CS304: Automata and Formal Languages

Lec 22

Turing Machines: Continued

Rachit Nimavat

October 10, 2025

Outline

1. Recap
2. TM: More Examples
3. Scope of this Course

Turing Machine: Informal Defn

Imagine a machine with a **read/write head** that can move along an infinitely long piece of **tape**. This tape is divided into **cells**, each capable of holding a single symbol.
This simple model has three fundamental abilities:

Read the symbol on the tape cell under the head.

Write a new symbol to that cell, erasing what was there.

Move the head one cell to the left or one cell to the right. **(Two-Way Head)**

That's it! The machine's behavior is directed by a finite set of states, just like an Finite Automaton.

First Example

Consider the language $L = \{a^n b^n c^n \mid n \geq 0\}$.

A string w is given on the tape, and the head starts at the leftmost symbol of w.

Infinitely many blank symbols (denoted by $\square$) follow w.

Turing Machine naturally mimics our intuition for algorithms:

1. Scan right until first $\square$ to check whether w is of the form $a^*b^*c^*$. If not, reject.
2. Return head to the leftmost symbol.
3. Scan right, crossing off single a, b, and c (replace them with $\times$) in each pass.
4. If all symbols are crossed off, accept.
5. If a symbol is missing, reject.
6. Go to step 2.

What is a Turing machine (TM)?

Definition

A Turing machine (TM) M is a 6-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, H)$, where,

1. Q: A finite set (set of states).
2. Σ: A finite set (input alphabet). Σ excludes $\triangleright, \square, \leftarrow, \rightarrow$.
3. Γ: A finite set (tape alphabet). $\Sigma \cup \{\triangleright, \square\} \subseteq \Gamma$. Γ excludes $\leftarrow, \rightarrow$.
4. $\delta : (Q - H) \times \Gamma$ to $Q \times (\Gamma \cup \{\leftarrow, \rightarrow\})$ is the transition function such that the tape head never falls off or erases $\triangleright$ symbol
   $\triangleright$ Time (computation)
5. q_0: The start state (belongs to Q).
6. $H = \{q_{\text{acc}}, q_{\text{rej}}\}$: The set of halting states (subset of Q).

Some notes on the Turing machine

Symbols

- $\triangleright$: Left end symbol
- $\square$: Blank symbol
- $\leftarrow, \rightarrow$: Left and right movement symbols
- Σ : Represents input/output/special symbols
- Γ : Represents symbols that can be present on the tape

Transition

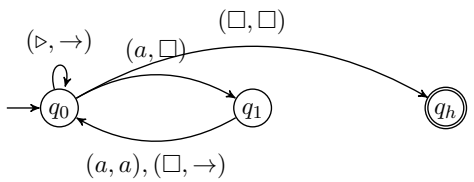
- M never falls off the left end of the tape i.e.,
  when the current symbol is $\triangleright$, the tape head has to move right
- M stops when it reaches an accept or a reject state i.e.,
  δ is not defined for states in H

Construct TM to erase the input string

Problem

- Construct a TM to erase the input string

Solution (continued)



	Current symbol (Γ)		
Current state ($Q - H$)	$\triangleright$	a	$\square$
q_0	$(q_0, \rightarrow)$	$(q_1, \square)$	$(q_h, \square)$
q_1	–	(q_0, a)	$(q_0, \rightarrow)$

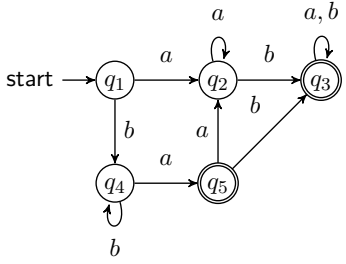
Construct TM for a regular language

Problem

- Construct a DFA that accepts all strings from the language $L = \{\text{strings containing } ab \text{ or end with } ba\}$

Solution

- Expression: $((a|b)^*ab(a|b)^*) \mid ((a|b)^*ba)$
- DFA:

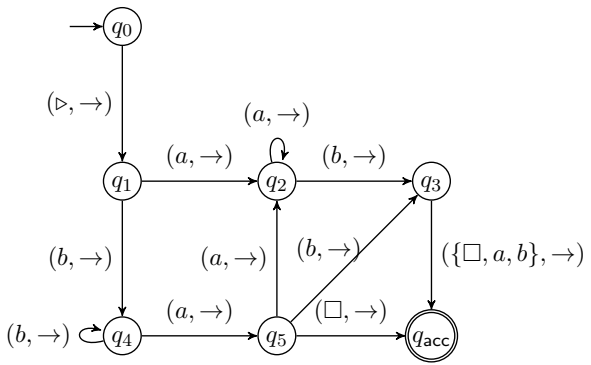


Construct TM for a regular language

Problem

- Construct a Turing machine that accepts all strings from the language $L = \{\text{strings containing } ab \text{ or end with } ba\}$

Solution



Construct TM for a regular language

Problem

- Construct a Turing machine that accepts all strings from the language $L = \{\text{strings containing } ab \text{ or end with } ba\}$

Solution (continued)

