Introduction

Formal Languages and Abstract Machines

Week 01

Baris E. Suzek, PhD

Outline

- Class Overview
- Formal Language Notations
- Mathematical Preliminaries/Notations
- Finite Automata

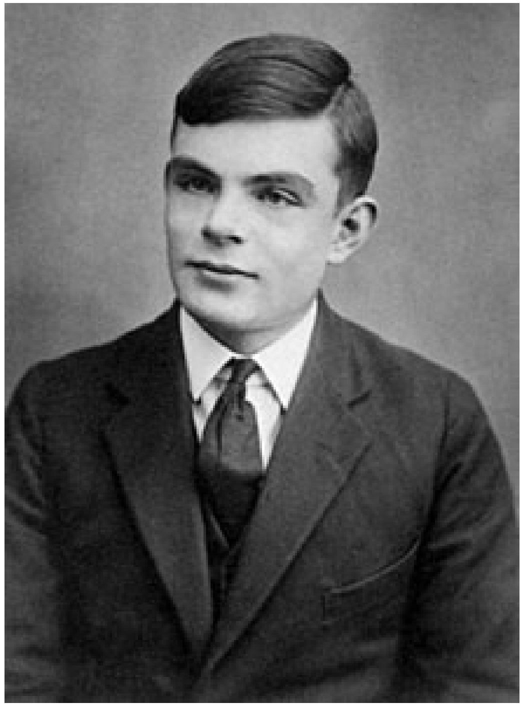
Course Info

- Grading
  - Midterm 40%
  - Final 60%
- Books
  - C.H.Elements of the Theory of Computation 2nd edition,Lewis, H.R and Papadimitriou, Prentice-Hall, 1998.
- WWW
  - We will use DYS
- Whatapp
  - https://chat.whatsapp.com/FG6v0sAgScOIUihaRTjzX8

Course Info

Week 01	24/Sep/2024	Introduction, basic concepts, mathematical preliminaries
Week 02	01/Oct/2024	Finite automata
Week 03	08/Oct/2024	Finite automata (cont'd)
Week 04	15/Oct/2024	Regular languages and grammars
Week 05	22/Oct/2024	Properties of regular languages
Week 06	29/Oct/2024	**National Holiday**
Week 07	05/Nov/2024	**MIDTERM**
Week 08	12/Nov/2024	Context-free languages
Week 09	19/Nov/2024	Context-free languages (cont'd)
Week 10	26/Nov/2024	Pushdown automata
Week 11	03/Dec/2024	Pushdown automata(cont'd)
Week 12	10/Dec/2024	Turing machines
Week 13	17/Dec/2024	Turing machines (cont'd)
Week 14	24/Dec/2024	A Hierarchy of Formal Languages and Automata
Week 15	31/Dec/2024	Computational Complexity, Final Review

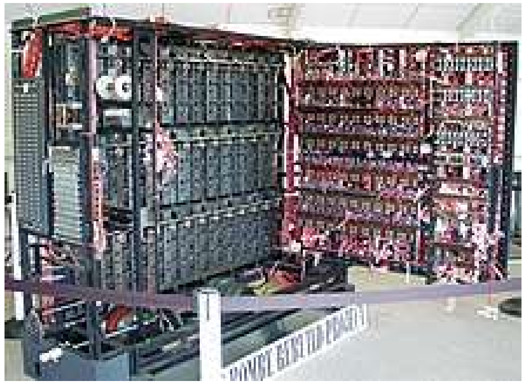
Course is about



Source: https://en.wikipedia.org/wiki/Alan_Turing

- The father of theoretical computer science and artificial intelligence
- Movie: The Imitation Game

Source: https://www.bbc.com/news/world-europe-40583718

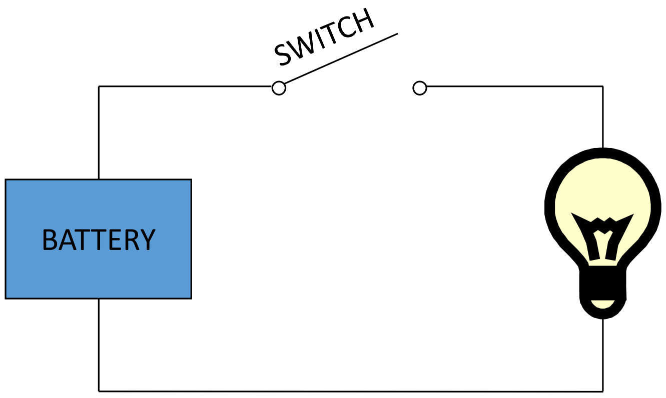


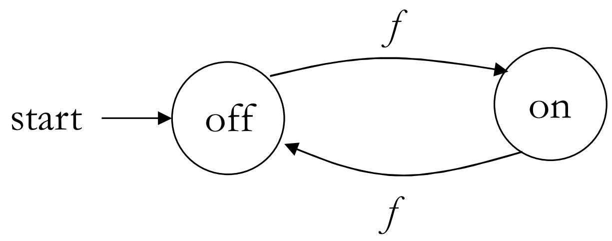
Source: https://en.wikipedia.org/wiki/Alan_Turing

Course is about

- **Abstract devices:** Simplified models of real computations
  - Has inputs/outputs
  - May have memory
  - Can make decisions on how to transform input to output
- **Formal languages:** Abstractions of the general characteristics of programming languages

An abstract device





input: switch

output: light

actions: f for "flip switch"

states: on, off

Odd number of flips => light is ON

Some device types

Finite automata	Devices with a finite amount of memory. Used to model "small" computers.
Push-down automata	Devices with infinite memory that can be accessed in a restricted way (stacks) Used to model parsers
Turing Machines	Devices with infinite memory. Used to model any computer.

Outline

- Class Overview
- Formal Language Notations
- Mathematical Preliminaries/Notations
- Finite Automata

Alphabet and String

- Alphabet: Set of letters e.g. (sigma) $\Sigma = \{a, b\}$

- String: Sequence of letters

$$a \qquad\qquad u = ab$$

$$ab \qquad\qquad v = bbbaaa$$

$$abba \qquad\qquad w = abba$$

$$baba$$

$$aaabbbaabab$$

Language and String

- A language is a set of strings
  - Language of zoo: "cat", "dog", "zebra", …
  - Defined over an alphabet:

$$\Sigma = \{a, b, c, \ldots, z\}$$

String Operations

$$w = a_1 a_2 \cdots a_n \qquad abba$$

$$v = b_1 b_2 \cdots b_m \qquad bbbaaa$$

Concatenation

$$wv = a_1 a_2 \cdots a_n b_1 b_2 \cdots b_m \qquad abbabbbaaa$$

String Operations

$$w = a_1 a_2 \cdots a_n$$

$$ababaaabbb$$

Reverse

$$w^R = a_n \cdots a_2 a_1$$

$$bbbaaababa$$

String Operations

$$w^n = \underbrace{ww\cdots w}_{n}$$

- Example: $$(abba)^2 = abbaabba$$

- Definition: $$w^0 = \lambda$$
$$(abba)^0 = \lambda$$

String Length

$$w = a_1 a_2 \cdots a_n$$

- Length: $|w| = n$

- Examples: $|abba| = 4$

$$|aa| = 2$$

$$|a| = 1$$

Empty String (lambda) λ

- A string with no letters:

$$|\lambda| = 0$$

- Observations:

$$\lambda w = w\lambda = w$$

$$\lambda abba = abba\lambda = abba$$

Substring

- Substring of string:
  - a subsequence of consecutive characters

String		Substring	
	$\underline{ab}bab$		ab
	$\underline{abba}b$		$abba$
	$ab\underline{b}ab$		b
	$a\underline{bbab}$		$bbab$

Prefix and Suffix

$$abbab$$

$$w = uv$$

prefix

suffix

- Prefixes λ Suffixes $abbab$

Prefixes	Suffixes
λ	$abbab$
a	$bbab$
ab	bab
abb	ab
$abba$	b
$abbab$	λ

The * Operation

- Σ *: the set of all possible strings from an alphabet Σ

$$\Sigma = \{a, b\}$$

$$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

The + Operation

Σ^+: the set of all possible strings from alphabet Σ except λ

$$\Sigma^+ = \Sigma^* - \lambda$$

$$\Sigma^+ = \{a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

Language with new notation

- A language is any subset of Σ^*

- Example: $\Sigma = \{a, b\}$

$$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, \ldots\}$$

- Languages: $\{\lambda\}$

$$\{a, aa, aab\}$$

$$\{\lambda, abba, baba, aa, ab, aaaaaa\}$$

Language Example

$$L = \{a^n b^n : n \geq 0\}$$

- An infinite language

$$\left.\begin{array}{l} \lambda \\ ab \\ aabb \\ aaaaabbbbb \end{array}\right\} \in L \qquad\qquad abb \notin L$$

Language Operations

- The usual set operations

$$\{a, ab, aaaa\} \cup \{bb, ab\} = \{a, ab, bb, aaaa\}$$

$$\{a, ab, aaaa\} \cap \{bb, ab\} = \{ab\}$$

$$\{a, ab, aaaa\} - \{bb, ab\} = \{a, aaaa\}$$

$$\overline{L} = \Sigma^* - L$$

- Complement:

$$\overline{\{a, ba\}} = \{\lambda, b, aa, ab, bb, aaa, \ldots\}$$

Language Operations
Reverse

$$L^R = \{w^R : w \in L\}$$

- Definition:

$$\{ab, aab, baba\}^R = \{ba, baa, abab\}$$

- Examples:

$$L = \{a^n b^n : n \geq 0\}$$

$$L^R = \{b^n a^n : n \geq 0\}$$

Language Operations
Concatenation

$$L_1 L_2 = \{xy : x \in L_1, y \in L_2\}$$

- Definition:

$$\{a, ab, ba\}\{b, aa\}$$

- Example:

$$= \{ab, aaa, abb, abaa, bab, baaa\}$$

Language Operations

$$L^n = \underbrace{LL\cdots L}_{n}$$

- Definition:

$$\{a,b\}^3 = \{a,b\}\{a,b\}\{a,b\} =$$
$$\{aaa,aab,aba,abb,baa,bab,bba,bbb\}$$

- Special case:

$$L^0 = \{\lambda\}$$

$$\{a, bba, aaa\}^0 = \{\lambda\}$$

Outline

- Class Overview
- Formal Language Notations
- Mathematical Preliminaries/Notations
- Finite Automata

