CS 6041
Theory of Computation

Review 1

Kun Suo

Computer Science, Kennesaw State University

https://kevinsuo.github.io/

Types of proof

- Proof by construction
  - Create a formula, graph, automata, Turing machine …
- Proof by contradiction
  - $\sqrt{2}$ is irrational
- Proof by induction
  - P(1) is true
  - Suppose P(n) is true, prove P(n+1) based on P(n)

Definition of finite automaton

- Finite automaton is a 5-tuple $M=(Q,\Sigma,\delta,q_0,F)$
  - Q: finite set called states
  - Σ: finite set called the alphabet
  - δ: $Q \times \Sigma \rightarrow Q$, transition function
  - $q_0 \in Q$: start state
  - $F \subseteq Q$: accept states

- Language on M: $L(M) = \{w \in \Sigma^* \mid \delta(q_0,w) \in F\}$

DFA example: definition - -> graph/language

- $M_2=(\{q_1,q_2\},\{0,1\},\delta, q_1,\{q_2\})$

Can anyone draw the DFA?

δ

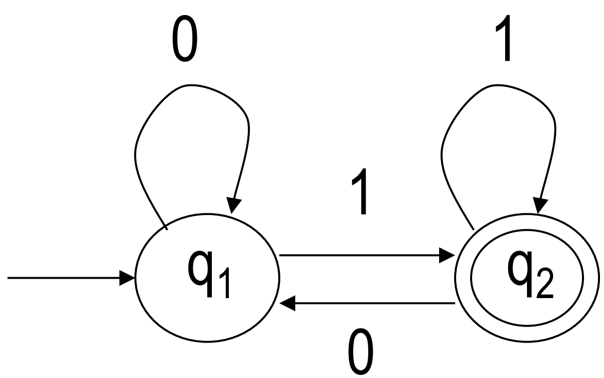
	0	1
q_1	q_1	q_2
q_2	q_1	q_2

What is the language of M2

DFA example: definition - -> graph/language

- $M_2=(\{q_1,q_2\},\{0,1\},\delta, q_1,\{q_2\})$

Can anyone draw the DFA?



δ

	0	1
q_1	q_1	q_2
q_2	q_1	q_2

What is the language of M2

$L(M_2) = \{ w \mid w \text{ ends with 1s} \}$

DFA example: definition - -> graph/language

- $M_3=(\{q_1,q_2\},\{0,1\},\delta, q_1,\{q_1\})$

Can anyone draw the DFA?

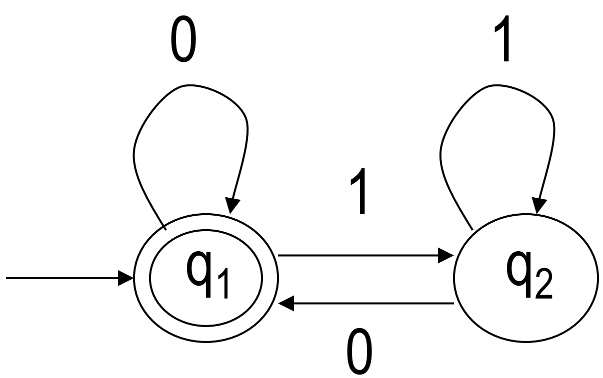
	0	1
q_1	q_1	q_2
q_2	q_1	q_2

What is the language of M3

DFA example: definition - -> graph/language

- $M_3=(\{q_1,q_2\},\{0,1\},\delta, q_1,\{q_1\})$

Can anyone draw the DFA?



