COL 352 Introduction to Automata and Theory of Computation

Nikhil Balaji

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

February 21, 2023

Lecture 13: Myhill-Nerode Theorem

Recap of Module 1

Recap of Module 1

- DFA, NFA, Regular expressions and their equivalence.

Recap of Module 1

- DFA, NFA, Regular expressions and their equivalence.
- Closure properties of regular languages.

Recap of Module 1

- DFA, NFA, Regular expressions and their equivalence.
- Closure properties of regular languages.
- Non-regular languages and Pigeon Hole Principle.

Recap of Module 1

- DFA, NFA, Regular expressions and their equivalence.
- Closure properties of regular languages.
- Non-regular languages and Pigeon Hole Principle.
- Pumping lemma and its applications.

Recap of Module 1

- DFA, NFA, Regular expressions and their equivalence.
- Closure properties of regular languages.
- Non-regular languages and Pigeon Hole Principle.
- Pumping lemma and its applications.
- Myhill Nerode relation and characterization of regular languages.

Recap of Module 1

- DFA, NFA, Regular expressions and their equivalence.
- Closure properties of regular languages.
- Non-regular languages and Pigeon Hole Principle.
- Pumping lemma and its applications.
- Myhill Nerode relation and characterization of regular languages.
- Algorithms for membership problem, emptiness problem and minimization problem.

Moving on

How do we add expressive power to DFA/NFA so that we can compute more functions?

2DFA: Two-way deterministic finite state automata

$\# \;\; w_1 \;\; w_2 \;\; \ldots \;\; \ldots \;\; \ldots \;\; \ldots \;\; \ldots \;\; \ldots \;\; w_n \;\; \$$

Input head moves left/right on this tape.

2DFA: Two-way deterministic finite state automata

$$\boxed{\# \; w_1 \; w_2 \; \ldots \; \ldots \; \ldots \; \ldots \; \ldots \; \ldots \; w_n \; \$}$$

Input head moves left/right on this tape.

It does not go to the left of $\#$.

2DFA: Two-way deterministic finite state automata

$\# \; w_1 \; w_2 \; \ldots \; \ldots \; \ldots \; \ldots \; \ldots \; \ldots \; w_n \; \$$

Input head moves left/right on this tape.

It does not go to the left of $\#$.

It does not go to the right of $\$$.

2DFA: Two-way deterministic finite state automata

$\# \;\; w_1 \;\; w_2 \;\; \dots \;\; \dots \;\; \dots \;\; \dots \;\; \dots \;\; \dots \;\; w_n \;\; \$$

Input head moves left/right on this tape.

It does not go to the left of $\#$.

It does not go to the right of \$.

Can potentially get stuck in an infinite loop!

Example

$$A = \{x \in \{a,b\}^* \mid \#_a(x) = 0 \bmod 3 \;\; \text{and} \;\; \#_b(x) = 0 \bmod 2\}$$

Example

$$A = \{x \in \{a,b\}^* \mid \#_a(x) = 0 \bmod 3 \ \text{ and } \#_b(x) = 0 \bmod 2\}$$

- Start state is at $\#$.
- Scan from left to right only counting number of $a \bmod 3$ ignoring b until \$ is encountered.
- If the number of a seen is not $0 \bmod 3$, then reject.

Example

$$A = \{x \in \{a,b\}^* \mid \#_a(x) = 0 \bmod 3 \;\; \text{and} \;\; \#_b(x) = 0 \bmod 2\}$$

- Start state is at $\#$.
- Scan from left to right only counting number of $a \bmod 3$ ignoring b until \$ is encountered.
- If the number of a seen is not $0 \bmod 3$, then reject.
- If the number of a seen is $0 \bmod 3$, move in the reverse direction now counting only $b \bmod 2$ ignoring a.
- If the number of b seen is not $0 \bmod 2$, then accept, else reject.