Mathematical Preliminaries/Notations

- Sets
- Functions
- Relations
- Graphs
- Proof Techniques

Sets

- A set is a collection of elements

$$A = \{1, 2, 3\}$$
$$B = \{train, bus, bike\}$$

- We write

$$1 \in A$$
$$ship \notin B$$

Set Representations

C = { a, b, c, d, e, f, g, h, i, j, k }

C = { a, b, …, k } → *finite set*

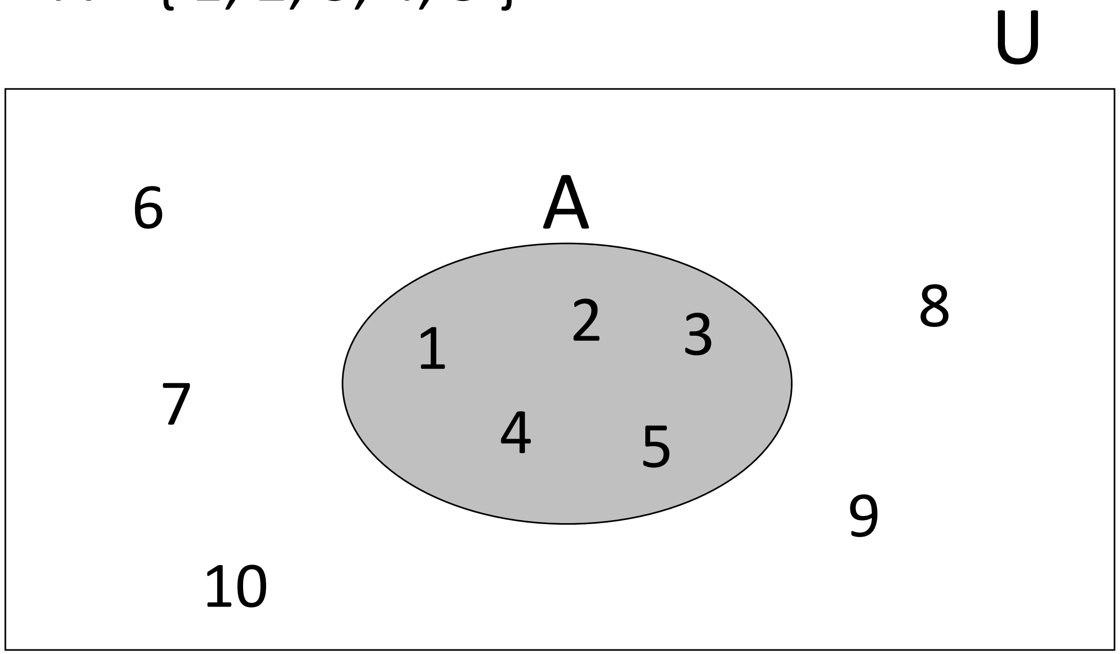
S = { 2, 4, 6, … } → *infinite set*

S = { j : j > 0, and j = 2k for some k>0 }

S = { j : j is greater than 0 and even }

Universal Set

A = { 1, 2, 3, 4, 5 }



Universal Set: all possible elements

U = { 1 , … , 10 }

Set Operations

A = { 1, 2, 3 }   B = { 2, 3, 4, 5}

- Union

  A U B = { 1, 2, 3, 4, 5 }

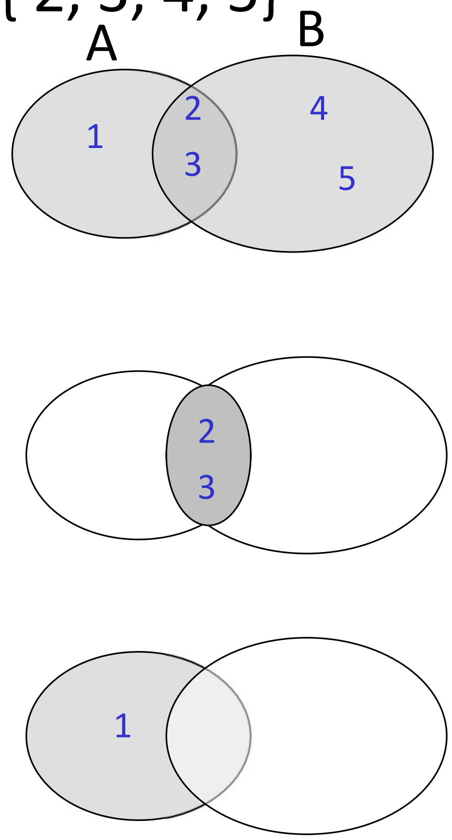
- Intersection

  A ∩ B = { 2, 3 }

- Difference

  A - B = { 1 }

  B - A = { 4, 5 }



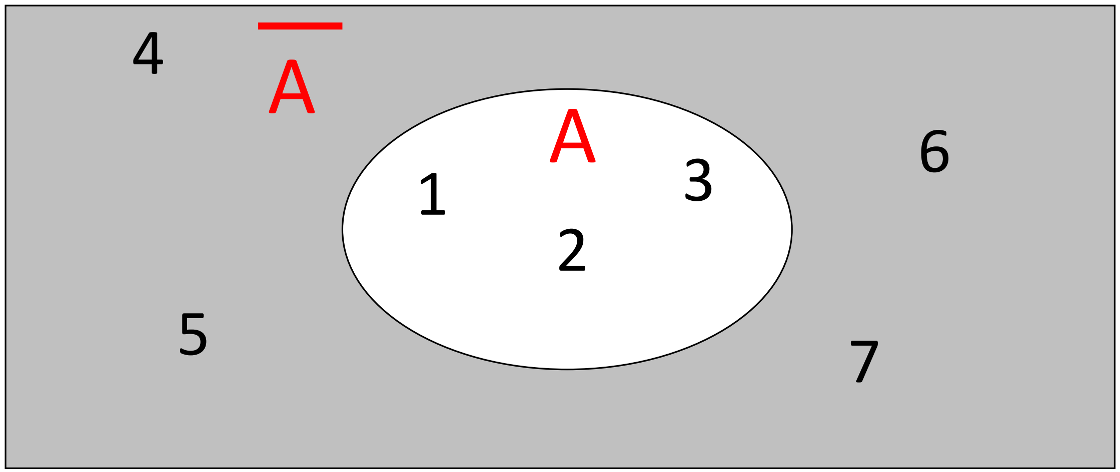
Venn diagrams

Set Operations

- Complement

Universal set = {1, ..., 7}

A = { 1, 2, 3 } ⇒ $\overline{A}$ = { 4, 5, 6, 7}



$\overline{\overline{A}} = A$

DeMorgan's Laws

$$\overline{A \cup B} = \overline{A} \cap \overline{B}$$

$$\overline{A \cap B} = \overline{A} \cup \overline{B}$$

Empty, Null Set $\emptyset = \{\ \}$

$S \cup \emptyset = S$

$S \cap \emptyset = \emptyset$

$S - \emptyset = S$

$\emptyset - S = \emptyset$

$\overline{\emptyset}$ = Universal Set

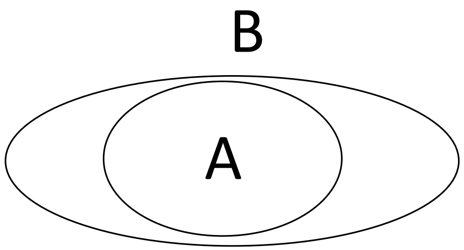
Subset

$A = \{ 1, 2, 3\}$ $B = \{ 1, 2, 3, 4, 5 \}$

Subset: $A \subseteq B$

Proper Subset: $A \subset B$

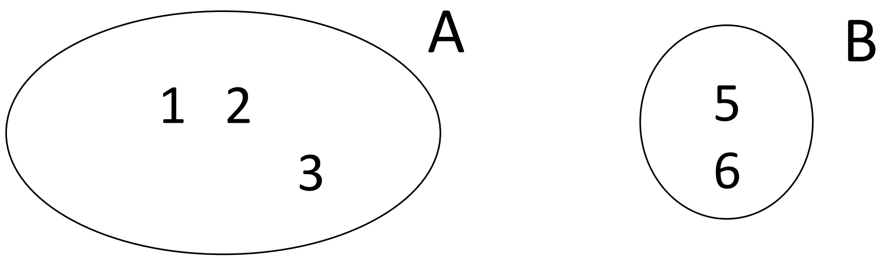
(A cannot be equal to B)



Disjoint Sets

$A = \{ 1, 2, 3 \}$ $B = \{ 5, 6\}$

$A \cap B = \emptyset$



Set Cardinality (Size)

- For finite sets

$A = \{ 2, 5, 7 \}$

$|A| = 3$

Powersets

Given:

$S = \{ a, b, c \}$

Powerset of S = the set of all the subsets of S

$2^S = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\} \}$

Observation (cardinality of powerset):

$| 2^S | = 2^{|S|} \qquad (8 = 2^3)$

Cartesian Product

$A \times B$ is the set of all ordered pairs (a, b) where $a \in A$ and $b \in B$.

$A = \{ 2, 4 \}$ $\qquad\qquad$ $B = \{ 2, 3, 5 \}$

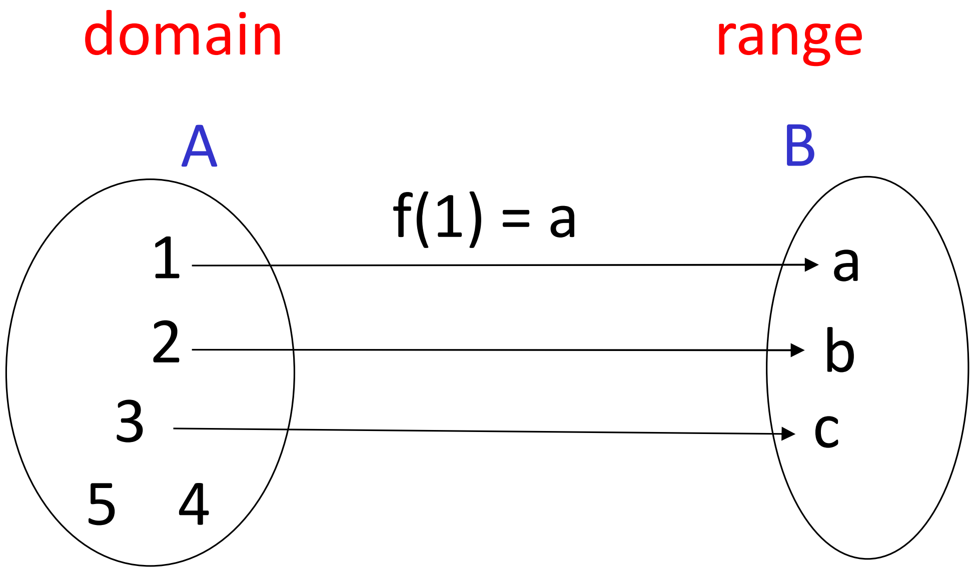
$A \times B = \{ (2, 2), (2, 3), (2, 5), (4, 2), (4, 3), (4, 5) \}$

$|A \times B| = |A| \; |B|$

Generalizes to more than two sets

$A \times B \times \ldots \times Z$

Functions



f : A -> B is a relation between members of domain and range.

If A = domain then f is a total function

otherwise f is a partial function (e.g. f(x)=1/x)

Relations

$R = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots\}$

$x_i \; R \; y_i$

e. g. if R = ‘>’: $2 > 1$, $3 > 2$, $3 > 1$

A function is a relation where the first value of every pair is unique through the set.

Equivalence Relations

A relation that is

- Reflexive: x R x
- Symmetric: x R y => y R x
- Transitive: x R y and y R z => x R z

e.g. R = ‘=‘

- x = x
- x = y => y = x
- x = y and y = z => x = z

Equivalence Classes

For equivalence relation R equivalence class of x = {y : x R y}

Example:

R = { (1, 1), (2, 2), (1, 2), (2, 1),

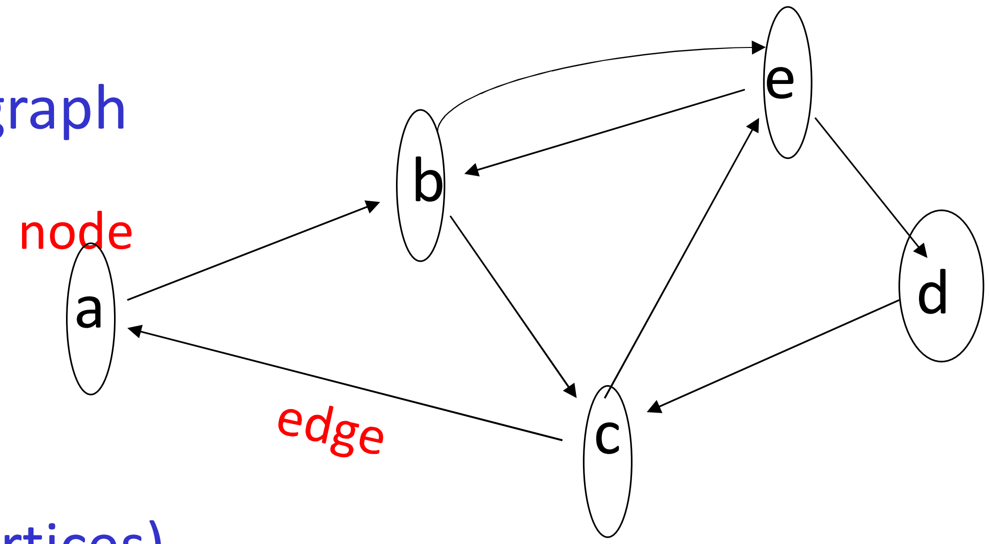
(3, 3), (4, 4), (3, 4), (4, 3) }

Equivalence class of 1 = {1, 2}

Equivalence class of 3 = {3, 4}

Graphs

A directed graph



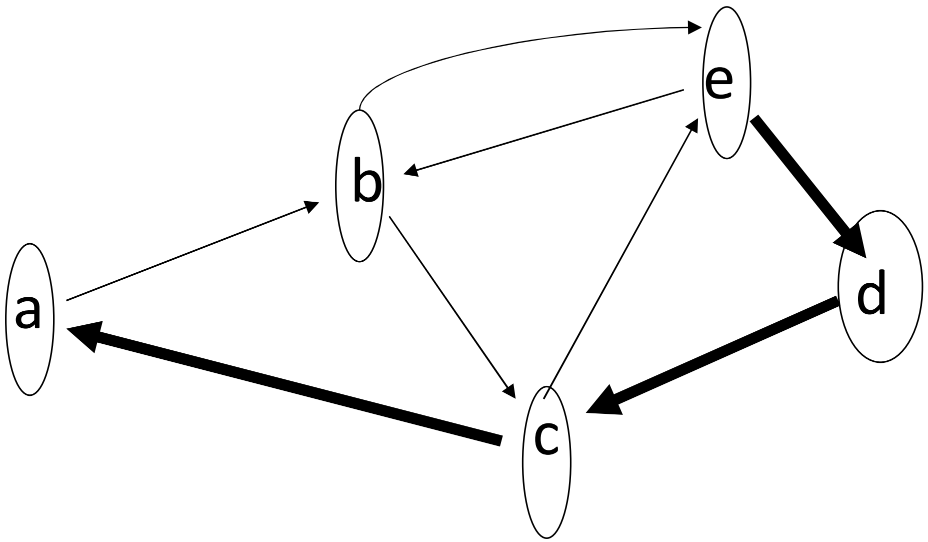
- Nodes (Vertices)

