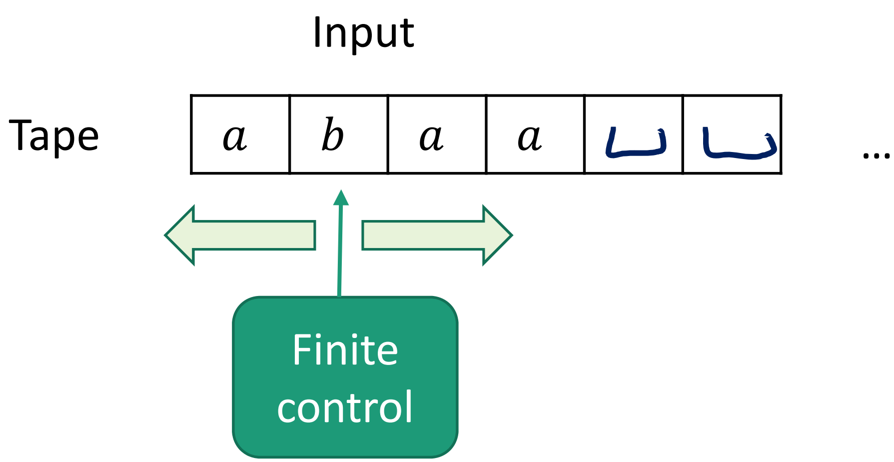
Sipser Ch 3.1-3.3

HW 4 deadline
pushed to
Friday 11:59PM

Mark Bun

October 13, 2022

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

Three Levels of Abstraction

High-Level Description

An algorithm (like CS 330)

Implementation-Level Description

Describe (in English) the instructions for a TM

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

Low-Level Description

State diagram or formal specification

Example

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

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

High-Level Description

$\cancel{0}\ \cancel{0}\ \cancel{0}\ \cancel{0}\ \cancel{0}\ \cancel{0}\ \cancel{0}\ 0$

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

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

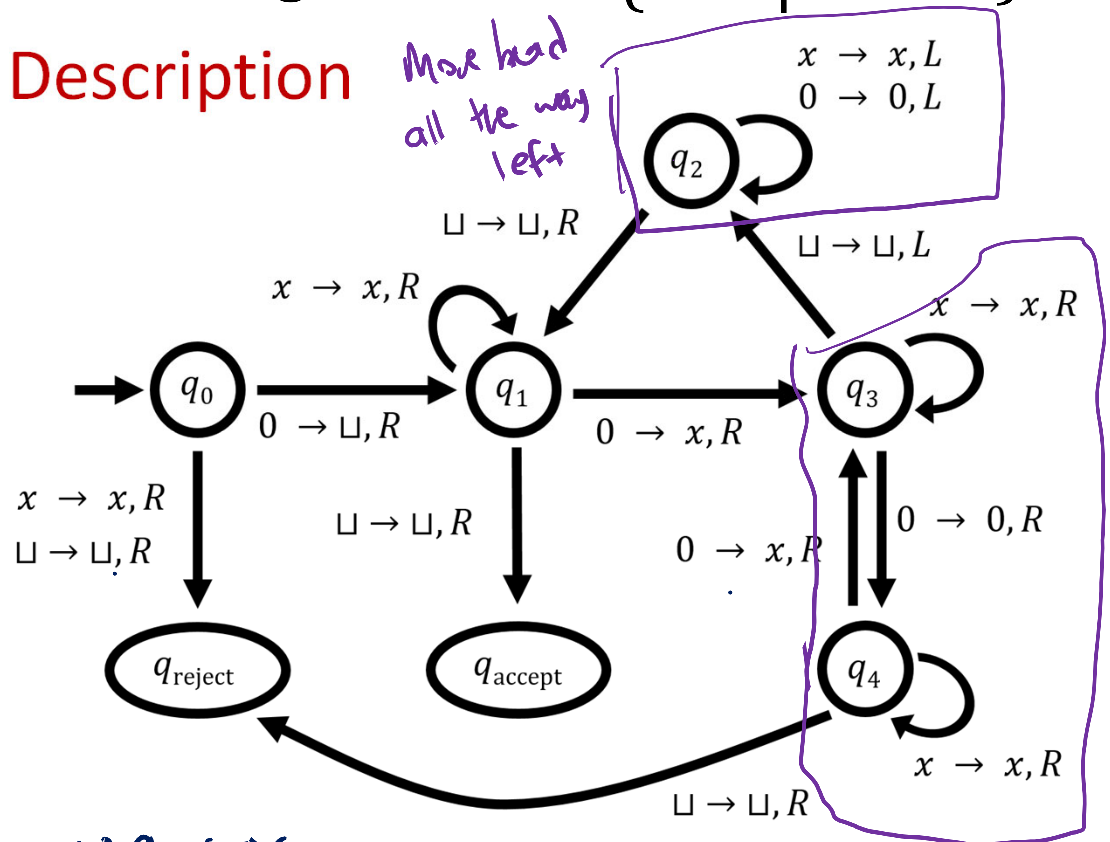
Implementation-Level Description

1. While moving the tape head left-to-right:
   a) Cross off every other 0 (replace every other 0 w/ X)
   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



Cross out every other 0

Ex: Input 0000

q_0: | 0 | 0 | 0 | 0 | ⊔ | ⊔ | ⊔ | ...

