COL 352 Introduction to Automata and Theory of Computation

Nikhil Balaji

Bharti 420
Indian Institute of Technology, Delhi
nbalaji@cse.iitd.ac.in

March 16, 2023

Lecture 21:Turing Machines

So far..

- Regular languages and Finite Automata
  - DFA = NFA = 2-DFA = 2-NFA = Regex
  - All of the above capture regular languages
  - Non-regular languages exist - Pumping Lemma, Myhill Nerode
  - Algorithms for manipulating/reasoning about DFAs exist! (and are sometimes efficient)
- Context-free languages and Pushdown Automata
  - Nondeterministic Pushdown Automata = Context-free Grammars
  - Deterministic PDAs = DCFL
  - DCFL $\neq$ CFL
  - There exist languages which are not CFLs
  - Pumping Lemma for CFLs

Other possible variants

- 2-NPDA vs NPDA?
- Do there exist grammars that capture them?

Other possible variants

- 2-NPDA vs NPDA?
- Do there exist grammars that capture them?
- There exist 2-NPDA that can accept $\{0^n 1^n 2^n \mid n \in \mathbb{N}\}$.
- What about machines with 2 stacks?
- What about all these machines with k pointers on the input tape?
- PDA/Grammars with weights for each transitions? (useful in NLP)

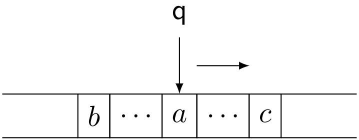
Other possible variants

- 2-NPDA vs NPDA?
- Do there exist grammars that capture them?
- There exist 2-NPDA that can accept $\{0^n 1^n 2^n \mid n \in \mathbb{N}\}$.
- What about machines with 2 stacks?
- What about all these machines with k pointers on the input tape?
- PDA/Grammars with weights for each transitions? (useful in NLP)

Is there a machine-independent notion of computation?

Turing machines

What is a Turing machine? (Informal description.)



- Read and write on the input tape. Head moves left/right.
- The tape is infinite.
- A special symbol & to indicate blank cells.
- Initially all cells blank except the part where the input is written.
- Special states for accepting and rejecting.

Formal definition

Definition

A Turing machine (TM) is given by $M = (Q, \Sigma, \Gamma, \delta, q_0, q_f, q_{rej})$

Q:	set of states	Σ:	input alphabet
q_0:	start state	Γ:	tape alphabet, $\Sigma \subseteq \Gamma$, $\& \in \Gamma$
q_{acc}:	accept state	q_{rej}:	reject state

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

Understanding δ

Formal definition

Definition

A Turing machine (TM) is given by $M = (Q, \Sigma, \Gamma, \delta, q_0, q_f, q_{rej})$

Q:	set of states	Σ:	input alphabet
q_0:	start state	Γ:	tape alphabet, $\Sigma \subseteq \Gamma$, $\& \in \Gamma$
q_{acc}:	accept state	q_{rej}:	reject state

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

Understanding δ

For a $q \in Q, a \in \Gamma$ if $\delta(q, a) = (p, b, L)$

then p is the new state of the machine,

b is the letter with which a gets overwritten,

the head moves to the left of the current position.

Configuration

Definition

The configuration of a TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_f, q_{rej})$ is given by

$$\Gamma^* \times Q \times \Gamma^*$$

Configuration

Definition

The configuration of a TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_f, q_{rej})$ is given by

$$\Gamma^* \times Q \times \Gamma^*$$

A configuration need not include blank symbols.

Configuration

Definition

The configuration of a TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_f, q_{rej})$ is given by

$$\Gamma^* \times Q \times \Gamma^*$$

A configuration need not include blank symbols.

- Let $u, v \in \Gamma^*$, $a, b, c \in \Gamma$ and $q, q' \in Q$.

Configuration

Definition

The configuration of a TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_f, q_{rej})$ is given by

$$\Gamma^* \times Q \times \Gamma^*$$

A configuration need not include blank symbols.

- Let $u, v \in \Gamma^*$, $a, b, c \in \Gamma$ and $q, q' \in Q$.
- Suppose $(q', c, L) \in \delta(q, b)$ is a transition in M

Configuration

Definition

The configuration of a TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_f, q_{rej})$ is given by

$$\Gamma^* \times Q \times \Gamma^*$$

A configuration need not include blank symbols.

- Let $u, v \in \Gamma^*$, $a, b, c \in \Gamma$ and $q, q' \in Q$.

- Suppose $(q', c, L) \in \delta(q, b)$ is a transition in M,
- then starting from $u \cdot a \cdot q \cdot b \cdot v$ in one step we get $u \cdot q' \cdot a \cdot c \cdot v$.