V = { a, b, c, d, e }

- Edges

E = { (a,b), (b,c), (b,e),(c,a), (c,e), (d,c), (e,b), (e,d) }

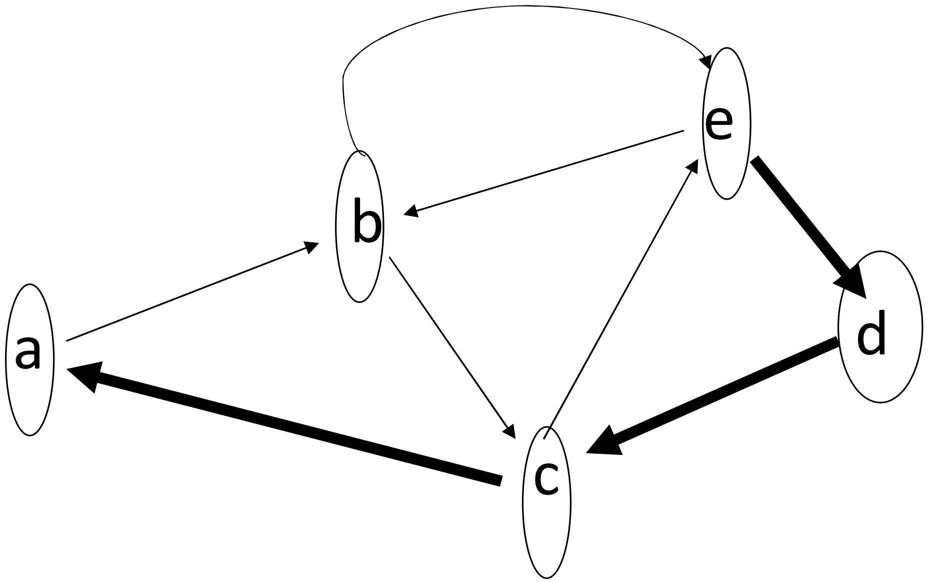
Walk



Walk is a sequence of adjacent edges

(e, d), (d, c), (c, a)

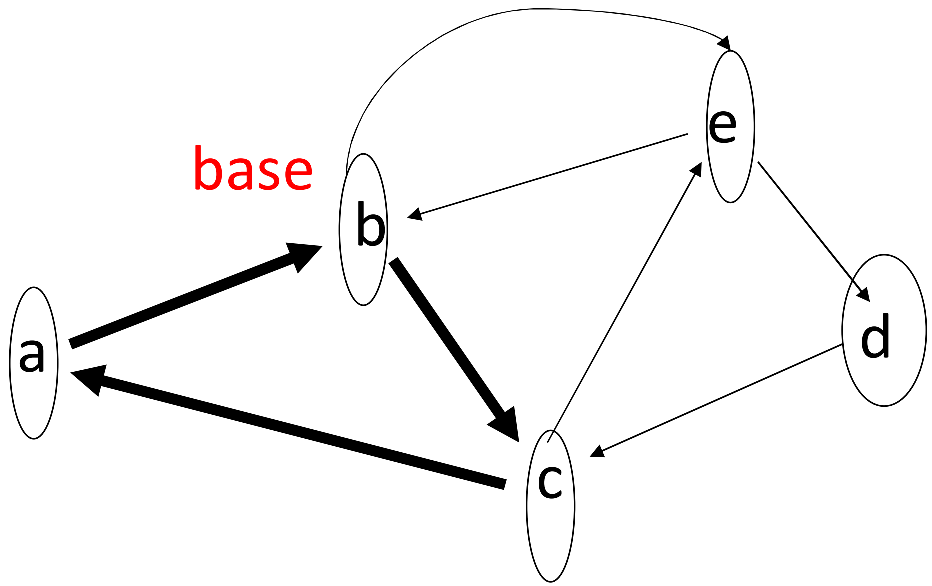
Path



Path: A walk where no edge is repeated

Simple path: No node is repeated

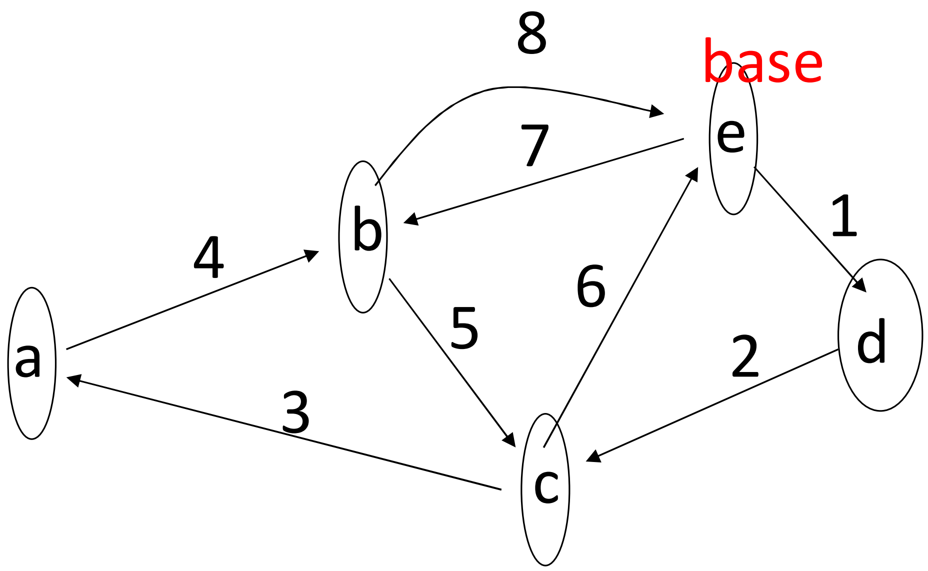
Cycle



Cycle: A walk from a node (base) to itself

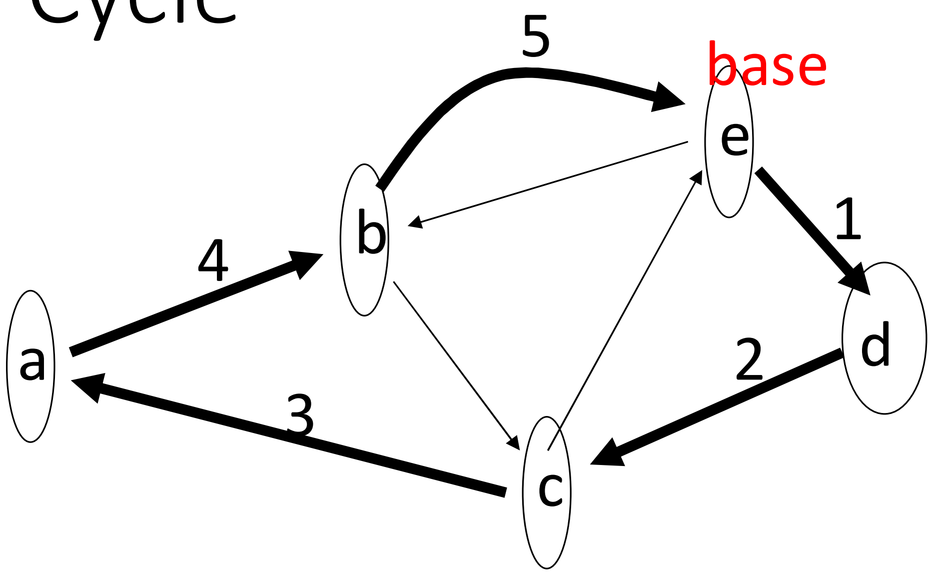
Simple cycle: Only the base node is repeated

Euler Tour



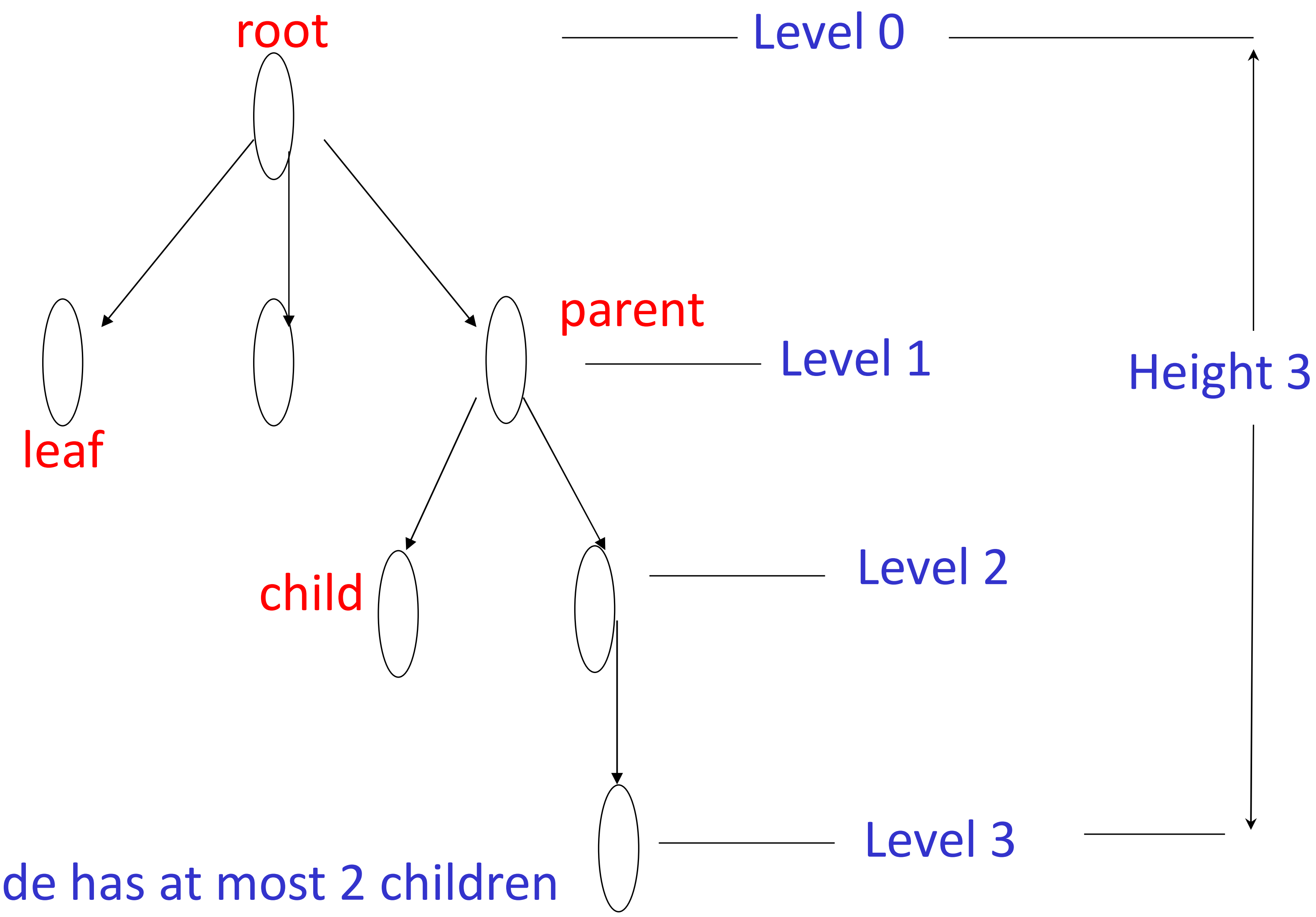
A cycle that contains each edge once

Hamiltonian Cycle



A simple cycle that contains all nodes

Trees



Trees have no cycles

Binary Trees: Each node has at most 2 children

Proof Techniques

- Proof by induction

- Proof by contradiction

Proof by Induction

We have statements P_1, P_2, P_3, …

- Inductive basis

  Find P_1, P_2, …, P_b which are true

- Inductive hypothesis

  Let’s assume P_1, P_2, …, P_k are true,

  for any k >= b

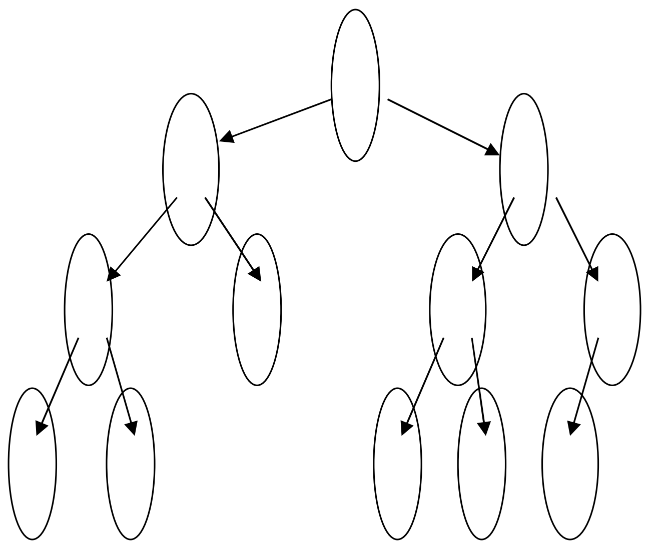
- Inductive step

  Show that P_{k+1} is true

Proof by Induction Example

<u>Theorem:</u> A binary tree of height n has at most 2^n leaves.