	Current symbol (Γ)			
Current state ($Q - H$)	$\triangleright$	a	b	$\square$
q_0	$(q_1, \rightarrow)$	–	–	–
q_1	–	$(q_2, \rightarrow)$	$(q_4, \rightarrow)$	–
q_2	–	$(q_2, \rightarrow)$	$(q_3, \rightarrow)$	–
q_3	–	$(q_{\text{acc}}, \rightarrow)$	$(q_{\text{acc}}, \rightarrow)$	$(q_{\text{acc}}, \rightarrow)$
q_4	–	$(q_5, \rightarrow)$	$(q_4, \rightarrow)$	–
q_5	–	$(q_2, \rightarrow)$	$(q_3, \rightarrow)$	$(q_{\text{acc}}, \rightarrow)$

Construct TM for a regular language

Problem

- Construct a Turing machine that accepts all strings from the language $L = \{\text{strings containing } ab \text{ or end with } ba\}$

Solution (continued)

- TM accepts the string bba because it enters the q_{acc} state

Time	State	Tape						
0	q_0	$\triangleright$	b	b	a	$\square$	$\square$	$\cdots$
1	q_1	$\triangleright$	b	b	a	$\square$	$\square$	$\cdots$
2	q_4	$\triangleright$	b	b	a	$\square$	$\square$	$\cdots$
3	q_4	$\triangleright$	b	b	a	$\square$	$\square$	$\cdots$
4	q_5	$\triangleright$	b	b	a	$\square$	$\square$	$\cdots$
5	q_{acc}	$\triangleright$	b	b	a	$\square$	$\square$	$\cdots$

Construct TM for a regular language

Problem

- Construct a Turing machine that accepts all strings from the language $L = \{\text{strings containing } ab \text{ or end with } ba\}$

Solution (continued)

- TM rejects the string bbb because it enters the q_{rej} state

Time	State	Tape						
0	q_0	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$
1	q_1	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$
2	q_4	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$
3	q_4	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$
4	q_4	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$
5	q_{rej}	$\triangleright$	b	b	b	$\square$	$\square$	$\cdots$

Construct TM for a regular language

Problem

- Construct a Turing machine that accepts all strings from the language $L = \{\text{strings containing } ab \text{ or end with } ba\}$

Solution (continued)

- TM accepts the string $aabbbbb$ because it enters the q_{acc} state
- Unlike DFA and CFG, a TM can accept a string without scanning the string completely

Time	State	Tape									
0	q_0	$\triangleright$	a	a	b	b	b	b	b	$\square$	$\cdots$
1	q_1	$\triangleright$	a	a	b	b	b	b	b	$\square$	$\cdots$
2	q_2	$\triangleright$	a	a	b	b	b	b	b	$\square$	$\cdots$
3	q_2	$\triangleright$	b	b	b	b	b	b	b	$\square$	$\cdots$
4	q_3	$\triangleright$	b	b	a	b	b	b	b	$\square$	$\cdots$
5	q_{acc}	$\triangleright$	b	b	b	b	b	b	b	$\square$	$\cdots$

Construct TM for a regular language

More problems

Use the TM to check acceptance of the following strings:

- ϵ
- aba ▷ contains ab and ends with ba
- aaa
- aab
- baa

Construct TM for copying strings

Problem

- Construct a Turing machine that copies a string from the language $L = \Sigma^*$ where $\Sigma = \{a, b\}$.

Construct TM for copying strings

Problem

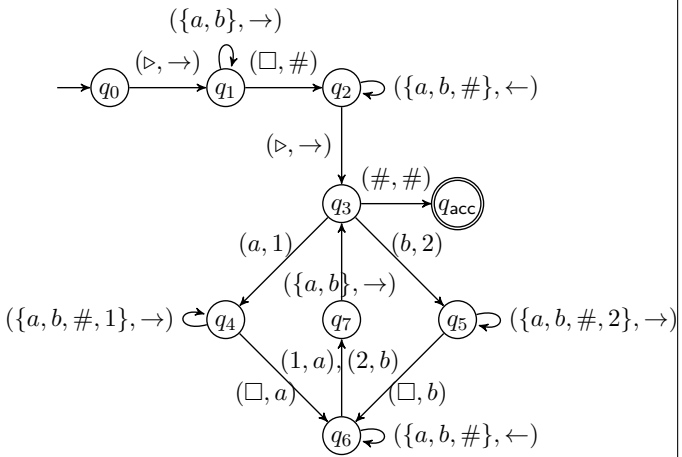
- Construct a Turing machine that copies a string from the language $L = \Sigma^*$ where $\Sigma = \{a, b\}$.

Solution

- Language recognizers such as DFA's cannot perform computational tasks such as copying strings.
  So, no DFA can be used for copying strings.
- Language generators such as CFG's cannot perform computational tasks such as copying strings.
  So, no CFG can be used for copying strings.
- TM's are more powerful than language recognizers and language generators.
  A TM can be used for copying strings.