Example

$$A = \{x \in \{a,b\}^* \mid \#_a(x) = 0 \bmod 3 \ \text{ and } \#_b(x) = 0 \bmod 2\}$$

- Start state is at $\#$.
- Scan from left to right only counting number of $a \bmod 3$ ignoring b until \$ is encountered.
- If the number of a seen is not $0 \bmod 3$, then reject.
- If the number of a seen is $0 \bmod 3$, move in the reverse direction now counting only $b \bmod 2$ ignoring a.
- If the number of b seen is not $0 \bmod 2$, then accept, else reject.

Note: Needs only one start and accept state. Halts immediately after entering accept state.

Example

$$A = \{x \in \{a,b\}^* \mid \#_a(x) = 0 \bmod 3 \ \text{ and } \#_b(x) = 0 \bmod 2\}$$

- Start state is at $\#$.
- Scan from left to right only counting number of $a \bmod 3$ ignoring b until \$ is encountered.
- If the number of a seen is not $0 \bmod 3$, then reject.
- If the number of a seen is $0 \bmod 3$, move in the reverse direction now counting only $b \bmod 2$ ignoring a.
- If the number of b seen is not $0 \bmod 2$, then accept, else reject.

Note: Needs only one start and accept state. Halts immediately after entering accept state. Can also loop indefinitely.

Formal definition of 2DFA

Definition

A 2DFA $A = (Q, \Sigma \cup \{\#, \$\}, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$

Formal definition of 2DFA

Definition

A 2DFA $A = (Q, \Sigma \cup \{\#, \$\}, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$, where

Q : set of states, Σ: input alphabet

Formal definition of 2DFA

Definition

A 2DFA $A = (Q, \Sigma \cup \{\#, \$\}, \delta, q_0, q_{\mathsf{acc}}, q_{\mathsf{rej}})$, where

Q :	set of states,	Σ:	input alphabet
$\#$:	left endmarker	$\$$:	right endmarker

Formal definition of 2DFA

Definition

A 2DFA $A = (Q, \Sigma \cup \{\#, \$\}, \delta, q_0, q_{\mathsf{acc}}, q_{\mathsf{rej}})$, where

Q :	set of states,	Σ:	input alphabet
$\#$:	left endmarker	$\$$:	right endmarker
q_0:	start state		

Formal definition of 2DFA

Definition

A 2DFA $A = (Q, \Sigma \cup \{\#, \$\}, \delta, q_0, q_{\mathsf{acc}}, q_{\mathsf{rej}})$, where

Q :	set of states,	Σ:	input alphabet
$\#$:	left endmarker	$\$$:	right endmarker
q_0:	start state		
q_{acc}:	accept state	q_{rej}:	reject state

Formal definition of 2DFA

Definition

A 2DFA $A = (Q, \Sigma \cup \{\#, \$\}, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$, where

Q :	set of states,	Σ:	input alphabet
$\#$:	left endmarker	$\$$:	right endmarker
q_0:	start state		
q_{acc}:	accept state	q_{rej}:	reject state

$$\delta : Q \times (\Sigma \cup \{\#, \$\} \rightarrow Q \times \{L, R\}$$

Formal definition of 2DFA

Definition

A 2DFA $A = (Q, \Sigma \cup \{\#, \$\}, \delta, q_0, q_{\mathsf{acc}}, q_{\mathsf{rej}})$, where

Q :	set of states,	Σ:	input alphabet
$\#$:	left endmarker	$\$$:	right endmarker
q_0:	start state		
q_{acc}:	accept state	q_{rej}:	reject state

$$\delta : Q \times (\Sigma \cup \{\#, \$\} \to Q \times \{L, R\}$$

The following conditions are forced:

$\forall q \in Q,\ \exists q', q'' \in Q$ s.t. $\delta(q, \#) = (q', R)$

Formal definition of 2DFA

Definition

A 2DFA $A = (Q, \Sigma \cup \{\#, \$\}, \delta, q_0, q_{\mathsf{acc}}, q_{\mathsf{rej}})$, where

Q :	set of states,	Σ:	input alphabet
$\#$:	left endmarker	$\$$:	right endmarker
q_0:	start state		
q_{acc}:	accept state	q_{rej}:	reject state

$$\delta : Q \times (\Sigma \cup \{\#, \$\} \to Q \times \{L, R\}$$

The following conditions are forced:

$\forall q \in Q$, $\exists q', q'' \in Q$ s.t. $\delta(q, \#) = (q', R)$ and $\delta(q, \$) = (q'', L)$.