let $L(i)$ be the maximum number of leaves of any subtree at height i



Proof by Induction Example

We want to show: $L(i) <= 2^i$

- Inductive basis

  $L(0) = 1$ (the root node)

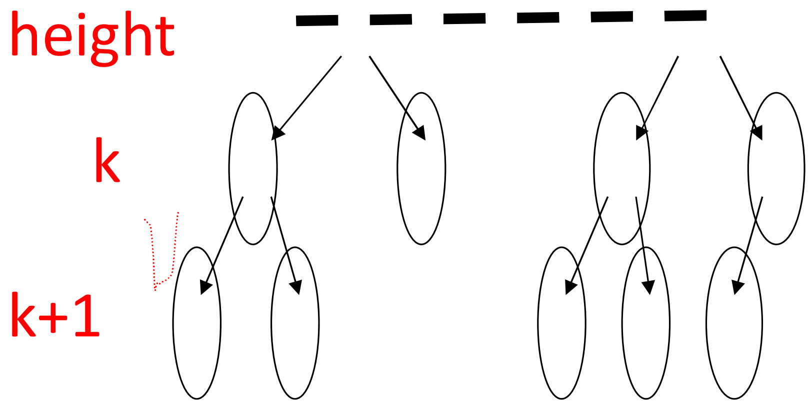
- Inductive hypothesis

  Let's assume $L(i) <= 2^i$ for all $i = 0, 1, \ldots, k$

- Induction step

  We need to show that $L(k + 1) <= 2^{k+1}$

Induction Step



$L(k) <= 2^k$

$L(k+1) <= 2 * L(k) <= 2 * 2^k = 2^{k+1}$

(at most two nodes for every leaf of level k)

Proof by Contradiction

We want to prove that a statement P is true

- we assume that P is false
- then we arrive at an incorrect conclusion
- therefore, statement P must be true

Proof by Contradiction Example

<u>Theorem:</u> $\sqrt{2}$ is not rational

<u>**Proof:**</u>

Assume by contradiction that it <u>is rational</u>

$\sqrt{2}$ = n/m

n and m have no common factors

Show that this is impossible

Proof by Contradiction Example

$\sqrt{2}$ = n/m => $2\ m^2 = n^2$

Therefore, n^2 is even => n is even

n = 2 k

$2\ m^2 = 4k^2$ => $m^2 = 2k^2$ => m is even

m = 2 p

Contradiction! since m and n have common factor 2

Outline

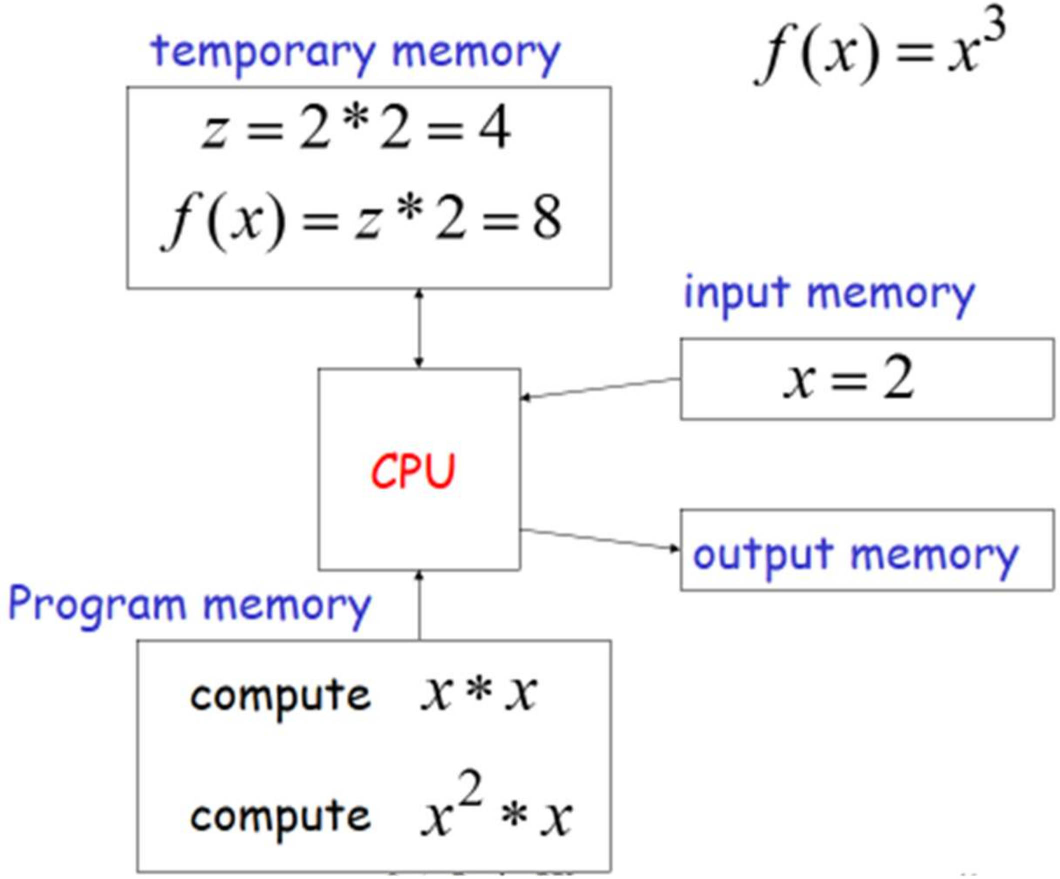
- Class Overview
- Formal Language Notations
- Mathematical Preliminaries/Notations
- Finite Automata

Finite Automata

Outline

- Introduction - Finite Automata, Types
- Finite Acceptors - Deterministic
- Regular Language
- Finite Acceptors - Nondeterministic