$q_0\,0000$
$\sqcup\, q_1\, 000$
$\sqcup\, x\, q_3\, 00$
$\sqcup\, x\, 0\, q_4\, 0$
$\sqcup\, x\, 0\, x\, q_3$
$\sqcup\, x\, 0\, q_2\, x\, \sqcup$
$\sqcup\, x\, q_2\, 0\, x$
$\sqcup\, q_2\, x\, 0\, x$
$q_2\, \sqcup\, x\, 0\, x$

$\sqcup\, q_1\, x\, 0\, x$
$\sqcup\, x\, q_1\, 0\, x$
$\sqcup\, x\, x\, q_3\, x$
$\sqcup\, x\, x\, x\, q_3$
$\sqcup\, x\, x\, q_2\, x$
$\sqcup\, x\, q_2\, x\, x$
$\sqcup\, q_2\, x\, x\, x$
$q_2\, \sqcup\, x\, x\, x$

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

Accept!

Differences between TMs and Finite Automata

TMs can move their head in both directions

TMs can write

TMs have unlimited memory in the form of the tape

TMs make a decision when they explicitly reach q_{accept} or q_{reject} vs. FAs make a decision when they reach the end of the input

TMs can loop forever w/o halting

TMs have one accept (and one reject) state

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
  e.g. $\Sigma = \{0, 1\}$ $\Gamma = \{0, 1, \sqcup, x\}$
- Σ is the input alphabet (does **not** include $\sqcup$)
- Γ is the tape alphabet (contains $\sqcup$ and Σ) $(\Sigma \subseteq \Gamma)$
  e.g. $x \in \Gamma$
- δ 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

$p \xrightarrow{0 \to 1, L} q$

$\delta(p, 0) = (q, 1, L)$

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

current state — current symbol — next state — symbol to write — head movement

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

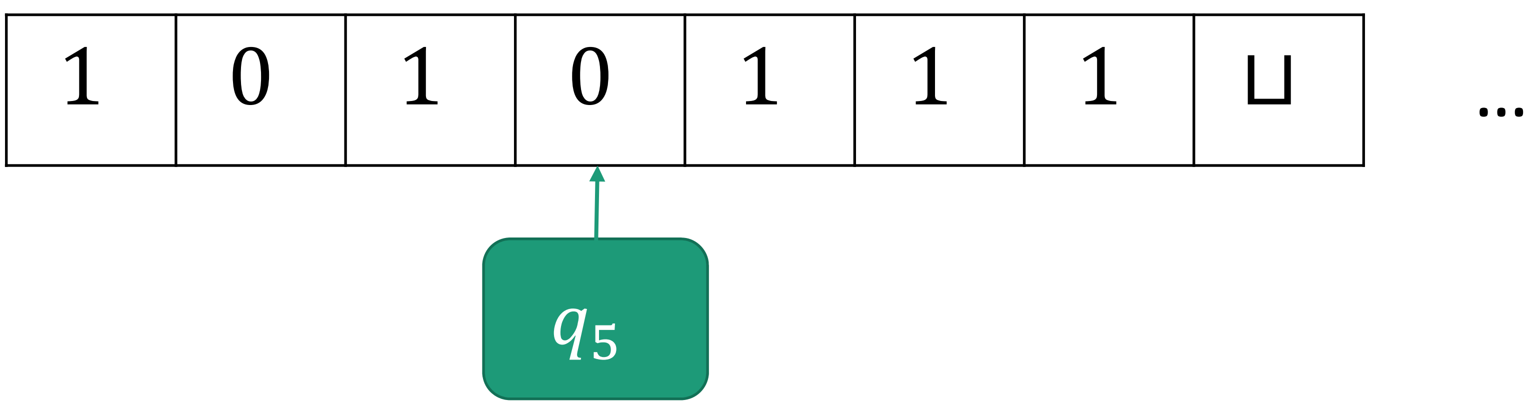
$\delta(p, a) = (q, b, L)$ means:

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

$1 0 1 q_5 0 1 1 1$

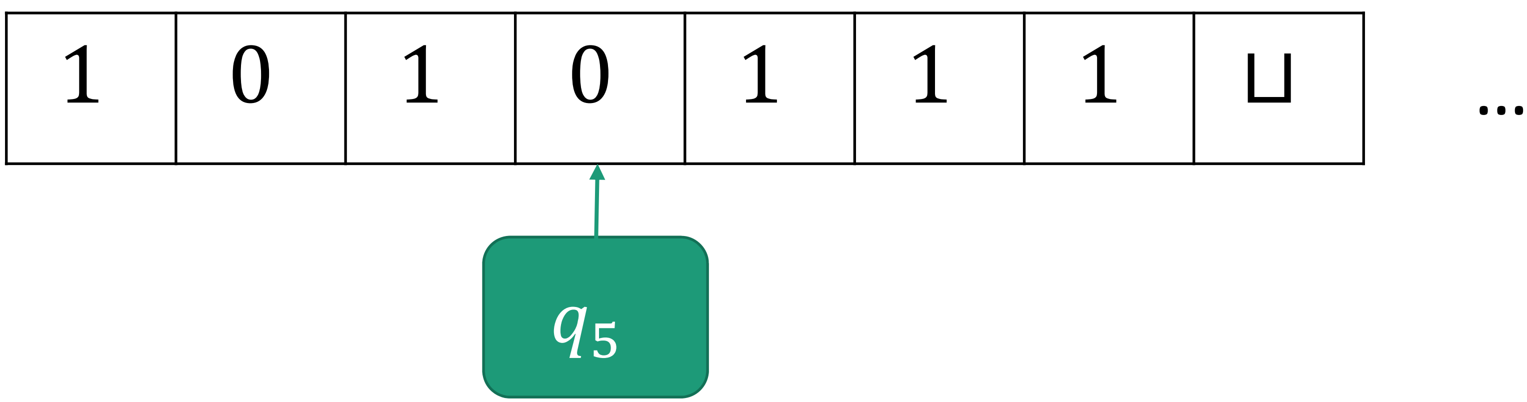


Configuration of a TM: Formally

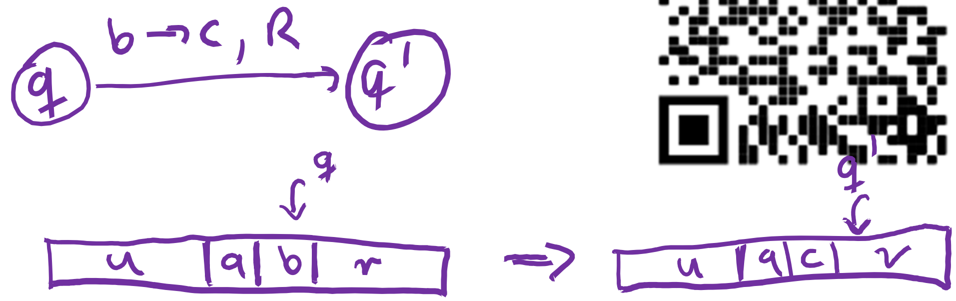
A configuration is a string uqv where $q \in Q$ and $u, v \in \Gamma^*$

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

Example: $101q_5 0111$



How a TM Computes

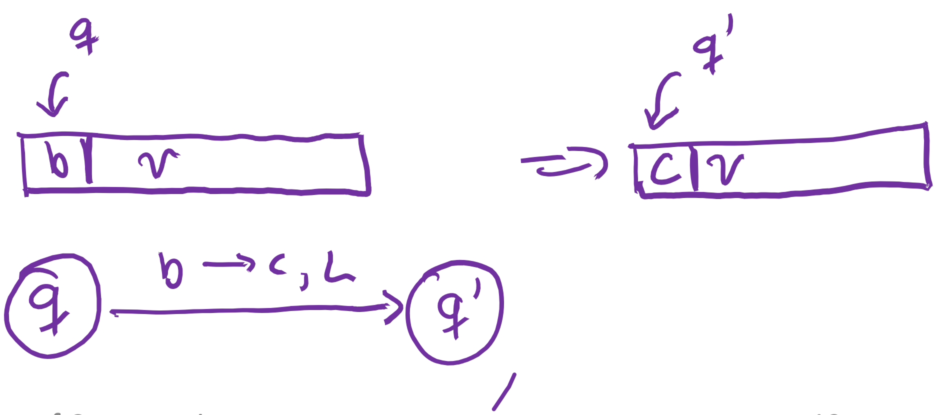


Start configuration: $q_0 w$

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_{accept}

Rejecting configuration: q = q_{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 [TM goes from config. C_i to C_{i+1} in one transition]
- C_k is an accepting configuration