Formal definition of 2DFA

Definition

A 2DFA $A = (Q, \Sigma \cup \{\#, \$\}, \delta, q_0, q_{\mathsf{acc}}, q_{\mathsf{rej}})$, where

Q :	set of states,	Σ:	input alphabet
$\#$:	left endmarker	$\$$:	right endmarker
q_0:	start state		
q_{acc}:	accept state	q_{rej}:	reject state

$$\delta : Q \times (\Sigma \cup \{\#, \$\} \to Q \times \{L, R\}$$

The following conditions are forced:

$\forall q \in Q$, $\exists q', q'' \in Q$ s.t. $\delta(q, \#) = (q', R)$ and $\delta(q, \$) = (q'', L)$.

Exercise: Write down a formal definition of the example from previous slide.

Configurations

Fix input $x = a_1 \ldots x_n$, let $a_0 = \#, a_{n+1} = \$$. So $a_0 a_1 \ldots a_n a_{n+1} = \# x \$$.

Configurations

Fix input $x = a_1 \ldots x_n$, let $a_0 = \#, a_{n+1} = \$$. So $a_0 a_1 \ldots a_n a_{n+1} = \#x\$$.

- Configuration: $(q, i) \in Q \times \mathbb{N}$, where $0 \leq i \leq n + 1$.

Configurations

Fix input $x = a_1 \dots x_n$, let $a_0 = \#, a_{n+1} = \$$. So $a_0 a_1 \dots a_n a_{n+1} = \#x\$$.

- Configuration: $(q, i) \in Q \times \mathbb{N}$, where $0 \leq i \leq n + 1$.
- Start configuration: $(q_0, 0)$.

Configurations

Fix input $x = a_1 \dots x_n$, let $a_0 = \#, a_{n+1} = \$$. So $a_0 a_1 \dots a_n a_{n+1} = \#x\$$.

- Configuration: $(q, i) \in Q \times \mathbb{N}$, where $0 \leq i \leq n + 1$.
- Start configuration: $(q_0, 0)$.
- Binary relation: $\xrightarrow[1]{x}$ defined as

  $\delta(p, a_i) = (q, L) \implies (p, i) \xrightarrow[1]{x} (q, i - 1)$
- $\delta(p, a_i) = (q, R) \implies (p, i) \xrightarrow[1]{x} (q, i + 1)$
- Inductively define n step relation $\xrightarrow[1]{x}$ as:

Configurations

Fix input $x = a_1 \dots x_n$, let $a_0 = \#, a_{n+1} = \$$. So $a_0 a_1 \dots a_n a_{n+1} = \#x\$$.

- Configuration: $(q, i) \in Q \times \mathbb{N}$, where $0 \leq i \leq n + 1$.
- Start configuration: $(q_0, 0)$.
- Binary relation: $\xrightarrow[1]{x}$ defined as

  $\delta(p, a_i) = (q, L) \implies (p, i) \xrightarrow[1]{x} (q, i - 1)$
- $\delta(p, a_i) = (q, R) \implies (p, i) \xrightarrow[1]{x} (q, i + 1)$
- Inductively define n step relation $\xrightarrow[1]{x}$ as:

  $(p, i) \xrightarrow[0]{x} (p, i)$

  $(p, i) \xrightarrow[n]{x} (q, j)$ and $(q, j) \xrightarrow[1]{x} (u, k) \implies (p, i) \xrightarrow[n+1]{x} (u, k)$.

Configurations

Fix input $x = a_1 \ldots x_n$, let $a_0 = \#, a_{n+1} = \$$. So $a_0 a_1 \ldots a_n a_{n+1} = \#x\$$.

