CS310 Automata Theory – 2016-2017

Nutan Limaye

Indian Institute of Technology, Bombay
nutan@cse.iitb.ac.in

Lecture 21: Turing machines, computability
March 07, 2017

Last two modules

Regular languages, NFA/DFA, Regular expressions, Myhill-Nerode relations.

Last two modules

Regular languages, NFA/DFA, Regular expressions, Myhill-Nerode relations.

2DFA: DFA + two-way head movement.

Last two modules

Regular languages, NFA/DFA, Regular expressions, Myhill-Nerode relations.

2DFA: DFA + two-way head movement. They recognize exactly regular languages.

Last two modules

Regular languages, NFA/DFA, Regular expressions, Myhill-Nerode relations.

2DFA: DFA + two-way head movement. They recognize exactly regular languages.

Pushdown automata: NFA + Stack.

Last two modules

Regular languages, NFA/DFA, Regular expressions, Myhill-Nerode relations.

2DFA: DFA + two-way head movement. They recognize exactly regular languages.

Pushdown automata: NFA + Stack. The class of languages recognized by these is called Context-free languages (CFLs).

Last two modules

Regular languages, NFA/DFA, Regular expressions, Myhill-Nerode relations.

2DFA: DFA + two-way head movement. They recognize exactly regular languages.

Pushdown automata: NFA + Stack. The class of languages recognized by these is called Context-free languages (CFLs).

Context-free grammars: Recursive programs.

Last two modules

Regular languages, NFA/DFA, Regular expressions, Myhill-Nerode relations.

2DFA: DFA + two-way head movement. They recognize exactly regular languages.

Pushdown automata: NFA + Stack. The class of languages recognized by these is called Context-free languages (CFLs).

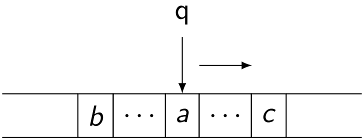
Context-free grammars: Recursive programs. The class of languages generated by these grammars is CFLs.

Turing machines

What is a Turing machine? (Informal description.)

Turing machines

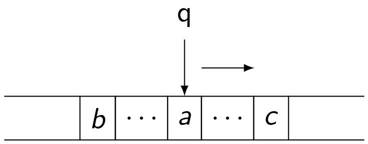
What is a Turing machine? (Informal description.)



Read and write on the input tape.

Turing machines

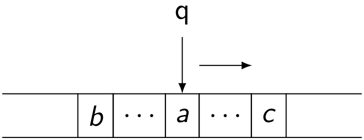
What is a Turing machine? (Informal description.)



Read and write on the input tape. Head moves left/right.

Turing machines

What is a Turing machine? (Informal description.)

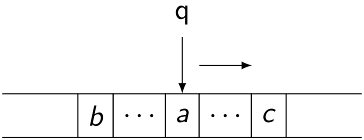


Read and write on the input tape. Head moves left/right.

The tape is infinite.

Turing machines

What is a Turing machine? (Informal description.)



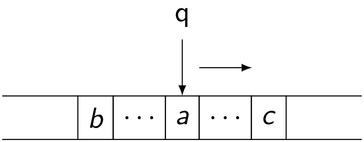
Read and write on the input tape. Head moves left/right.

The tape is infinite.

A special symbol & to indicate blank cells.

Turing machines

What is a Turing machine? (Informal description.)



Read and write on the input tape. Head moves left/right.

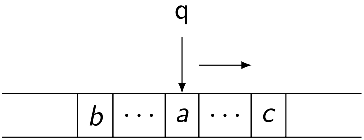
The tape is infinite.

A special symbol & to indicate blank cells.

Initially all cells blank except the part where the input is written.

Turing machines

What is a Turing machine? (Informal description.)



Read and write on the input tape. Head moves left/right.

The tape is infinite.

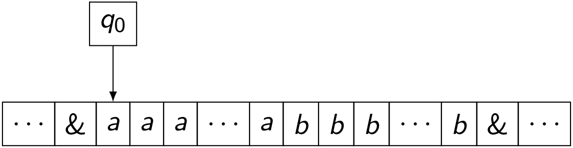
A special symbol & to indicate blank cells.

Initially all cells blank except the part where the input is written.

Special states for accepting and rejecting.

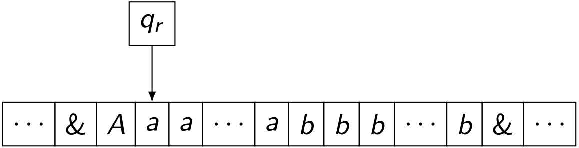
Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.



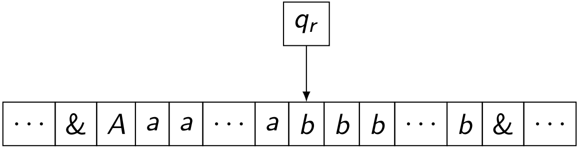
Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.



Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.