	0	1
q_1	q_1	q_2
q_2	q_1	q_2

What is the language of M3

$L(M_3) = \{ w \mid w = \varepsilon \text{ or } w \text{ ends with 0s} \}$

DFA example: definition - -> graph/language

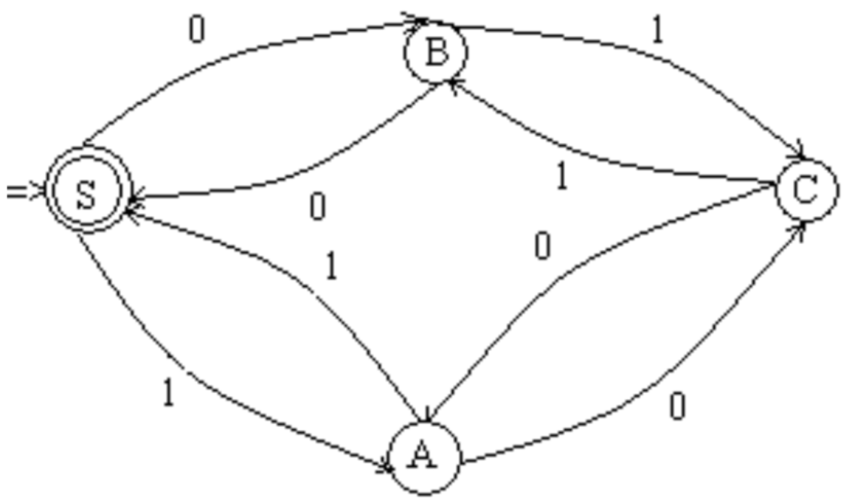
- $M = (\{S,A,B,C\},\{0,1\},f, S,\{S\})$
  - $f(S,0)=B$, $f(B,0)=S$
  - $f(S,1)=A$, $f(B,1)=C$
  - $f(A,0)=C$, $f(C,0)=A$
  - $f(A,1)=S$, $f(C,1)=B$

Can anyone draw the DFA?

DFA example: definition - -> graph/language

- M =({S,A,B,C},{0,1},f, S,{S})
  - f(S,0)=B, f(B,0)=S
  - f(S,1)=A, f(B,1)=C
  - f(A,0)=C, f(C,0)=A
  - f(A,1)=S, f(C,1)=B

Can anyone draw the DFA?

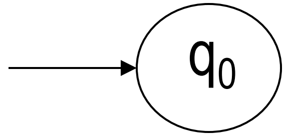


DFA example: definition - -> graph/language

- Ø
- M =$(\{q_0\},\{\},f, q_0,\{\})$

DFA example: definition - -> graph/language

- Ø
- M =({q_0},{},f, q_0,{})

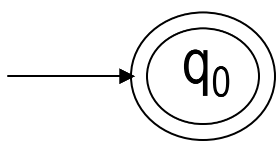


DFA example: definition - -> graph/language

- $\{\varepsilon\}$
- M =$(\{q_0\},\{\},f, q_0,\{q_0\})$

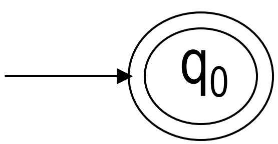
DFA example: definition - -> graph/language