- Configuration: $(q, i) \in Q \times \mathbb{N}$, where $0 \leq i \leq n + 1$.
- Start configuration: $(q_0, 0)$.
- Binary relation: $\xrightarrow[1]{x}$ defined as

  $$\delta(p, a_i) = (q, L) \implies (p, i) \xrightarrow[1]{x} (q, i - 1)$$

- $\delta(p, a_i) = (q, R) \implies (p, i) \xrightarrow[1]{x} (q, i + 1)$
- Inductively define n step relation $\xrightarrow[1]{x}$ as:

  $$(p, i) \xrightarrow[0]{x} (p, i)$$

  $(p, i) \xrightarrow[n]{x} (q, j)$ and $(q, j) \xrightarrow[1]{x} (u, k) \implies (p, i) \xrightarrow[n+1]{x} (u, k)$.

-
  $$(p, i) \xrightarrow[*]{x} (q, j) \iff \exists n \geq 0 (p, i) \xrightarrow[n]{x} (q, j)$$

Acceptance by 2DFA

Definition

Let A be a 2DFA.

- A word w is said to be accepted by A if A reaches q_{acc} on w.

  That is, there exists some i such that $(q_0, 0) \underset{*}{\rightarrow} (q_{acc}, i)$

Acceptance by 2DFA

Definition

Let A be a 2DFA.

- A word w is said to be accepted by A if A reaches q_{acc} on w. That is, there exists some i such that $(q_0, 0) \underset{*}{\rightarrow} (q_{acc}, i)$
- A word w is said to be rejected by A if A reaches q_{rej} on w. That is, there exists some i such that $(q_0, 0) \underset{*}{\rightarrow} (q_{rej}, i)$

Acceptance by 2DFA

Definition

Let A be a 2DFA.

- A word w is said to be accepted by A if A reaches q_{acc} on w.
  That is, there exists some i such that $(q_0, 0) \underset{*}{\rightarrow} (q_{acc}, i)$
- A word w is said to be rejected by A if A reaches q_{rej} on w.
  That is, there exists some i such that $(q_0, 0) \underset{*}{\rightarrow} (q_{rej}, i)$
- A is said to accept a language L if $\forall w \in L$, A reaches q_{acc}.

Acceptance by 2DFA

Definition

Let A be a 2DFA.

- A word w is said to be accepted by A if A reaches q_{acc} on w.
  That is, there exists some i such that $(q_0, 0) \underset{*}{\rightarrow} (q_{acc}, i)$
- A word w is said to be rejected by A if A reaches q_{rej} on w.
  That is, there exists some i such that $(q_0, 0) \underset{*}{\rightarrow} (q_{rej}, i)$
- A is said to accept a language L if $\forall w \in L$, A reaches q_{acc}.

2DFA may loop forever if $w \notin L$ or may enter q_{rej}.

Acceptance by 2DFA

Definition

Let A be a 2DFA.

- A word w is said to be accepted by A if A reaches q_{acc} on w.
  That is, there exists some i such that $(q_0, 0) \underset{*}{\rightarrow} (q_{acc}, i)$
- A word w is said to be rejected by A if A reaches q_{rej} on w.
  That is, there exists some i such that $(q_0, 0) \underset{*}{\rightarrow} (q_{rej}, i)$
- A is said to accept a language L if $\forall w \in L$, A reaches q_{acc}.

2DFA may loop forever if $w \notin L$ or may enter q_{rej}.

Lemma

If L is regular then there is a 2DFA accepting L.

Acceptance by 2DFA

Definition

Let A be a 2DFA.

- A word w is said to be accepted by A if A reaches q_{acc} on w.
  That is, there exists some i such that $(q_0, 0) \xrightarrow[*]{} (q_{acc}, i)$
- A word w is said to be rejected by A if A reaches q_{rej} on w.
  That is, there exists some i such that $(q_0, 0) \xrightarrow[*]{} (q_{rej}, i)$
- A is said to accept a language L if $\forall w \in L$, A reaches q_{acc}.

2DFA may loop forever if $w \notin L$ or may enter q_{rej}.

Lemma

If L is regular then there is a 2DFA accepting L.