Construct TM for copying strings

Solution (continued)



Construct TM for copying strings

Solution (continued)

State	Tape									
q_0	$\triangleright$	b	b	a	$\square$	$\square$	$\square$	$\square$	$\square$	$\cdots$
q_2	$\triangleright$	b	b	a	$\#$	$\square$	$\square$	$\square$	$\square$	$\cdots$
q_2	$\triangleright$	b	b	a	$\#$	$\square$	$\square$	$\square$	$\square$	$\cdots$
q_5	$\triangleright$	2	b	a	$\#$	$\square$	$\square$	$\square$	$\square$	$\cdots$
q_6	$\triangleright$	2	b	a	$\#$	b	$\square$	$\square$	$\square$	$\cdots$
q_7	$\triangleright$	b	b	a	$\#$	b	$\square$	$\square$	$\square$	$\cdots$
q_5	$\triangleright$	b	2	a	$\#$	b	$\square$	$\square$	$\square$	$\cdots$
q_6	$\triangleright$	b	2	a	$\#$	b	b	$\square$	$\square$	$\cdots$
q_7	$\triangleright$	b	b	a	$\#$	b	b	$\square$	$\square$	$\cdots$
q_4	$\triangleright$	b	b	1	$\#$	b	b	$\square$	$\square$	$\cdots$
q_6	$\triangleright$	b	b	1	$\#$	b	b	a	$\square$	$\cdots$
q_7	$\triangleright$	b	b	a	$\#$	b	b	a	$\square$	$\cdots$
q_3	$\triangleright$	b	b	a	$\#$	b	b	a	$\square$	$\cdots$
q_{acc}	$\triangleright$	b	b	a	$\#$	b	b	a	$\square$	$\cdots$

Construct TM for copying strings

Problem

- Construct a Turing machine that copies a string from the language $L = \Sigma^*$ where $\Sigma = \{a, b\}$.

Solution (continued)

- $\Gamma = \Sigma \cup \{\triangleright, \square, \#, 1, 2\}$
- Cells with "$-$" means that the TM terminates in q_{rej} state

	Current symbol (Γ)						
State	$\triangleright$	a	b	$\#$	1	2	$\square$
q_0	$(q_1, \rightarrow)$	$-$	$-$	$-$	$-$	$-$	$-$
q_1	$-$	$(q_1, \rightarrow)$	$(q_1, \rightarrow)$	$-$	$-$	$-$	$(q_2, \#)$
q_2	$(q_3, \rightarrow)$	$(q_2, \leftarrow)$	$(q_2, \leftarrow)$	$(q_2, \leftarrow)$	$-$	$-$	$-$
q_3	$-$	$(q_4, 1)$	$(q_5, 2)$	$(q_{\text{acc}}, \#)$	$-$	$-$	$-$
q_4	$-$	$(q_4, \rightarrow)$	$(q_4, \rightarrow)$	$(q_4, \rightarrow)$	$(q_4, \rightarrow)$	$-$	(q_6, a)
q_5	$-$	$(q_5, \rightarrow)$	$(q_5, \rightarrow)$	$(q_5, \rightarrow)$	$-$	$(q_5, \rightarrow)$	(q_6, b)
q_6	$-$	$(q_6, \leftarrow)$	$(q_6, \leftarrow)$	$(q_6, \leftarrow)$	(q_7, a)	(q_7, b)	$-$
q_7	$-$	$(q_3, \rightarrow)$	$(q_3, \rightarrow)$	$-$	$-$	$-$	$-$

Construct TM for copying strings

More problems

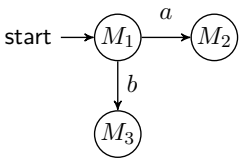
Use the TM to copy the following strings:

- ϵ
- a
- b
- aab

How to construct complicated TM's?

Constructing complicated TM's

- We can make complicated TM's from simpler TM's using the structure of a finite automaton
- Nodes of the automaton are the simpler TM's
- A connection $M_i \xrightarrow{k} M_j$ means when TM M_i halts and the current tape symbol is k, then TM M_j can start.



- Can you think of some examples of complicated TM's built from simpler TM's?

Why So Complicated?!

Turing Machine is a very low-level model of computation.

It is designed to be as simple as possible while still being able to perform any computation that can be done by a computer.

The complexity of the examples arises from the need to explicitly manage the tape and head movements, which are abstracted away in higher-level models like programming languages.

The goal is to understand the fundamental capabilities and limitations of computation, which requires working with this minimalistic model.

We will **not focus on designing Turing Machines for specific tasks**, but rather on **understanding what can and cannot be computed in principle**.

How are TM's different from DFA's and PDA's?

Feature	DFA	PDA	TM
Memory size	Finite	Infinite	Infinite
Halts?	✓	✓	✓, ✗
Input scanning	Left-to-right	Left-to-right	Arbitrary
#Passes	1 pass	1 pass	Any
Halting	End of input	End of input	Accept state
Computing power	Least	Medium	Highest
Language recognizer?	✓	✓	✓
Function calculator?	✗	✗	✓
Decide RL's?	✓	✓	✓
Decide CFL's?	✗	✓	✓
Decide REL's?	✗	✗	✓