- $\{\varepsilon\}$
- $M = (\{q_0\}, \{\}, f, q_0, \{q_0\})$

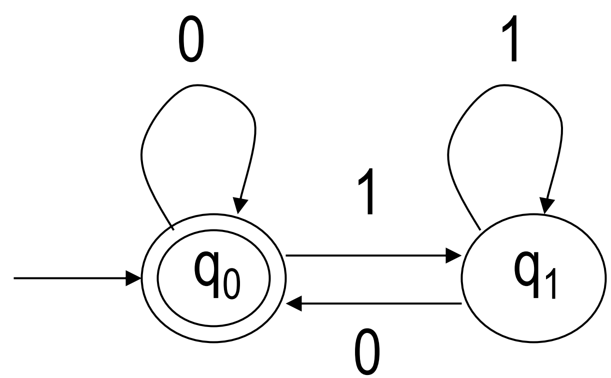


DFA example: definition - -> graph/language

- $\{\varepsilon\}$
- $M = (\{q_0\}, \{\}, f, q_0, \{q_0\})$



subgraph

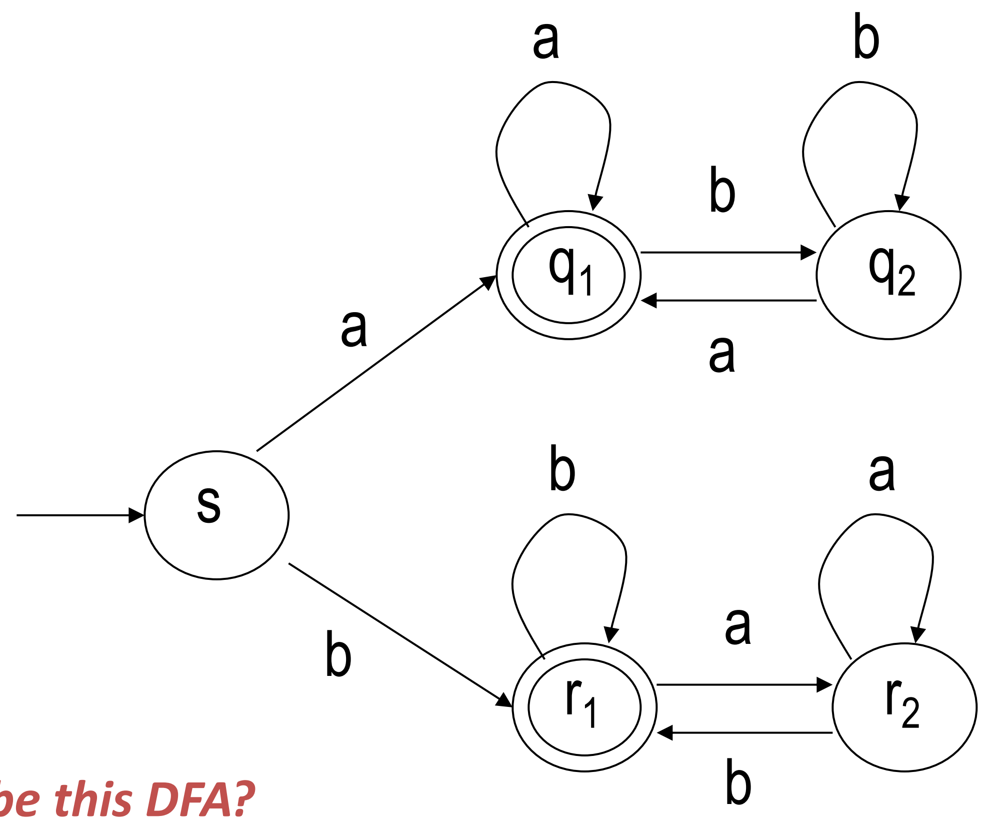


$\{\varepsilon\}$

subset

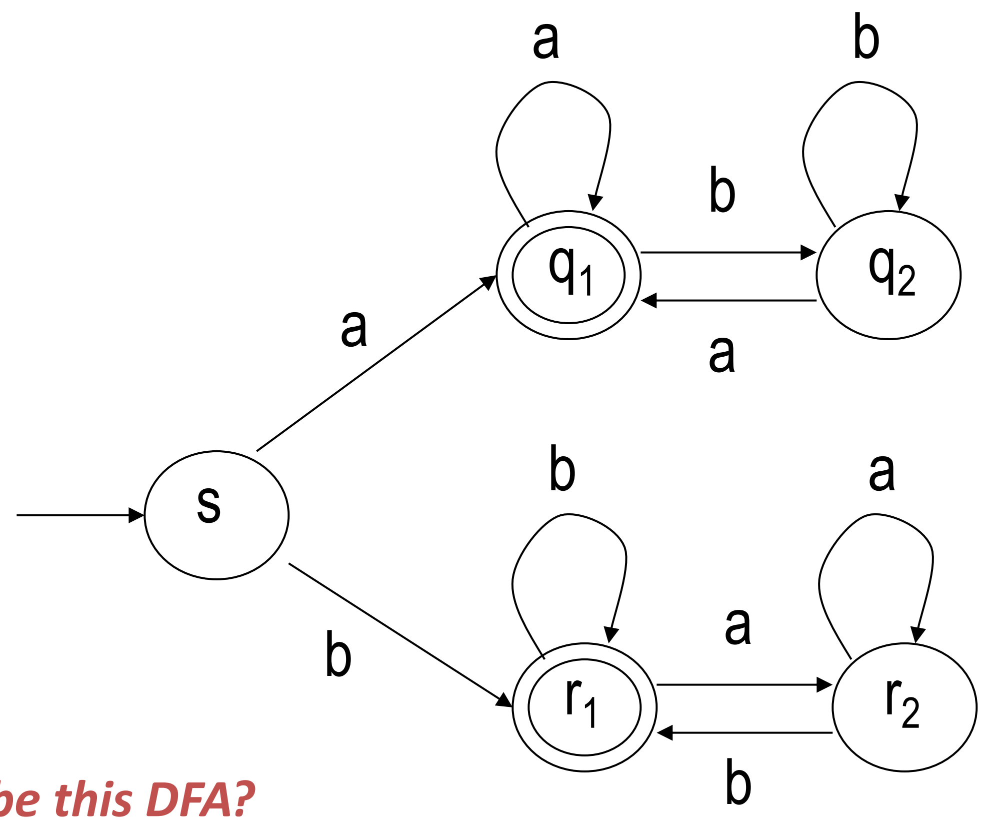
$L(M_3) = \{ w \mid w = \varepsilon \text{ or } w \text{ ends with 0s} \}$