This holds trivially.

2DFA: Two-way deterministic finite state automata

Example

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}.$

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA is best described by giving its δ function.

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA is best described by giving its δ function.

Assume that initially the input head is on $\#$.

Read the input and move right till \$ is encountered.

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA is best described by giving its δ function.

Assume that initially the input head is on $\#$.

Read the input and move right till \$ is encountered.

Up on seeing \$ move left two positions.

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA is best described by giving its δ function.

Assume that initially the input head is on $\#$.

Read the input and move right till \$ is encountered.

Up on seeing \$ move left two positions.

If that letter is a, then accept else reject.

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA is best described by giving its δ function.

Assume that initially the input head is on $\#$.

- Read the input and move right till \$ is encountered.
- Up on seeing \$ move left two positions.
- If that letter is a, then accept else reject.
- Handle other corner cases such as the word length is less than 2.

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA is best described by giving its δ function.

Assume that initially the input head is on $\#$.

- Read the input and move right till \$ is encountered.
- Up on seeing \$ move left two positions.
- If that letter is a, then accept else reject.
- Handle other corner cases such as the word length is less than 2.

2DFA: Two-way deterministic finite state automata

Example

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA is best described by giving its δ function.

Assume that initially the input head is on $\#$.

The description of δ for the 2DFA for L_1 is given below.

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA is best described by giving its δ function.

Assume that initially the input head is on $\#$.

The description of δ for the 2DFA for L_1 is given below.

$\delta(q_0, \#) = (q_1, R)$
$\delta(q_1, \$) = (q_{rej}, L)$
$\delta(q_1, c) = (q_2, R)$ for all $c \in \Sigma$
$\delta(q_2, \$) = (q_{rej}, L)$
$\delta(q_2, c) = (q_3, R)$ for all $c \in \Sigma$
$\delta(q_3, c) = (q_3, R)$ for all $c \in \Sigma$
$\delta(q_3, \$) = (q_4, L)$

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA is best described by giving its δ function.

Assume that initially the input head is on $\#$.

The description of δ for the 2DFA for L_1 is given below.

$\delta(q_0, \#) = (q_1, R)$
$\delta(q_1, \$) = (q_{rej}, L)$
$\delta(q_1, c) = (q_2, R)$ for all $c \in \Sigma$
$\delta(q_2, \$) = (q_{rej}, L)$
$\delta(q_2, c) = (q_3, R)$ for all $c \in \Sigma$
$\delta(q_3, c) = (q_3, R)$ for all $c \in \Sigma$
$\delta(q_3, \$) = (q_4, L)$
$\delta(q_4, c) = (q_5, L)$ for all $c \in \Sigma$
$\delta(q_5, a) = (q_{acc}, L)$
$\delta(q_5, c) = (q_{rej}, L)$ for all $c \in (\Sigma \setminus \{a\})$.

2DFA: Two-way deterministic finite state automata

Example

$L_1 = \{w \in \Sigma^* \mid \text{the second last letter is } a\}$.

2DFA is best described by giving its δ function.

Assume that initially the input head is on $\#$.

The description of δ for the 2DFA for L_1 is given below.

$\delta(q_0, \#) = (q_1, R)$
$\delta(q_1, \$) = (q_{rej}, L)$
$\delta(q_1, c) = (q_2, R)$ for all $c \in \Sigma$
$\delta(q_2, \$) = (q_{rej}, L)$
$\delta(q_2, c) = (q_3, R)$ for all $c \in \Sigma$
$\delta(q_3, c) = (q_3, R)$ for all $c \in \Sigma$
$\delta(q_3, \$) = (q_4, L)$
$\delta(q_4, c) = (q_5, L)$ for all $c \in \Sigma$
$\delta(q_5, a) = (q_{acc}, L)$
$\delta(q_5, c) = (q_{rej}, L)$ for all $c \in (\Sigma \setminus \{a\})$.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

2DFA keeps two copies of states of A, say Q^1 and Q^2.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

2DFA keeps two copies of states of A, say Q^1 and Q^2. Additionally it has an extra start state q_0' and $|Q|$ many special state $q_{\leftarrow,i}$ for each $1 \leq i \leq |Q|$.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

2DFA keeps two copies of states of A, say Q^1 and Q^2. Additionally it has an extra start state q_0' and $|Q|$ many special state $q_{\leftarrow,i}$ for each $1 \le i \le |Q|$.