Computation – An Abstraction

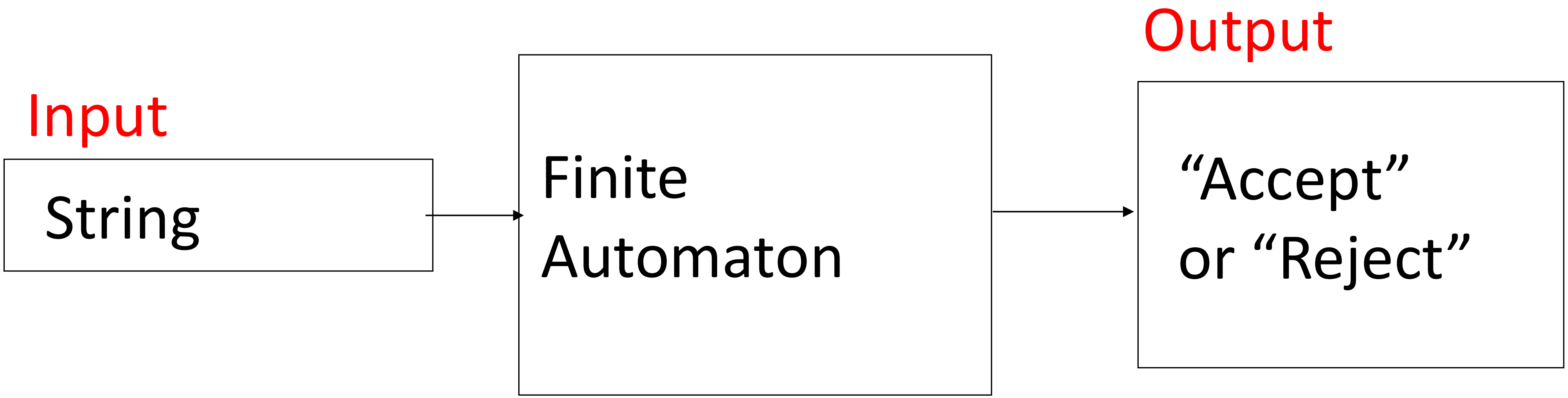


Finite Automaton

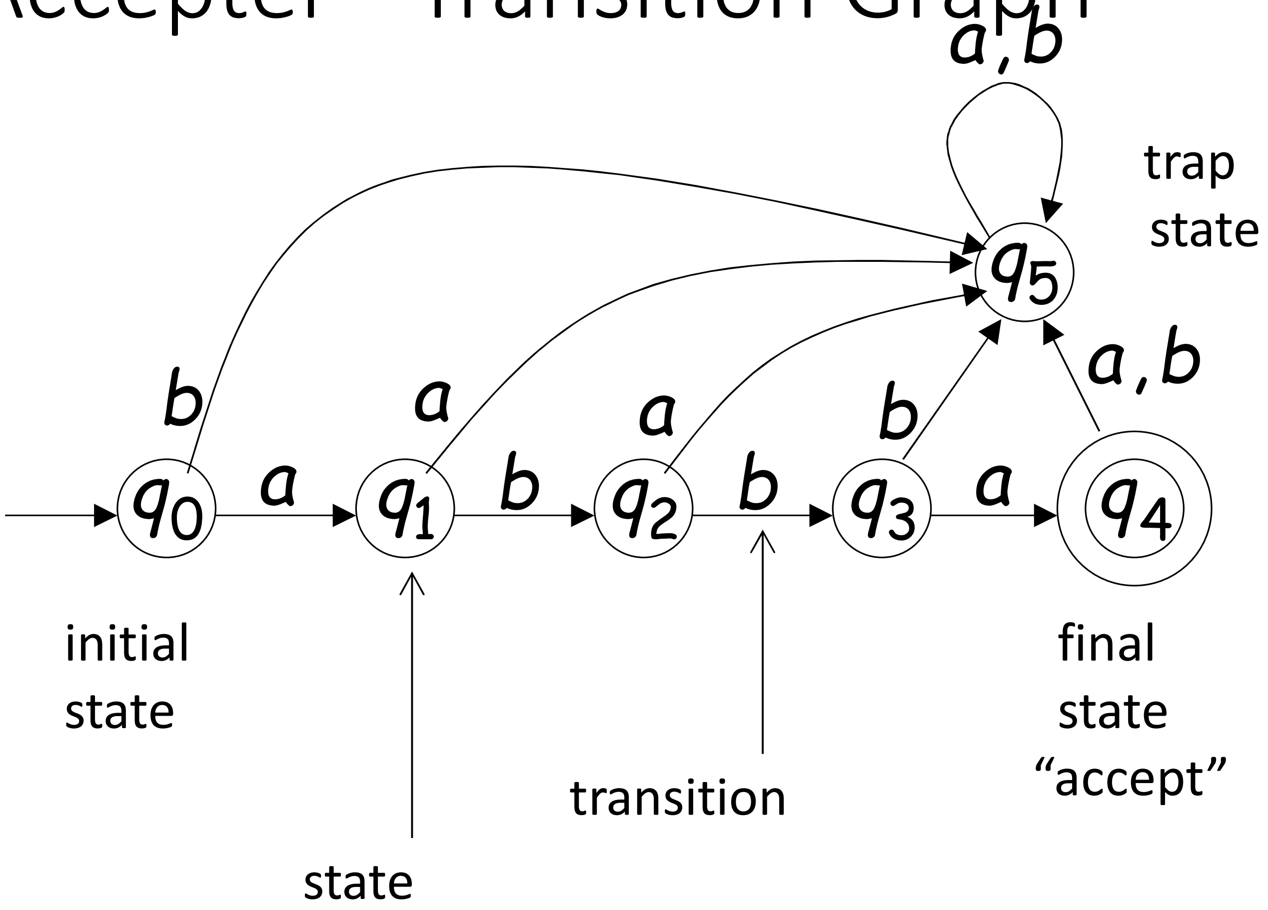
<u>No temporary memory</u>

Finite amount of information is retained through the state machine is in

Finite Automata Type: Finite Acceptor

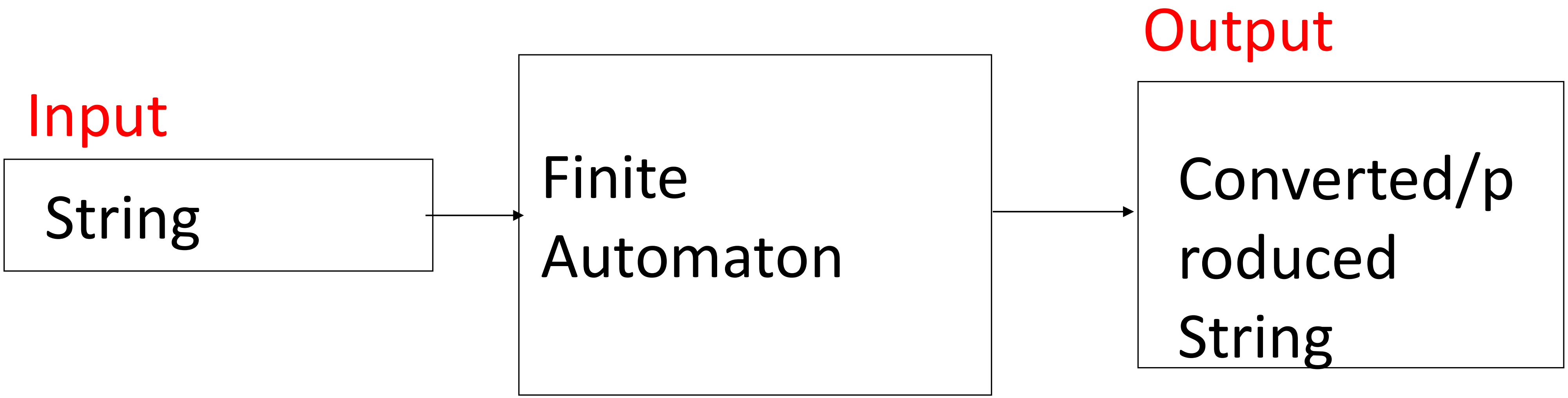


Finite Accepter - Transition Graph



Deterministic FA: produces a unique run of the automaton for each input string

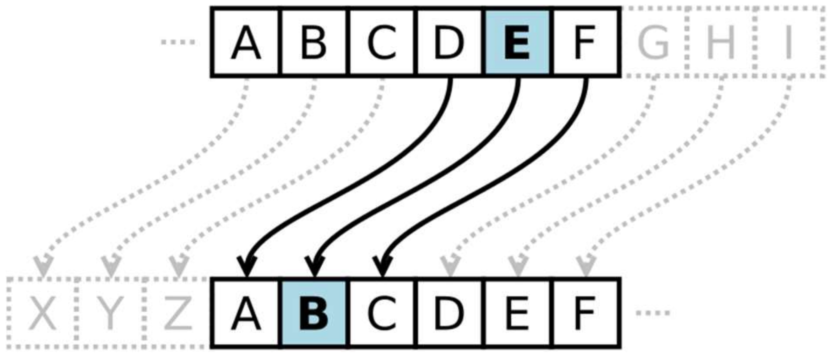
Finite Automata Type: Transducer



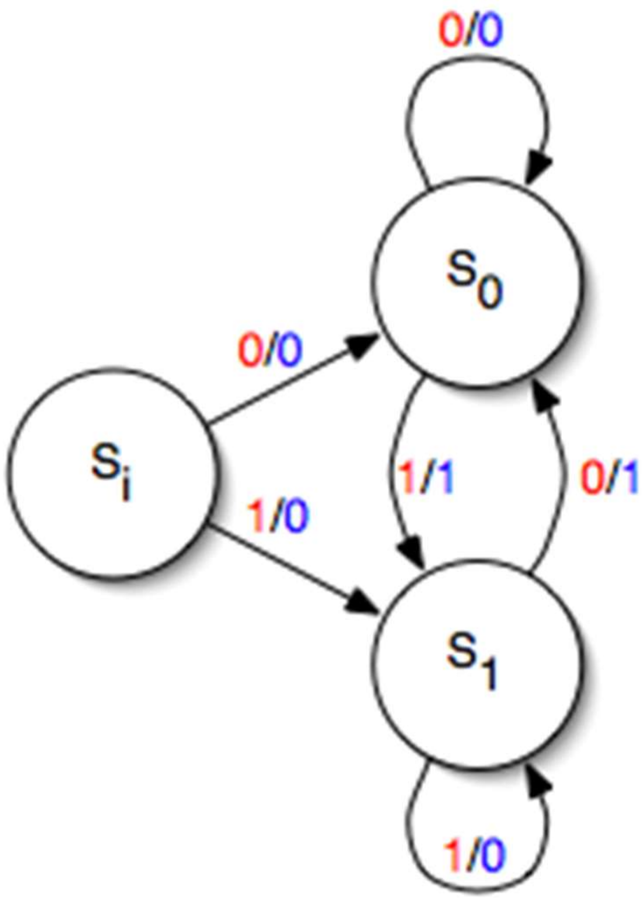
Transducer - Transition Graph

Mealy Machine:
Transitions have outputs.

Can model simple cipher
like shift cipher



From Wiki for Caesar Cipher



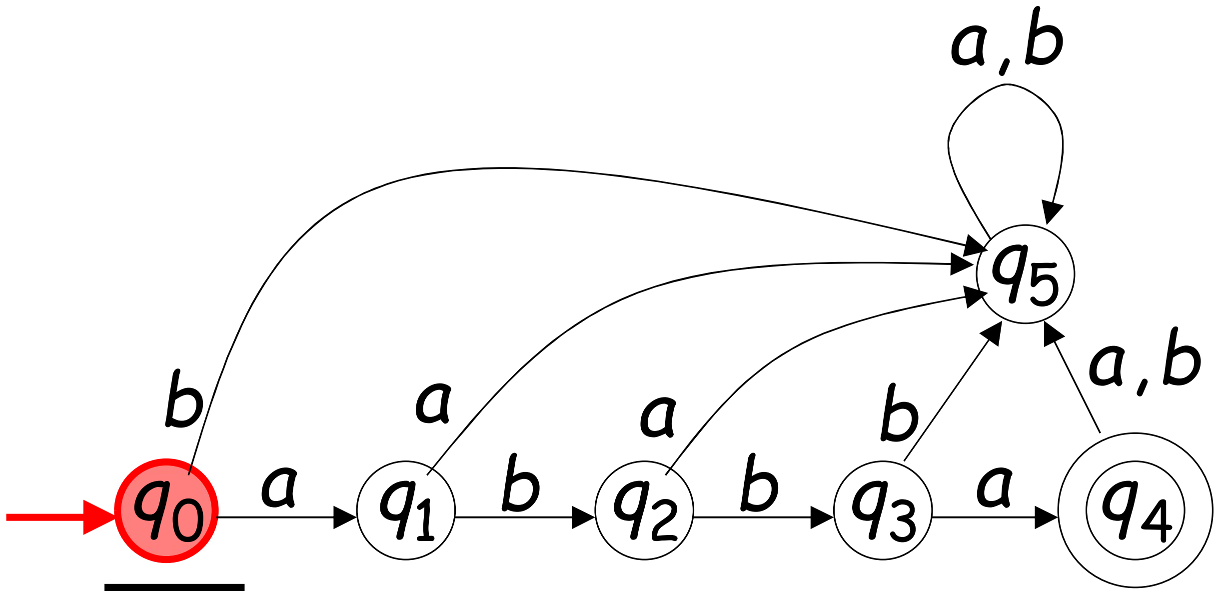
Outline

- Introduction - Finite Automata, Types
- Finite Acceptors - Deterministic
- Regular Language
- Finite Acceptors - Nondeterministic