Configuration

Definition

The configuration of a TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_f, q_{rej})$ is given by

$$\Gamma^* \times Q \times \Gamma^*$$

A configuration need not include blank symbols.

- Let $u, v \in \Gamma^*$, $a, b, c \in \Gamma$ and $q, q' \in Q$.
- Suppose $(q', c, L) \in \delta(q, b)$ is a transition in M,
- then starting from $u \cdot a \cdot q \cdot b \cdot v$ in one step we get $u \cdot q' \cdot a \cdot c \cdot v$.
- We say that $u \cdot a \cdot q \cdot b \cdot v$ **yields** $u \cdot q' \cdot a \cdot c \cdot v$.

Configuration

Definition

The configuration of a TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_f, q_{rej})$ is given by

$$\Gamma^* \times Q \times \Gamma^*$$

A configuration need not include blank symbols.

- Let $u, v \in \Gamma^*$, $a, b, c \in \Gamma$ and $q, q' \in Q$.
- Suppose $(q', c, L) \in \delta(q, b)$ is a transition in M,
- then starting from $u \cdot a \cdot q \cdot b \cdot v$ in one step we get $u \cdot q' \cdot a \cdot c \cdot v$.
- We say that $u \cdot a \cdot q \cdot b \cdot v$ **yields** $u \cdot q' \cdot a \cdot c \cdot v$.
- We denote it by $u \cdot a \cdot q \cdot b \cdot v \mapsto u \cdot q' \cdot a \cdot c \cdot v$.

Special configurations

Start configuration

- We assume that the head is on the left of the input in the beginning.

Special configurations

Start configuration

- We assume that the head is on the left of the input in the beginning.
- Therefore, $q_0 \cdot w$ is the start configuration.

Special configurations

Start configuration

- We assume that the head is on the left of the input in the beginning.
- Therefore, $q_0 \cdot w$ is the start configuration.

Accepting configuration

Special configurations

Start configuration

- We assume that the head is on the left of the input in the beginning.
- Therefore, $q_0 \cdot w$ is the start configuration.

Accepting configuration

Any configulation that contains q_{acc} is an accepting configuration.

Special configurations

Start configuration

- We assume that the head is on the left of the input in the beginning.
- Therefore, $q_0 \cdot w$ is the start configuration.

Accepting configuration

Any configulation that contains q_{acc} is an accepting configuration.

Rejecting configuration

Special configurations

Start configuration

- We assume that the head is on the left of the input in the beginning.
- Therefore, $q_0 \cdot w$ is the start configuration.

Accepting configuration

Any configulation that contains q_{acc} is an accepting configuration.

Rejecting configuration

Any configulation that contains q_{rej} is a rejecting configuration.

Special configurations

Start configuration

- We assume that the head is on the left of the input in the beginning.
- Therefore, $q_0 \cdot w$ is the start configuration.

Accepting configuration

Any configulation that contains q_{acc} is an accepting configuration.

Rejecting configuration

Any configulation that contains q_{rej} is a rejecting configuration.

Halting configurations: if a configuration is accepting or rejecting then it is called a halting configuration.

Special configurations

Start configuration

- We assume that the head is on the left of the input in the beginning.
- Therefore, $q_0 \cdot w$ is the start configuration.

Accepting configuration

Any configulation that contains q_{acc} is an accepting configuration.

Rejecting configuration

Any configulation that contains q_{rej} is a rejecting configuration.

Halting configurations: if a configuration is accepting or rejecting then it is called a halting configuration.

A TM may not halt!

Acceptance by a TM

A TM M is said to accept a word $w \in \Sigma^*$ if there exists a sequence of configurations $C_0, C_1, \ldots, C_k$ such that

- C_0 is a start configuration

Acceptance by a TM

A TM M is said to accept a word $w \in \Sigma^*$ if there exists a sequence of configurations $C_0, C_1, \ldots, C_k$ such that

- C_0 is a start configuration,
- $C_i \mapsto C_{i+1}$ for all $0 \leq i \leq k-1$

Acceptance by a TM

A TM M is said to accept a word $w \in \Sigma^*$ if there exists a sequence of configurations $C_0, C_1, \ldots, C_k$ such that

- C_0 is a start configuration,
- $C_i \mapsto C_{i+1}$ for all $0 \leq i \leq k-1$,
- C_k is an accepting configuration.

Acceptance by a TM

A TM M is said to accept a word $w \in \Sigma^*$ if there exists a sequence of configurations $C_0, C_1, \ldots, C_k$ such that

- C_0 is a start configuration,
- $C_i \mapsto C_{i+1}$ for all $0 \leq i \leq k-1$,
- C_k is an accepting configuration.

Let $\rho = C_0, C_1, \ldots, C_k$ be a sequence of configuration of M on w.