DFA example: graph- -> language



How to describe this DFA?

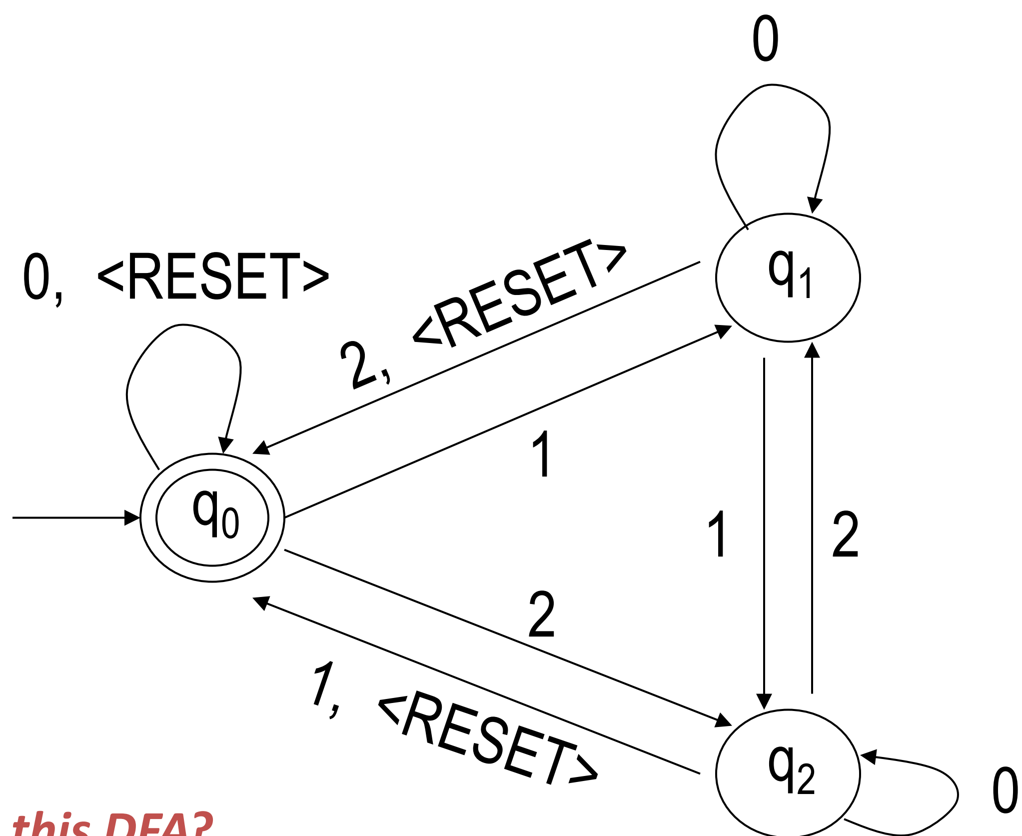
DFA example: graph- -> language



How to describe this DFA?