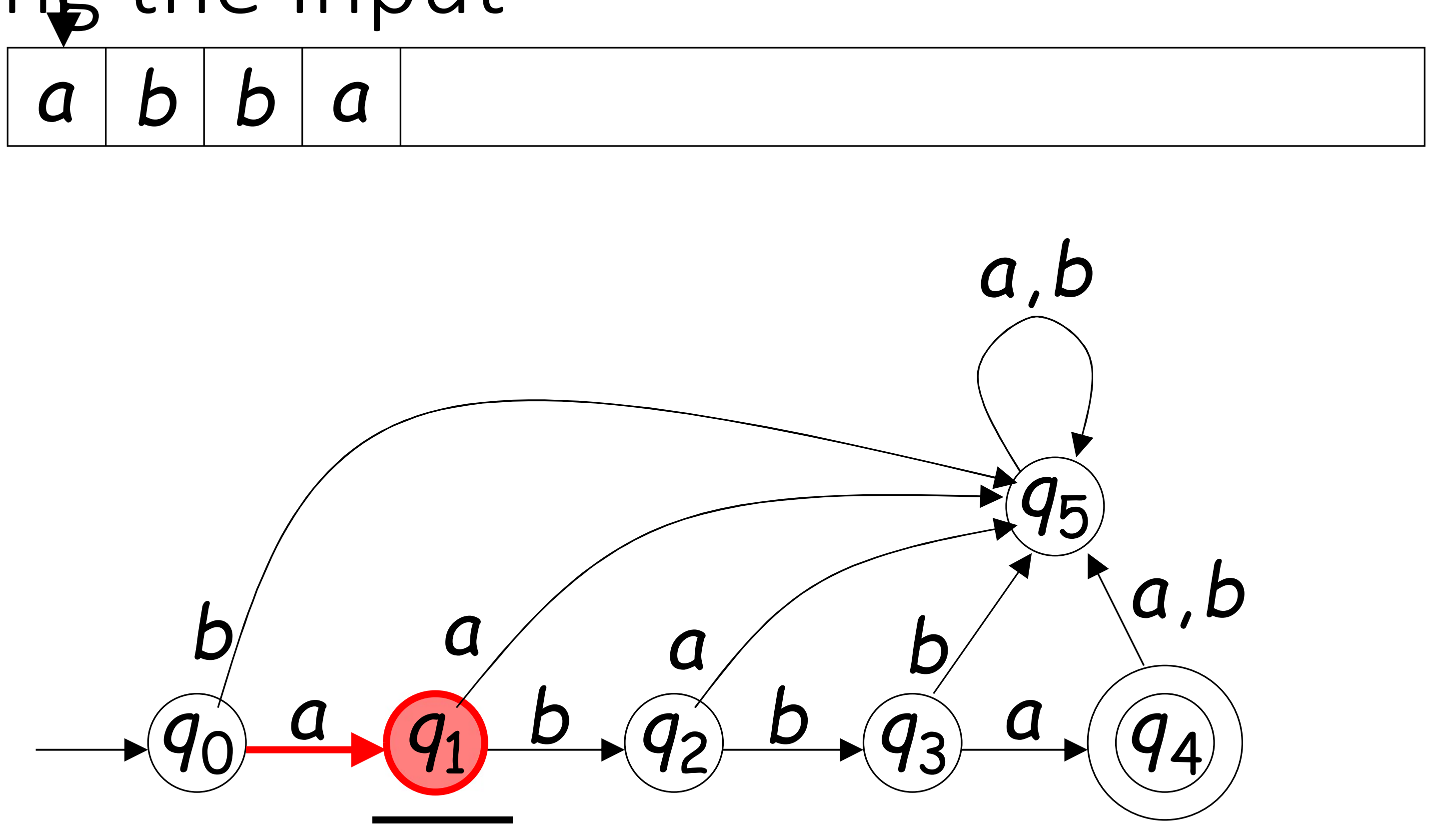
Initial Configuration

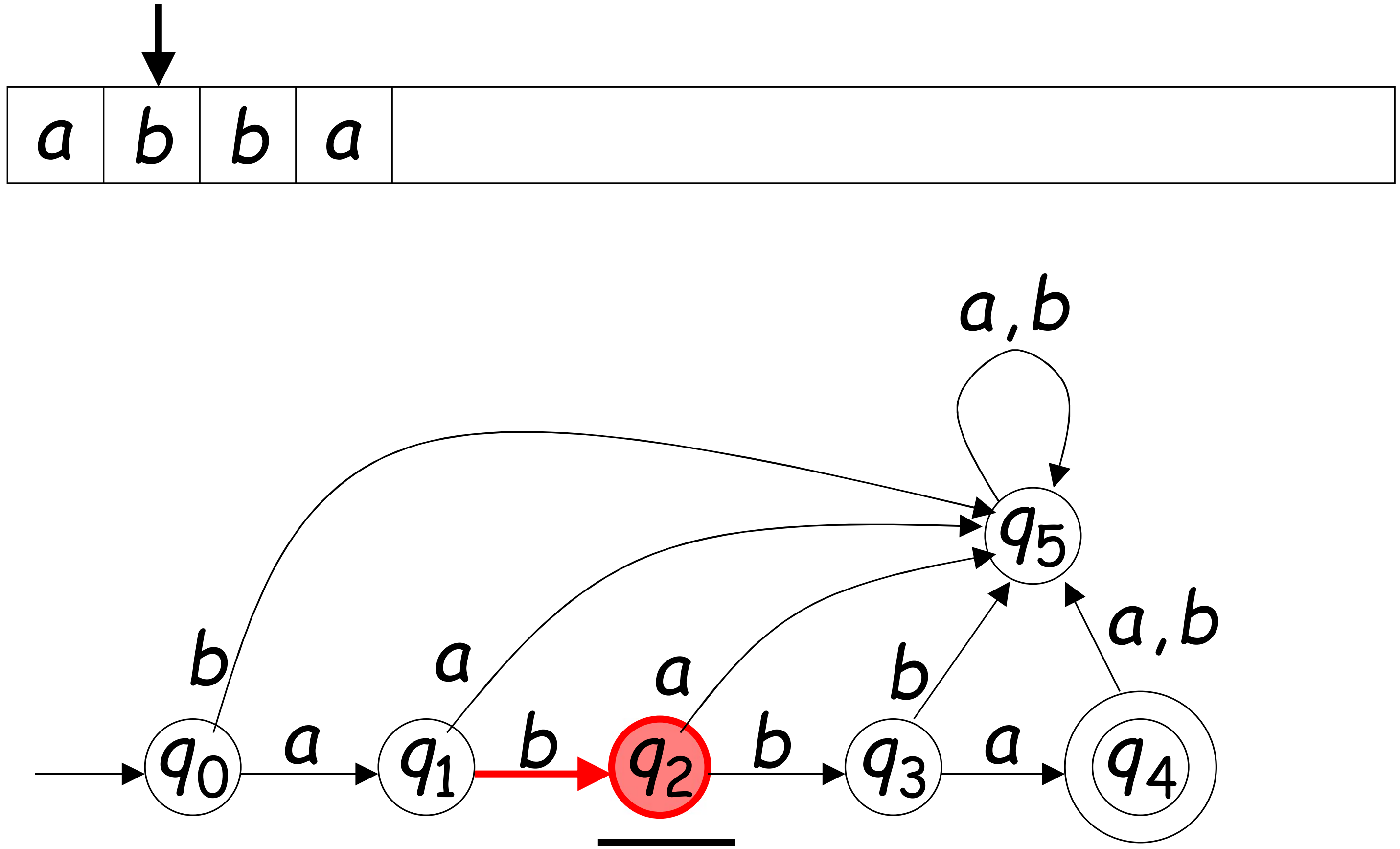
Input String

a	b	b	a	



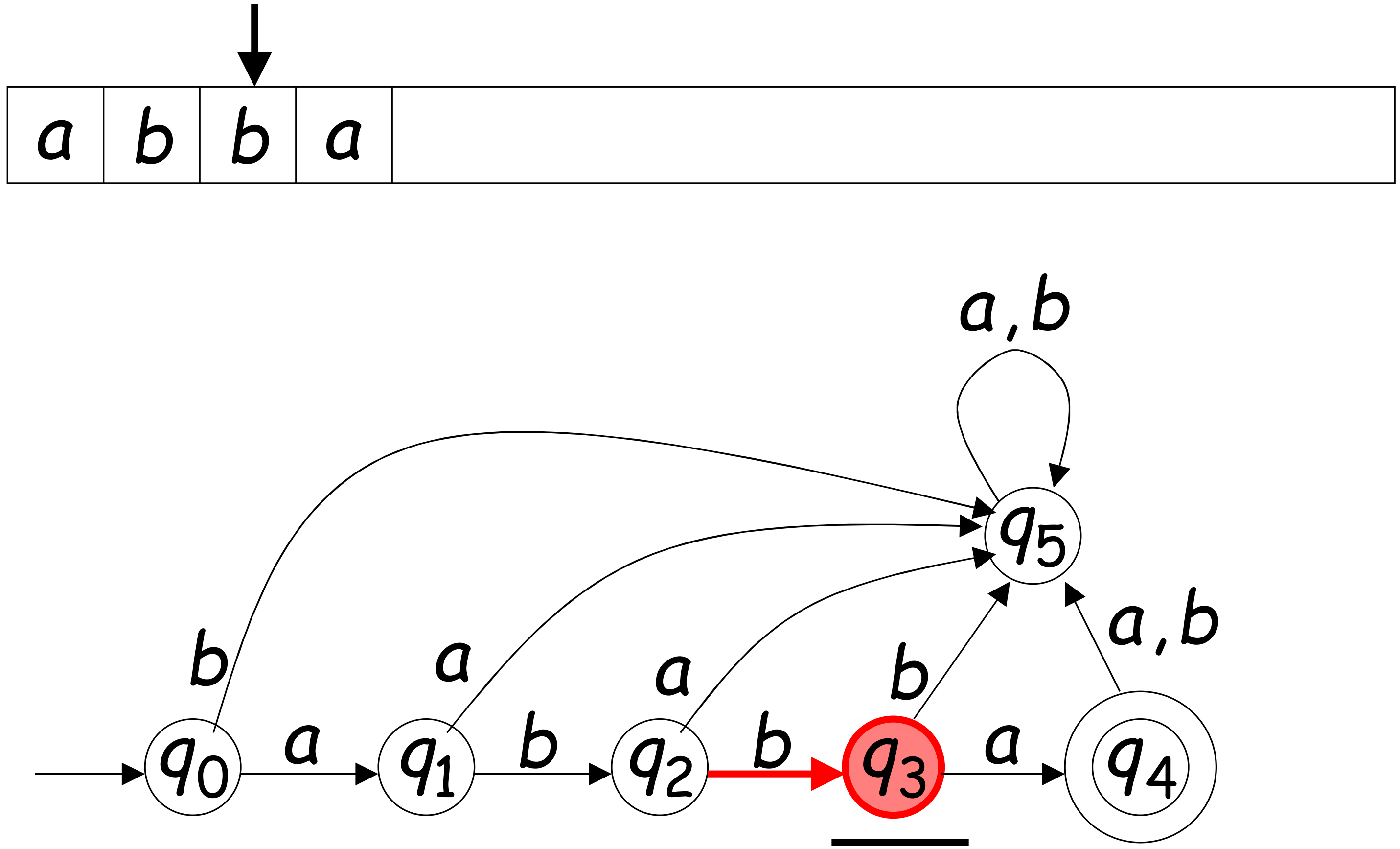
Reading the Input



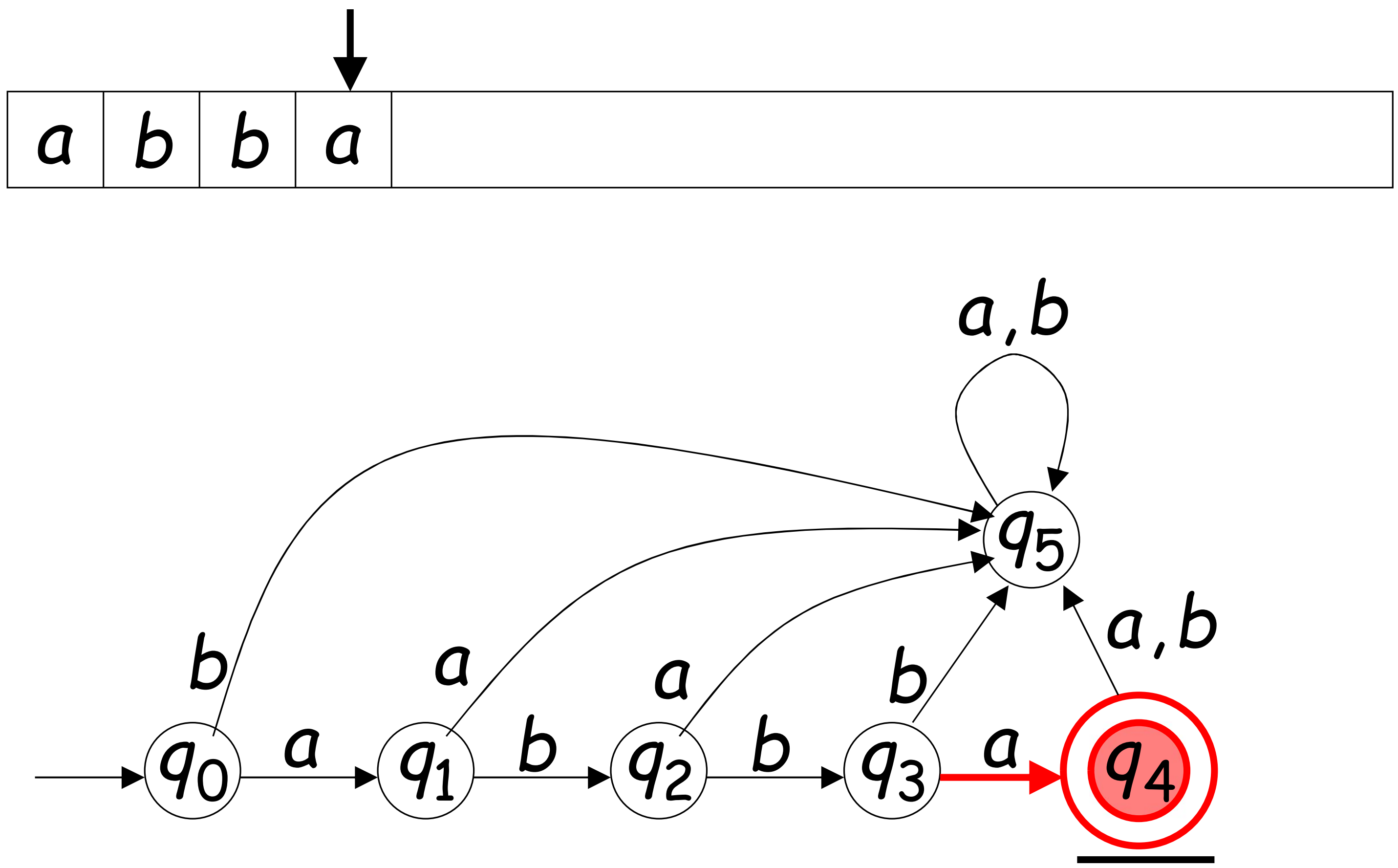


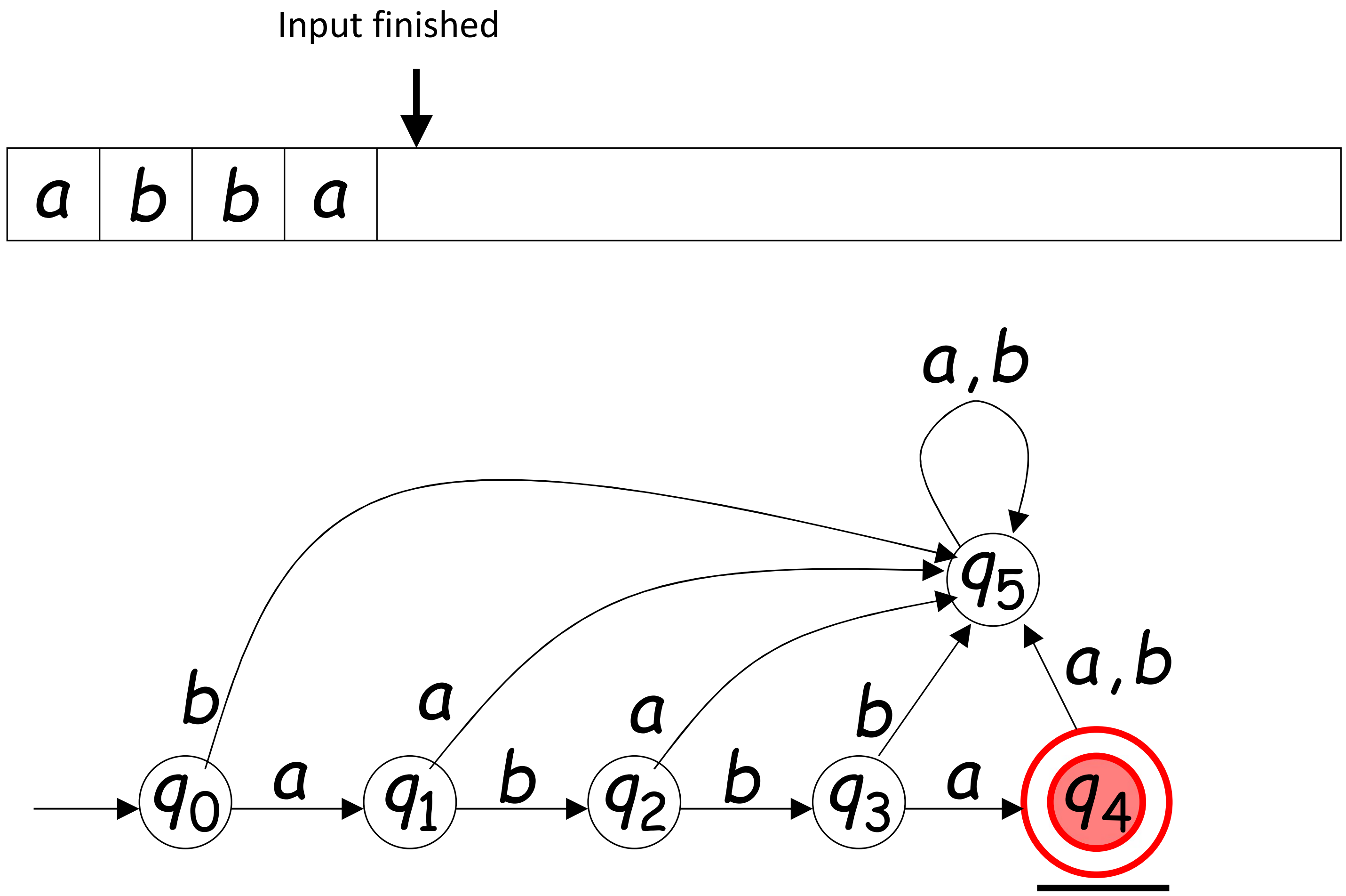
a
b
b
a
a,b
b
a
a
b
a,b
q_0
a
q_1
b
q_2
b
q_3
a
q_4
q_5