Acceptance by a TM

A TM M is said to accept a word $w \in \Sigma^*$ if there exists a sequence of configurations $C_0, C_1, \ldots, C_k$ such that

- C_0 is a start configuration,
- $C_i \mapsto C_{i+1}$ for all $0 \leq i \leq k-1$,
- C_k is an accepting configuration.

Let $\rho = C_0, C_1, \ldots, C_k$ be a sequence of configuration of M on w.

This sequence ρ is called a run of the machine M on w.

Acceptance by a TM

A TM M is said to accept a word $w \in \Sigma^*$ if there exists a sequence of configurations $C_0, C_1, \ldots, C_k$ such that

- C_0 is a start configuration,
- $C_i \mapsto C_{i+1}$ for all $0 \leq i \leq k - 1$,
- C_k is an accepting configuration.

Let $\rho = C_0, C_1, \ldots, C_k$ be a sequence of configuration of M on w.

This sequence ρ is called a run of the machine M on w.

If C_k is an accepting configuration then ρ is called an accepting run.

Acceptance by a TM

A TM M is said to accept a word $w \in \Sigma^*$ if there exists a sequence of configurations $C_0, C_1, \ldots, C_k$ such that

- C_0 is a start configuration,
- $C_i \mapsto C_{i+1}$ for all $0 \leq i \leq k-1$,
- C_k is an accepting configuration.

Let $\rho = C_0, C_1, \ldots, C_k$ be a sequence of configuration of M on w.

This sequence ρ is called a run of the machine M on w.

If C_k is an accepting configuration then ρ is called an accepting run.

If C_k is a rejecting configuration then ρ is called a rejecting run.

Acceptance by a TM

A TM M is said to accept a word $w \in \Sigma^*$ if there exists a sequence of configurations $C_0, C_1, \ldots, C_k$ such that

- C_0 is a start configuration,
- $C_i \mapsto C_{i+1}$ for all $0 \leq i \leq k-1$,
- C_k is an accepting configuration.

Let $\rho = C_0, C_1, \ldots, C_k$ be a sequence of configuration of M on w.

This sequence ρ is called a run of the machine M on w.

If C_k is an accepting configuration then ρ is called an accepting run.

If C_k is a rejecting configuration then ρ is called a rejecting run.

Turing recognizable languages

Definition

A language L is said to be Turing recognizable if there is a Turing machine M such that

Turing recognizable languages

Definition

A language L is said to be Turing recognizable if there is a Turing machine M such that $\forall w \in L$, M has at least one accepting run on w.

Turing recognizable languages

Definition

A language L is said to be Turing recognizable if there is a Turing machine M such that $\forall w \in L$, M has at least one accepting run on w.

We say that M recognizes L.

For words not in L

Turing recognizable languages

Definition

A language L is said to be Turing recognizable if there is a Turing machine M such that $\forall w \in L$, M has at least one accepting run on w.

We say that M recognizes L.

For words not in L

the machine may run forever

Turing recognizable languages

Definition

A language L is said to be Turing recognizable if there is a Turing machine M such that $\forall w \in L$, M has at least one accepting run on w.

We say that M recognizes L.

For words not in L

the machine may run forever,

or may reach q_{rej}

Turing recognizable languages

Definition

A language L is said to be Turing recognizable if there is a Turing machine M such that $\forall w \in L$, M has at least one accepting run on w.

We say that M recognizes L.

For words not in L

the machine may run forever,

or may reach q_{rej},

both are valid outcomes

Turing recognizable languages

Definition

A language L is said to be Turing recognizable if there is a Turing machine M such that $\forall w \in L$, M has at least one accepting run on w.

We say that M recognizes L.

For words not in L

the machine may run forever,

or may reach q_{rej},

both are valid outcomes,

and the machine is allowed to do either of the two.

Turning decidable languages

Definition

A language L is said to be Turing decidable if there is a Turing machine M such that

Turning decidable languages

Definition

A language L is said to be Turing decidable if there is a Turing machine M such that for all $w \in \Sigma^*$, M halts on w

Turning decidable languages

Definition

A language L is said to be Turing decidable if there is a Turing machine M such that for all $w \in \Sigma^*$, M halts on w and

if $w \in L$, M has an accepting run on w.

Turning decidable languages

Definition

A language L is said to be Turing decidable if there is a Turing machine M such that for all $w \in \Sigma^*$, M halts on w and

if $w \in L$, M has an accepting run on w.

if $w \notin L$, all runs of M on w are rejecting runs.

Turning decidable languages

Definition

A language L is said to be Turing decidable if there is a Turing machine M such that for all $w \in \Sigma^*$, M halts on w and

if $w \in L$, M has an accepting run on w.

if $w \notin L$, all runs of M on w are rejecting runs.