Example

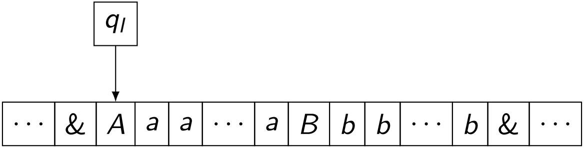
$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.

q_I

$\cdots$	&	A	a	a	$\cdots$	a	B	b	b	$\cdots$	b	&	$\cdots$

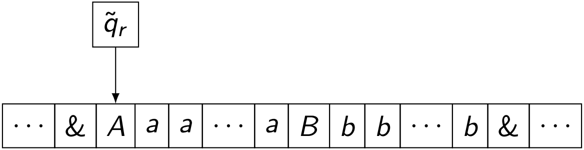
Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.



Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}.$



Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.

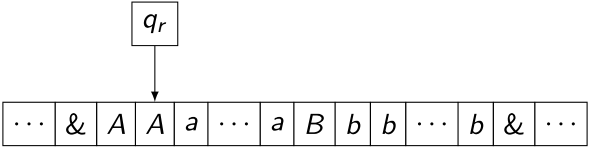
$\tilde{q}_r$

$\downarrow$

$\cdots$	&	A	a	a	$\cdots$	a	B	b	b	$\cdots$	b	&	$\cdots$

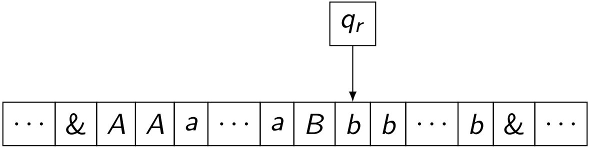
Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.



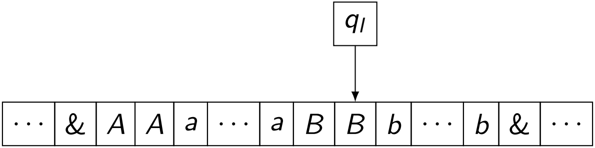
Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.



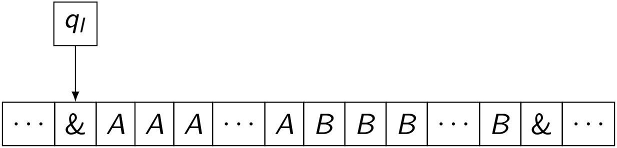
Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}.$



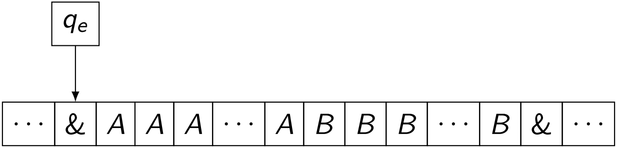
Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.



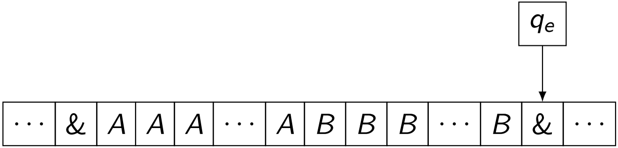
Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.



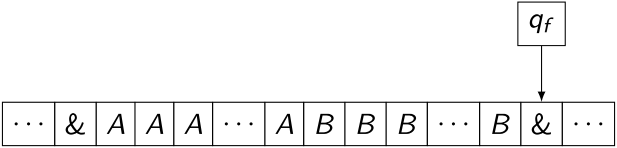
Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.



Example

$L_{a,b} = \{a^n b^n \mid n \geq 0\}$.



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

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

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		

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

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, $\Sigma \subseteq \Gamma$

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, $\Sigma \subseteq \Gamma$, & $\in \Gamma$

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		

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

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

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

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

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

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

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

Understanding δ

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

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

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

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

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

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

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

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

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

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.

Turing machine for a non-context free language

Example

Turing machine for a non-context free language

Example

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

Turing machine for a non-context free language

Example

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

Give a full description of a Turing machine for the above language.

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.

The notion of rejection by TM is not as straightforward!

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 reaches an accepting configuration 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 reaches an accepting configuration 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 reaches an accepting configuration 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 reaches an accepting configuration 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 reaches an accepting configuration 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 reaches an accepting configuration 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

$$\forall w \in L,\ M \text{ reaches the accepting configuration on } w.$$

Turning decidable languages

Definition

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

$$\forall w \in L,\ M \text{ reaches the accepting configuration on } w.$$

$$\forall w \notin L,\ M \text{ reaches the rejecting configuration on } w.$$

Turning decidable languages

Definition

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

$$\forall w \in L, \ M \text{ reaches the accepting configuration on } w.$$

$$\forall w \notin L, \ M \text{ reaches the rejecting configuration on } w.$$

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

$$\forall w \in L,\ M \text{ reaches the accepting configuration on } w.$$

$$\forall w \notin L,\ M \text{ reaches the rejecting configuration on } w.$$

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

$$\forall w \in L,\ M \text{ reaches the accepting configuration on } w.$$

$$\forall w \notin L,\ M \text{ reaches the rejecting configuration on } w.$$

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

$\forall w \in L$, M reaches the accepting configuration on w.

$\forall w \notin L$, M reaches the rejecting configuration on w.

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

$\forall w \in L$, M reaches the accepting configuration on w.

$\forall w \notin L$, M reaches the rejecting configuration on w.

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.

Comparing decidability and recognizability

Theorem

A language L is Turing decidable if and only if L and $\overline{L}$ are both Turing recognizable.

Comparing decidability and recognizability

Theorem

A language L is Turing decidable if and only if L and $\overline{L}$ are both Turing recognizable.