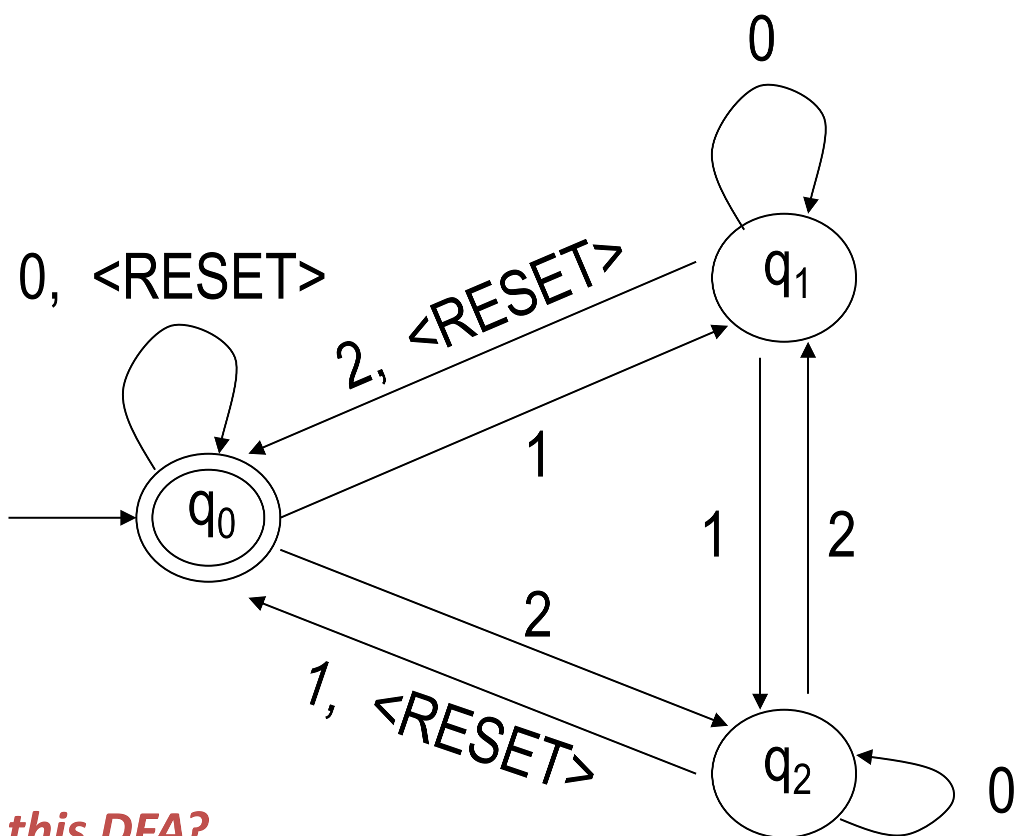
$L(M_4)$ = { w | w starts and ends with the same letter}

DFA example: graph- -> language



How to describe this DFA?

DFA example: graph- -> language



How to describe this DFA?

$L(M_5)$ = { w | after the last <RESET>, the sum of w is 0 modulo 3}

Design a DFA for a language

- Step 1: list all possible states

- Step 2: draw all the transitions between the states

- Step 3: add start and accept states

Example 1: language - - > figure

- $L(E_1)$={ w | w has odd amount of 1s }, Σ={0,1}

Step 1: define states

Example 1: language - - > figure

- $L(E_1)$={ w | w has odd amount of 1s }, Σ={0,1}

q_{even}: even amount of 1s
q_{odd} : odd amount of 1s

Example 1: language - - > figure

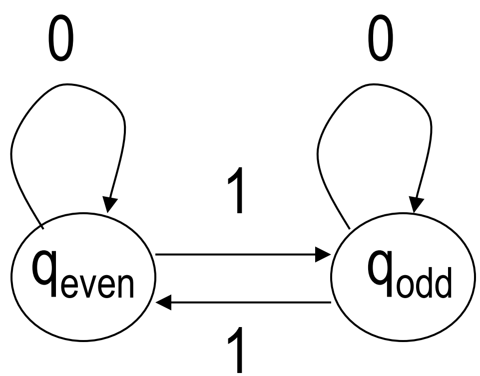
- $L(E_1)=\{ w \mid w \text{ has odd amount of 1s} \}, \Sigma=\{0,1\}$