We say that M decides L.

Turning decidable languages

Definition

A language L is said to be Turing decidable if there is a Turing machine M such that for all $w \in \Sigma^*$, M halts on w and

if $w \in L$, M has an accepting run on w.

if $w \notin L$, all runs of M on w are rejecting runs.

We say that M decides L.

If a language L is Turing decidable then

Turning decidable languages

Definition

A language L is said to be Turing decidable if there is a Turing machine M such that for all $w \in \Sigma^*$, M halts on w and

if $w \in L$, M has an accepting run on w.

if $w \notin L$, all runs of M on w are rejecting runs.

We say that M decides L.

If a language L is Turing decidable then

the TM deciding L always halts.

Turning decidable languages

Definition

A language L is said to be Turing decidable if there is a Turing machine M such that for all $w \in \Sigma^*$, M halts on w and

if $w \in L$, M has an accepting run on w.

if $w \notin L$, all runs of M on w are rejecting runs.

We say that M decides L.

If a language L is Turing decidable then

the TM deciding L always halts.

L is also Turing recognizable.

Turning decidable languages

Definition

A language L is said to be Turing decidable if there is a Turing machine M such that for all $w \in \Sigma^*$, M halts on w and

if $w \in L$, M has an accepting run on w.

if $w \notin L$, all runs of M on w are rejecting runs.

We say that M decides L.

If a language L is Turing decidable then

the TM deciding L always halts.

L is also Turing recognizable.

Turing decidable languages form a subclass of Turing recognizable languages.

Turing machine for a non-context free language

Example

Turing machine for a non-context free language

Example

$$\mathsf{EQ} = \{w \cdot \# \cdot w \mid w \in \Sigma^*\}.$$

Turing machine for a non-context free language

Example

$\text{EQ} = \{w \cdot \# \cdot w \mid w \in \Sigma^*\}$.

On input string w:

1. Check if corresponding positions on either side of the # symbol are the same character. If not, or if no # is found, reject. Cross off symbols as they are checked.

Turing machine for a non-context free language

Example

EQ $= \{w \cdot \# \cdot w \mid w \in \Sigma^*\}$.

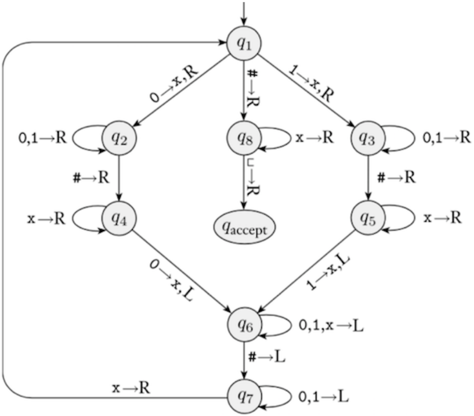
On input string w:

1. Check if corresponding positions on either side of the $\#$ symbol are the same character. If not, or if no $\#$ is found, reject. Cross off symbols as they are checked.
2. When all symbols to the left of the $\#$ have been crossed off, check for any remaining symbols to the right of the $\#$. If any symbols remain, reject; otherwise, accept.

Turing machine for a non-context free language

Example

$$\mathsf{EQ} = \{w \cdot \# \cdot w \mid w \in \Sigma^*\}.$$

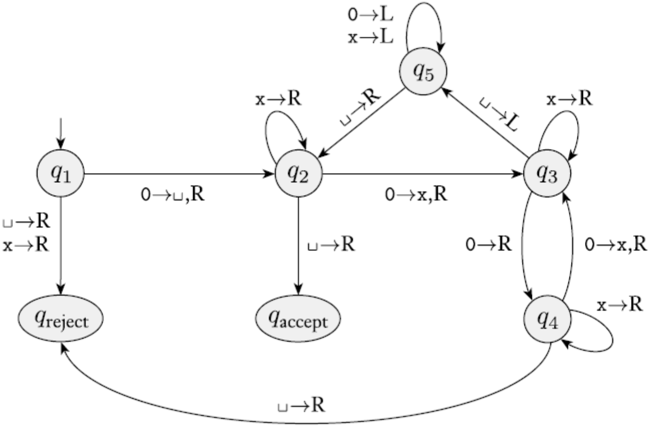


More examples

- $L_1 = \{0^{2^n} \mid n \in \mathbb{N}\}$

More examples

- $L_1 = \{0^{2^n} \mid n \in \mathbb{N}\}$



More examples

- $L_1 = \{0^{2^n} \mid n \in \mathbb{N}\}$

- $L_2 = \{a^i b^j c^k \mid i \times j = k$ and $i, j, k \geq 1\}$.

- $L_3 = \{0^n \mid n$ prime$\}$