The first copy Q^1 is used to read the whole input the first time and do as per δ in that copy.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

2DFA keeps two copies of states of A, say Q^1 and Q^2. Additionally it has an extra start state q_0' and $|Q|$ many special state $q_{\leftarrow,i}$ for each $1 \leq i \leq |Q|$.

The first copy Q^1 is used to read the whole input the first time and do as per δ in that copy.Suppose we reach state q_i^1 at the end.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

2DFA keeps two copies of states of A, say Q^1 and Q^2. Additionally it has an extra start state q_0' and $|Q|$ many special state $q_{\leftarrow,i}$ for each $1 \leq i \leq |Q|$.

The first copy Q^1 is used to read the whole input the first time and do as per δ in that copy.Suppose we reach state q_i^1 at the end.

The special state $q_{\leftarrow,i}$ is used to move left after having read the input once.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

2DFA keeps two copies of states of A, say Q^1 and Q^2. Additionally it has an extra start state q_0' and $|Q|$ many special state $q_{\leftarrow,i}$ for each $1 \le i \le |Q|$.

The first copy Q^1 is used to read the whole input the first time and do as per δ in that copy.Suppose we reach state q_i^1 at the end.

The special state $q_{\leftarrow,i}$ is used to move left after having read the input once.

The second copy Q^2 is used to read the input the second time.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

2DFA keeps two copies of states of A, say Q^1 and Q^2. Additionally it has an extra start state q_0' and $|Q|$ many special state $q_{\leftarrow,i}$ for each $1 \leq i \leq |Q|$.

The first copy Q^1 is used to read the whole input the first time and do as per δ in that copy.Suppose we reach state q_i^1 at the end.

The special state $q_{\leftarrow,i}$ is used to move left after having read the input once.

The second copy Q^2 is used to read the input the second time.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

Start from a special start state q_0'.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

Start from a special start state q_0'.

Reading $\#$ move to the state q_0^1 and move right.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

Start from a special start state q_0'.

Reading $\#$ move to the state q_0^1 and move right.

Do exactly as per δ in Q^1 till \$ is encontered. Say the state reached is q_i^1 just before reading \$.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

Start from a special start state q_0'.

Reading $\#$ move to the state q_0^1 and move right.

Do exactly as per δ in Q^1 till \$ is encontered. Say the state reached is q_i^1 just before reading \$.

Upon seeing \$, move to a special state $q_{\leftarrow,i}$ and left.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

Start from a special start state q_0'.

Reading $\#$ move to the state q_0^1 and move right.

Do exactly as per δ in Q^1 till \$ is encontered. Say the state reached is q_i^1 just before reading \$.

Upon seeing \$, move to a special state $q_{\leftarrow,i}$ and left.

In $q_{\leftarrow,i}$, reading any letter (other than $\#$), stay in $q_{\leftarrow,i}$ and move left.

2DFA: Two-way deterministic finite state automata

Examples

- Let $\Sigma = \{a, b\}$ and L be a regular language.
- $L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

- Start from a special start state q_0'.
- Reading $\#$ move to the state q_0^1 and move right.
- Do exactly as per δ in Q^1 till \$ is encontered. Say the state reached is q_i^1 just before reading \$.
- Upon seeing \$, move to a special state $q_{\leftarrow,i}$ and left.
- In $q_{\leftarrow,i}$, reading any letter (other than $\#$), stay in $q_{\leftarrow,i}$ and move left.
- In $q_{\leftarrow,i}$, reading $\#$, move to the state q_i^2 and move right.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

Start from a special start state q_0'.

Reading $\#$ move to the state q_0^1 and move right.

Do exactly as per δ in Q^1 till \$ is encontered. Say the state reached is q_i^1 just before reading \$.