L(M) = the language recognized by M

$L(M)$ $=$ 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 on w

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 on w

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}

Recognizers vs. Deciders

Which of the following is true about the relationship between $\underbrace{\text{decidable}}_{\text{languages recognized by TM deciders}}$ and $\underbrace{\text{recognizable languages}}_{\text{languages recognized by arbitrary TMs}}$?

a) The decidable languages are a subset of the recognizable languages Decidable langs $\subseteq$ Recognizable langs

b) The recognizable languages are a subset of the decidable languages

c) They are incomparable: There might be decidable languages which are not recognizable and vice versa

Example: Arithmetic on a TM

The following TM decides $\text{MULT} = \{a^i b^j c^k \mid i \times j = k\}$:

On input string w: High-level

1. Check w is formatted correctly i.e. $w \in L(a^* b^* c^*)$
2. For each a appearing in w:
3. For each b appearing in w: a abbb
4. Attempt to cross off a c. If none exist, reject.
5. If all c's are crossed off, accept. Else, reject.

Example: Arithmetic on a TM

The following TM decides $\mathrm{MULT} = \{a^i b^j c^k \mid i \times j = k\}$:

On input string w: Implementation-level

1. Scan the input from left to right to determine whether it is a member of $L(a^* b^* c^*)$ Can be done by a DFA ⇒ can be done in one read-only pass by a TM
2. Return head to left end of tape
3. Cross off an a if one exists. Scan right until a b occurs. Shuttle between b's and c's crossing off one of each until all b's are gone. Reject if all c's are gone but some b's remain.

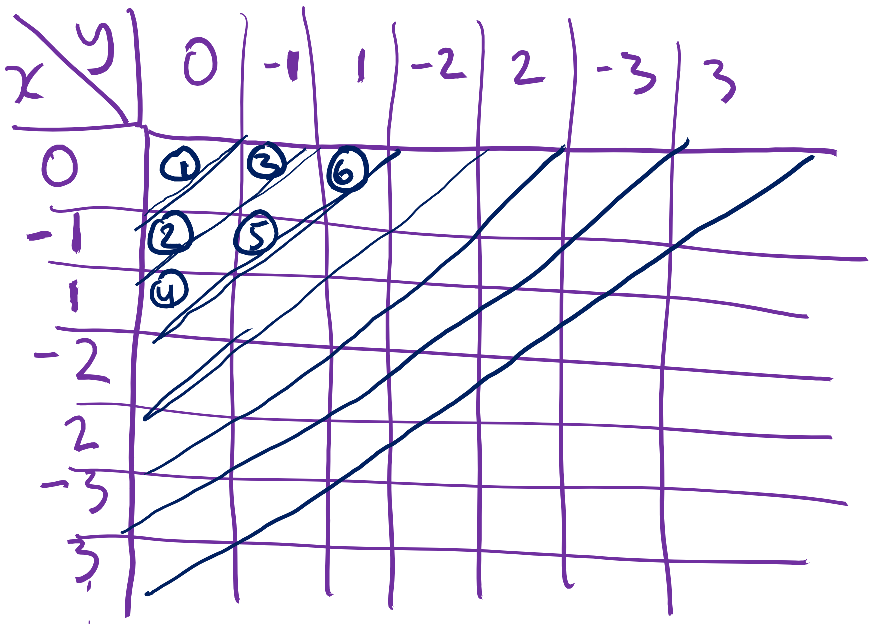
   ~~a~~ a ~~bbb~~ ~~ccc~~ ccc → ~~a~~ ~~a~~ ~~bbb~~ ~~cccccc~~
4. Restore crossed off b's. If any a's remain, repeat step 3.
5. If all c's are crossed off, accept. Else, reject.

Back to Hilbert's Tenth Problem

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

$L = \{ p(x_1, \ldots, x_n) \mid n \geq 0,\ \exists (x_1, \ldots, x_n) \in \mathbb{Z}^n \text{ s.t. } p(x_1, \ldots, x_n) = 0 \}$

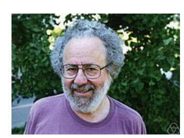
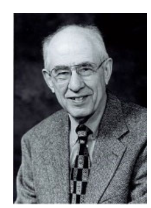
- L is Turing-recognizable Special case: $L_2 = \{ p(x, y) \mid \ldots\ldots \}$

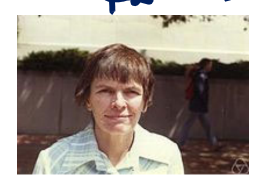


Alg: For each cell (x, y) in grid, test if $p(x,y) = 0$; if so, accept

Analysis: If $p \in L_2$ then $\exists (x,y) \in$ grid s.t. $p(x,y) = 0 \Rightarrow$ alg. accepts

If $p \notin L_2$, alg. loops forever



- L is not decidable (1949-70)