q_{even}: even amount of 1s
q_{odd} : odd amount of 1s

Step 2: define transitions

Example 1: language - - > figure

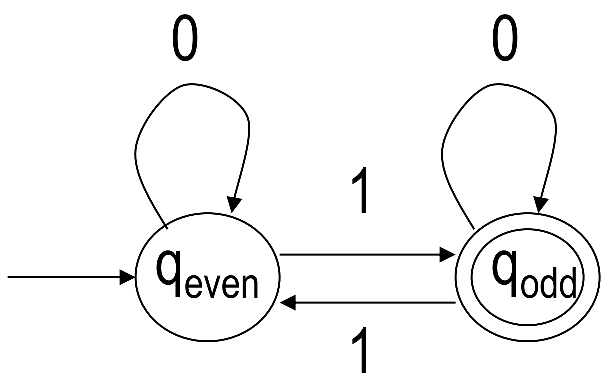
- $L(E_1)$={ w | w has odd amount of 1s }, Σ={0,1}



Step 3: define start state and accept states

Example 1: language - - > figure

- $L(E_1)$={ w | w has odd amount of 1s }, Σ={0,1}



Example 2: language - - > figure

- $L(E_2)=\{ w \mid w \text{ has substring } 001 \}, \Sigma=\{0,1\}$

Example 2: language - - > figure

- $L(E_2)=\{ w \mid w \text{ has substring } 001 \}, \Sigma=\{0,1\}$

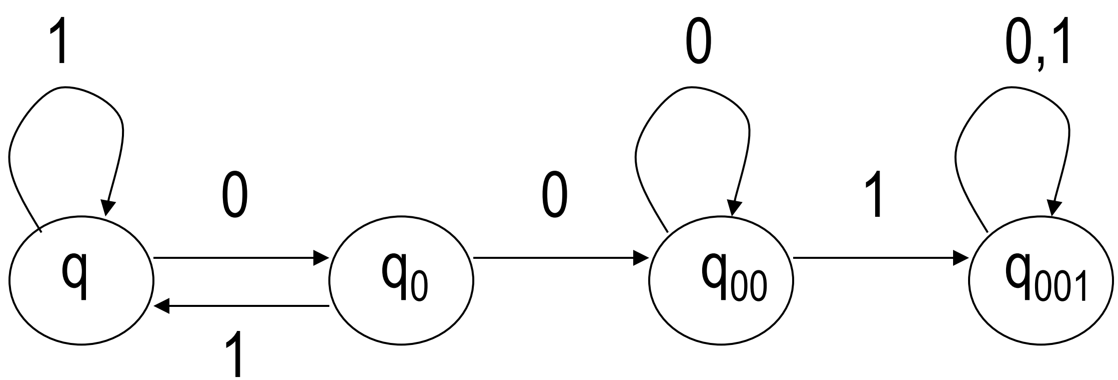
q: empty string
q_0: has substring 0
q_{00}: has substring 00
q_{001}: has substring 001