•


a
b
b
a
a,b
b
a
a
b
a
b
a,b
a
q0
q1
q2
q3
q4
q5

•

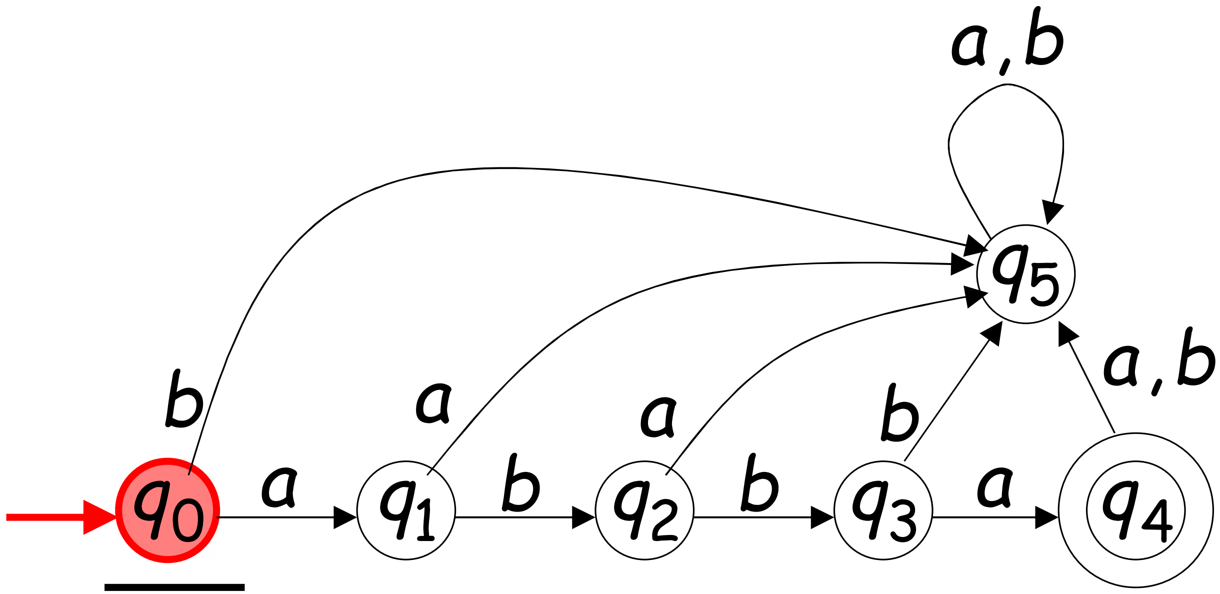

a
b
b
a
a, b
b
a
a
b
a, b
a
b
b
a
q_0
q_1
q_2
q_3
q_4
q_5

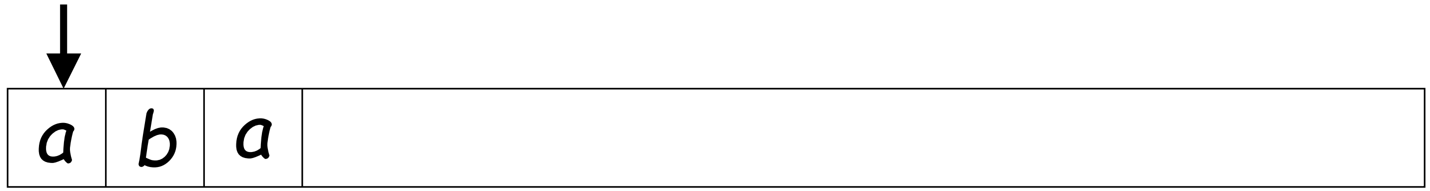


Output: “accept”

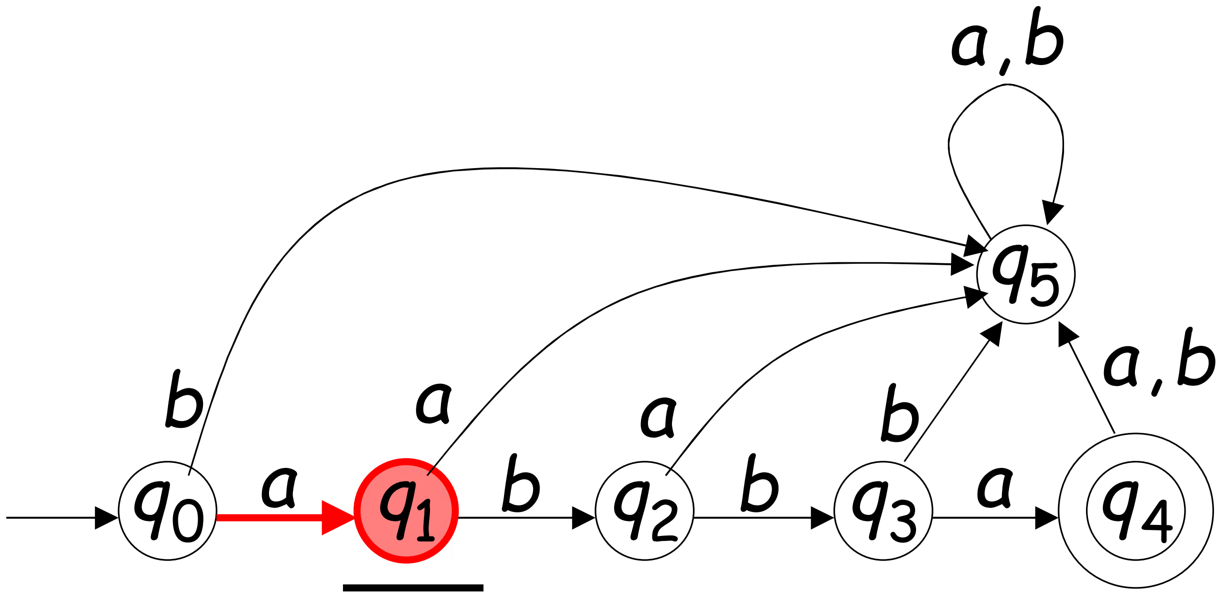
Rejection

•



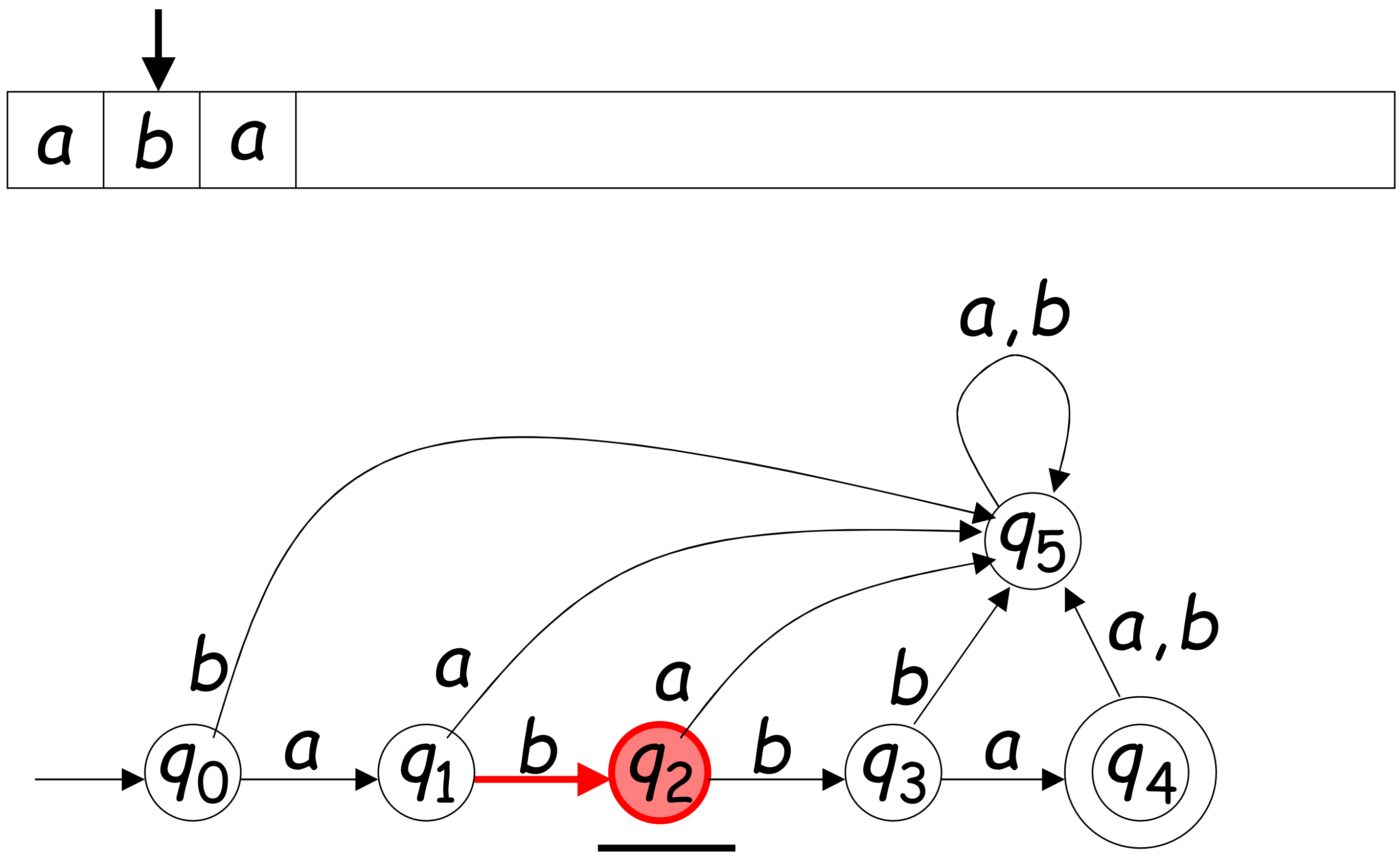
- 

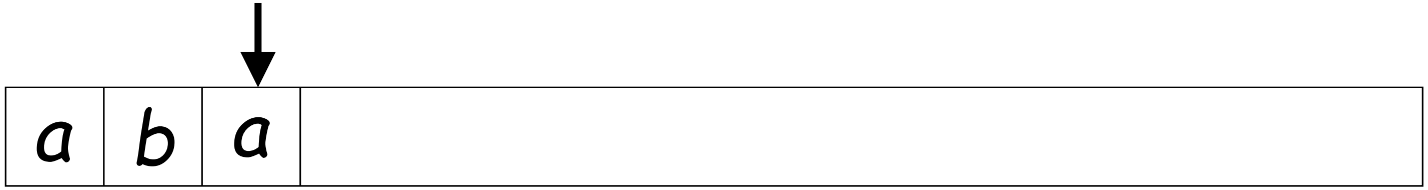
a
b
a



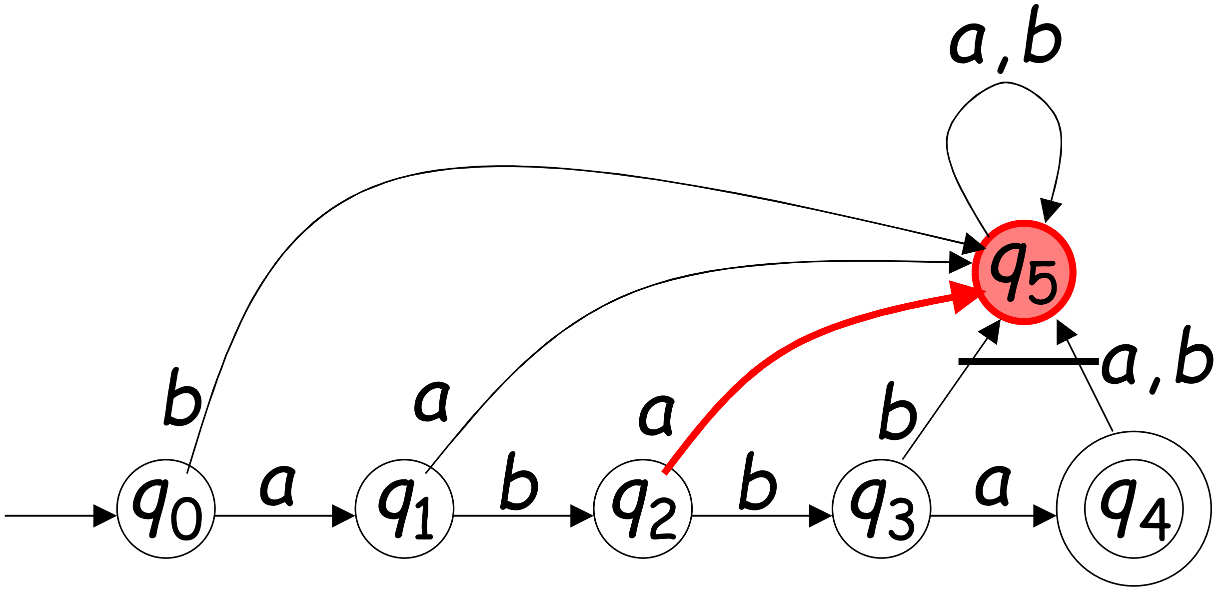
a,b
b
a
a
b
a,b
q_0
a
q_1
b
q_2
b
q_3
a
q_4
q_5