Upon seeing \$, move to a special state $q_{\leftarrow,i}$ and left.

In $q_{\leftarrow,i}$, reading any letter (other than $\#$), stay in $q_{\leftarrow,i}$ and move left.

In $q_{\leftarrow,i}$, reading $\#$, move to the state q_i^2 and move right.

Do exactly as per δ in Q^2 till \$ is encontered.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

Start from a special start state q_0'.

Reading $\#$ move to the state q_0^1 and move right.

Do exactly as per δ in Q^1 till \$ is encontered. Say the state reached is q_i^1 just before reading \$.

Upon seeing \$, move to a special state $q_{\leftarrow,i}$ and left.

In $q_{\leftarrow,i}$, reading any letter (other than $\#$), stay in $q_{\leftarrow,i}$ and move left.

In $q_{\leftarrow,i}$, reading $\#$, move to the state q_i^2 and move right.

Do exactly as per δ in Q^2 till \$ is encontered.

If the state is q_f^2, where $q_f \in F$, when reading \$ then go to state q_{acc} and move left

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. The 2DFA for L_2 works as follows:

Start from a special start state q_0'.

Reading $\#$ move to the state q_0^1 and move right.

Do exactly as per δ in Q^1 till \$ is encontered. Say the state reached is q_i^1 just before reading \$.

Upon seeing \$, move to a special state $q_{\leftarrow,i}$ and left.

In $q_{\leftarrow,i}$, reading any letter (other than $\#$), stay in $q_{\leftarrow,i}$ and move left.

In $q_{\leftarrow,i}$, reading $\#$, move to the state q_i^2 and move right.

Do exactly as per δ in Q^2 till \$ is encontered.

If the state is q_f^2, where $q_f \in F$, when reading \$ then go to state q_{acc} and move left, else go to state q_{rej} and move right.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_1 = \{w \in \Sigma^* \mid \text{second letter from the end if } a\}$.

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_1 = \{w \in \Sigma^* \mid \text{second letter from the end if } a\}$.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

2DFA: Two-way deterministic finite state automata

Examples

Let $\Sigma = \{a, b\}$ and L be a regular language.

$L_1 = \{w \in \Sigma^* \mid \text{second letter from the end if } a\}$.

$L_2 = \{w \in \Sigma^* \mid w \cdot w \in L\}$

Languages accepted by 2DFA

Example

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

Languages accepted by 2DFA

Example

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

Can a 2DFA accept $L_{a,b}$?

Languages accepted by 2DFA

Example

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

Can a 2DFA accept $L_{a,b}$?

$PAL = \{w \cdot w^R \mid w \in \{a,b\}^*\}$.

Languages accepted by 2DFA

Example

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

Can a 2DFA accept $L_{a,b}$?

$PAL = \{w \cdot w^R \mid w \in \{a,b\}^*\}$.

Can a 2DFA accept PAL?

Languages accepted by 2DFA

Example

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

Can a 2DFA accept $L_{a,b}$?

$PAL = \{w \cdot w^R \mid w \in \{a,b\}^*\}$.

Can a 2DFA accept PAL?

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof idea.

For any language accepted by a 2DFA we will define a Myhill-Nerode relation.

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof idea.

For any language accepted by a 2DFA we will define a Myhill-Nerode relation.

How should we form word equivalences?

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof idea.

For any language accepted by a 2DFA we will define a Myhill-Nerode relation.

How should we form word equivalences?

Using the behaviour of the input head for the given set of words.

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof idea.

For any language accepted by a 2DFA we will define a Myhill-Nerode relation.

How should we form word equivalences?

Using the behaviour of the input head for the given set of words.

To obtain finite index property, the equivalence should be with respect to states.

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof idea.

For any language accepted by a 2DFA we will define a Myhill-Nerode relation.

How should we form word equivalences?

Using the behaviour of the input head for the given set of words.

To obtain finite index property, the equivalence should be with respect to states.

□

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof.

Let $T_x : Q \cup \{\bowtie\} \to Q \cup \{\perp\}$, which is defined as follows:

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof.

Let $T_x : Q \cup \{\bowtie\} \to Q \cup \{\bot\}$, which is defined as follows:

$$T_x(p) := q \quad \text{if whenever } A \text{ enters } x \text{ on } p \text{ it leaves } x \text{ on } q.$$

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof.

Let $T_x : Q \cup \{\bowtie\} \to Q \cup \{\bot\}$, which is defined as follows:

$T_x(p) := q$ if whenever A enters x on p it leaves x on q.

$T_x(\bowtie) := q$ q is the state in which A emerges on x the first time.

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof.

Let $T_x : Q \cup \{\bowtie\} \to Q \cup \{\perp\}$, which is defined as follows:

$$T_x(p) := q \quad \text{if whenever } A \text{ enters } x \text{ on } p \text{ it leaves } x \text{ on } q.$$

$$T_x(\bowtie) := q \quad q \text{ is the state in which } A \text{ emerges on } x \text{ the first time.}$$

$$T_x(q) := \perp \quad \text{if } A \text{ loops on } x \text{ forever.}$$

We will say that two words, x, y are equivalent, i.e. $x \sim y$, if $T_x = T_y$.

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof.

Let $T_x : Q \cup \{\bowtie\} \to Q \cup \{\perp\}$, which is defined as follows:

$$T_x(p) := q \quad \text{if whenever } A \text{ enters } x \text{ on } p \text{ it leaves } x \text{ on } q.$$

$$T_x(\bowtie) := q \quad q \text{ is the state in which } A \text{ emerges on } x \text{ the first time.}$$

$$T_x(q) := \perp \quad \text{if } A \text{ loops on } x \text{ forever.}$$

We will say that two words, x, y are equivalent, i.e. $x \sim y$, if $T_x = T_y$.

Show that $\sim$ is a Myhill-Nerode relation.

$\square$

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof.

Let $T_x : Q \cup \{\bowtie\} \to Q \cup \{\perp\}$, which is defined as follows:

$T_x(p) := q$ if whenever A enters x on p it leaves x on q.

$T_x(\bowtie) := q$ q is the state in which A emerges on x the first time.

$T_x(q) := \perp$ if A loops on x forever.

□

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof.

Let $T_x : Q \cup \{\bowtie\} \to Q \cup \{\perp\}$, which is defined as follows:

$T_x(p) := q$ if whenever A enters x on p it leaves x on q.

$T_x(\bowtie) := q$ q is the state in which A emerges on x the first time.

$T_x(q) := \perp$ if A loops on x forever.

Total number of functions of the type

$T_x \leq (|Q| + 1)^{(|Q|+1)}$

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof.

Let $T_x : Q \cup \{\bowtie\} \rightarrow Q \cup \{\perp\}$, which is defined as follows:

$T_x(p) := q$ if whenever A enters x on p it leaves x on q.

$T_x(\bowtie) := q$ q is the state in which A emerges on x the first time.

$T_x(q) := \perp$ if A loops on x forever.

Total number of functions of the type

$T_x \leq (|Q| + 1)^{(|Q|+1)}$

$T_x = T_y \Rightarrow \forall z(xz \in F \Leftrightarrow yz \in F)$. Prove this.

Power of 2DFAs

Lemma

The class of language recognized by 2DFAs is regular.

Proof.

Let $T_x : Q \cup \{\bowtie\} \to Q \cup \{\perp\}$, which is defined as follows:

$T_x(p) := q$ if whenever A enters x on p it leaves x on q.

$T_x(\bowtie) := q$ q is the state in which A emerges on x the first time.

$T_x(q) := \perp$ if A loops on x forever.

Total number of functions of the type

$T_x \leq (|Q| + 1)^{(|Q|+1)}$

$T_x = T_y \Rightarrow \forall z(xz \in F \Leftrightarrow yz \in F)$. Prove this.

$T_x = T_y \Leftrightarrow x \equiv_A y$

□

Moving on

How to we add expressive power to DFA/NFA so that we can compute more functions?