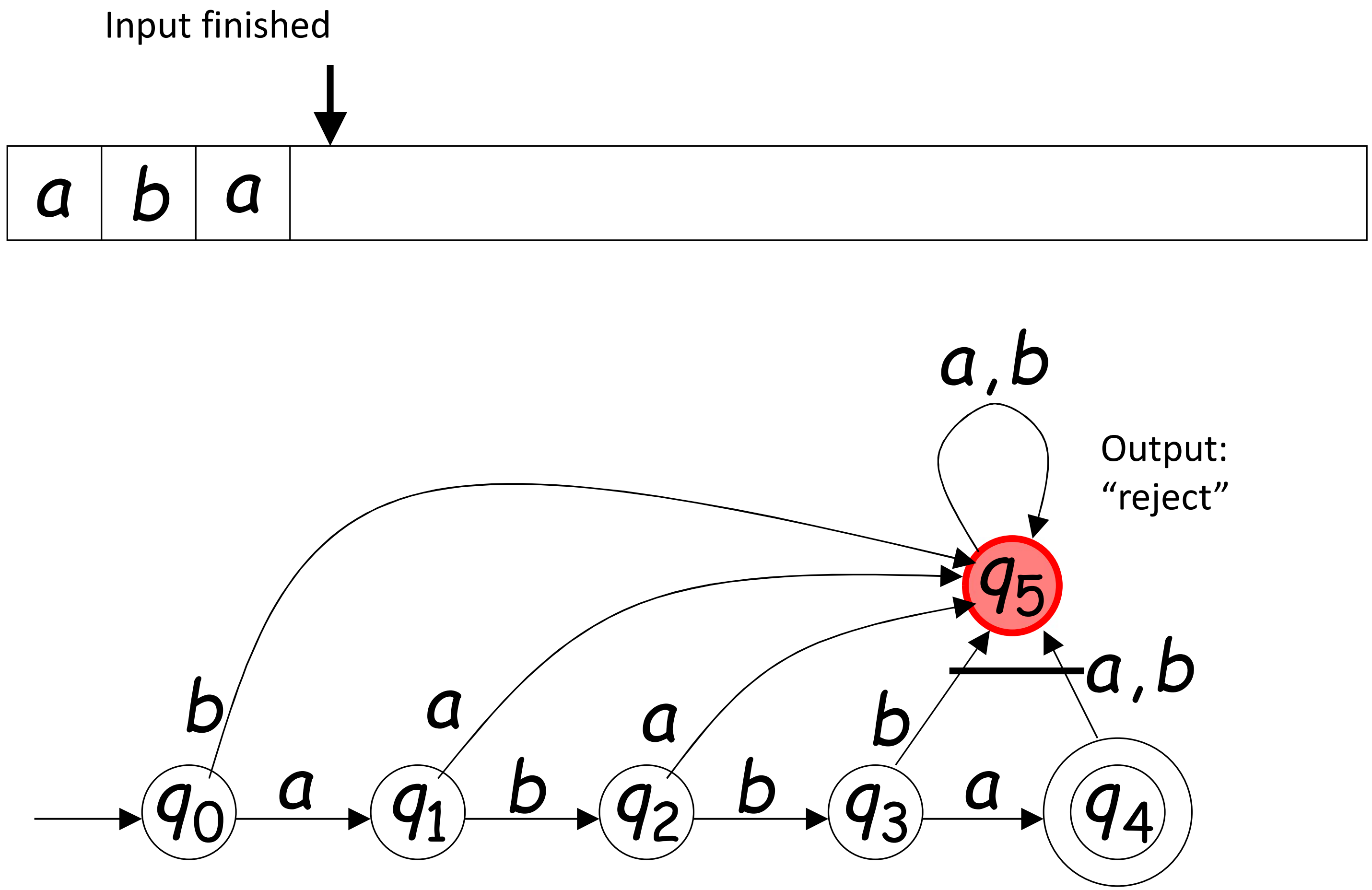
•


a
b
a
a,b
q5
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4


a
b
a

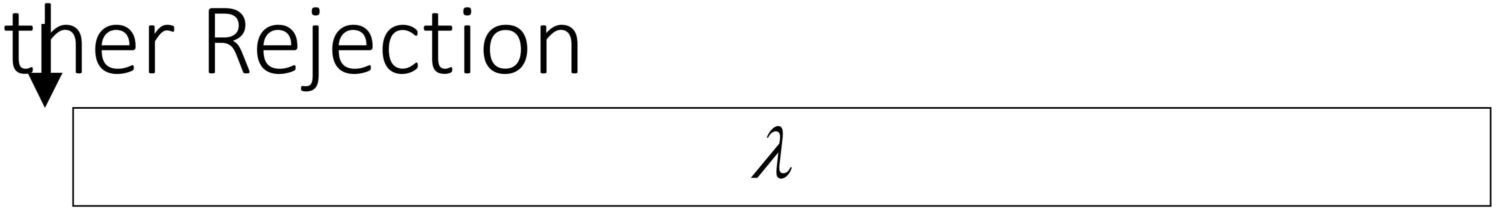
-


a,b
b
a
a
b
a,b
q_0
a
q_1
b
q_2
b
q_3
a
q_4
q_5

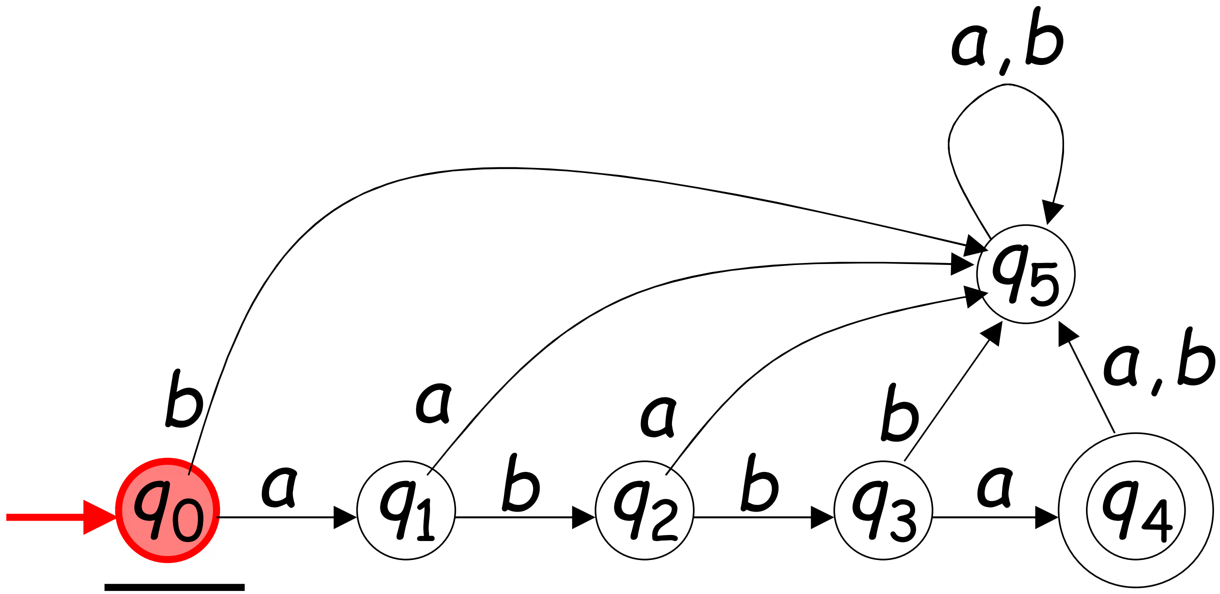


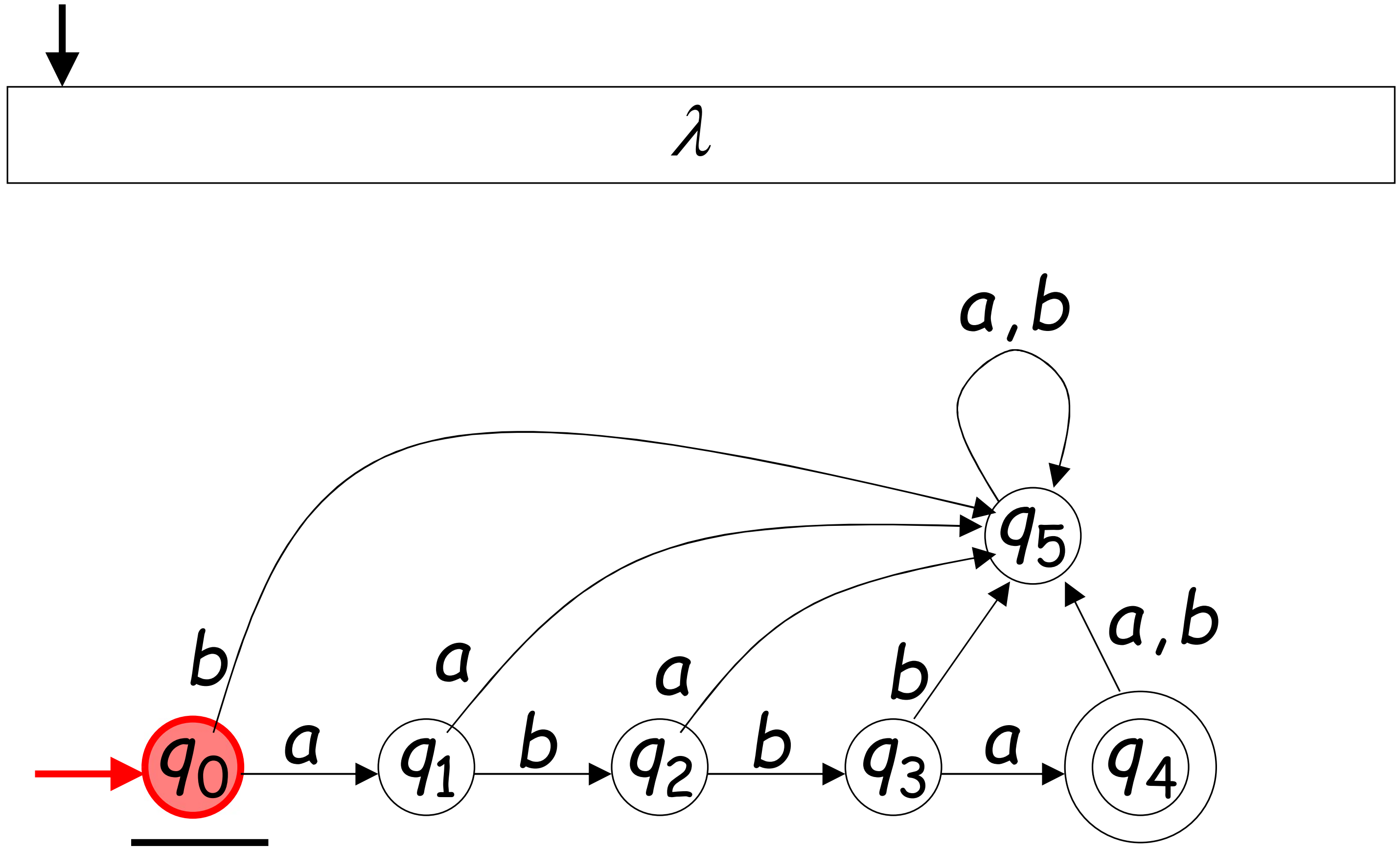
Input finished
a
b
a
a,b
Output:
“reject”
b
a
a
b
a,b
q_0
a
q_1
b
q_2
b
q_3
a
q_4
q_5

Another Rejection



•





Output:
"reject"