Example 2: language - - > figure

- $L(E_2)$={ w | w has substring 001 }, Σ={0,1}

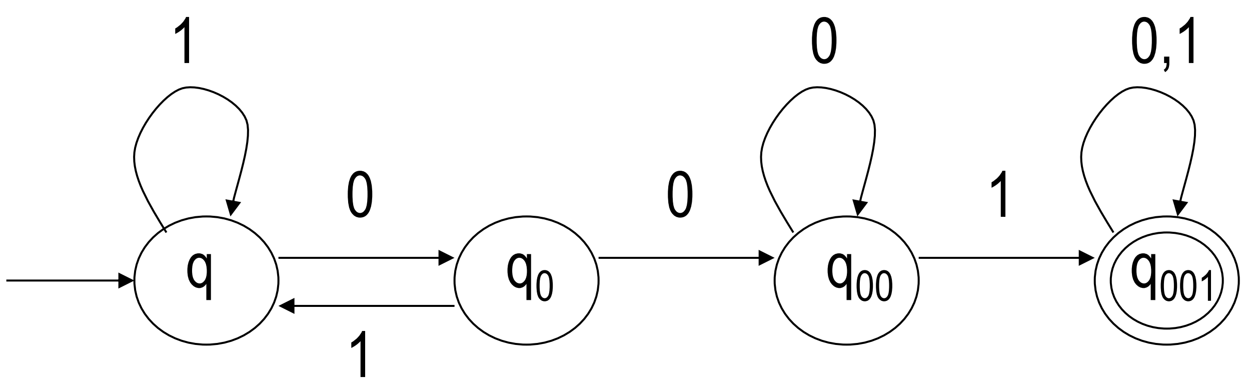
Example 2: language - - > figure

- $L(E_2)=\{ w \mid w \text{ has substring } 001 \}, \Sigma=\{0,1\}$



Example 2: language - - > figure

- $L(E_2)=\{ w \mid w \text{ has substring } 001 \}, \Sigma=\{0,1\}$



Example 3: language - - > figure

- L = Set of all strings that start with 0

  = {0, 00, 01, 000, 010, ...}

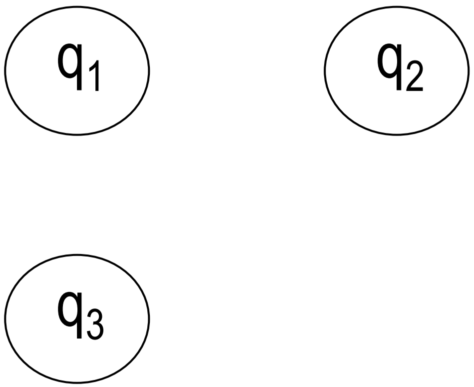
Can anyone draw the DFA?

Example 3: language - - > figure

- L = Set of all strings that start with 0

  = {0, 00, 01, 000, 010, ...}

q_1: ε
q_2: start with 0
q_3: start with 1

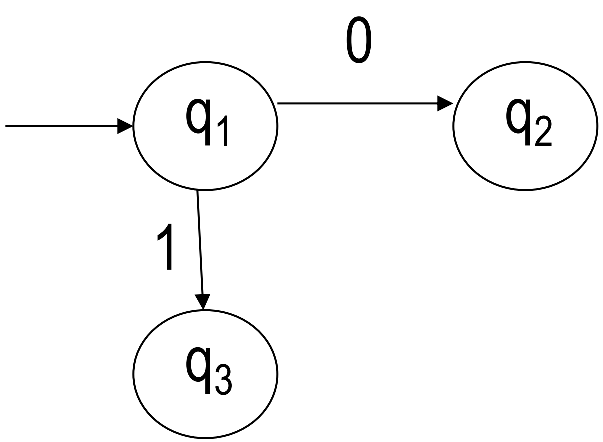


Example 3: language - - > figure

- L = Set of all strings that start with 0

  = {0, 00, 01, 000, 010, …}

q_1: ε
q_2: start with 0
q_3: start with 1

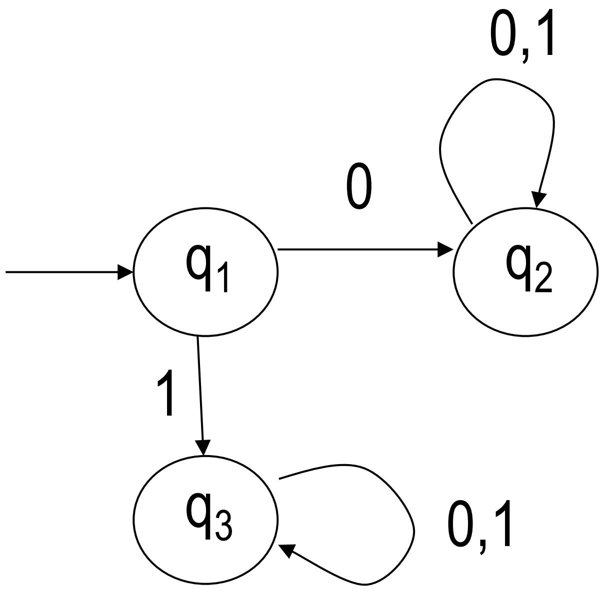


Example 3: language - - > figure

- L = Set of all strings that start with 0

  = {0, 00, 01, 000, 010, ...}

q_1: ε
q_2: start with 0
q_3: start with 1

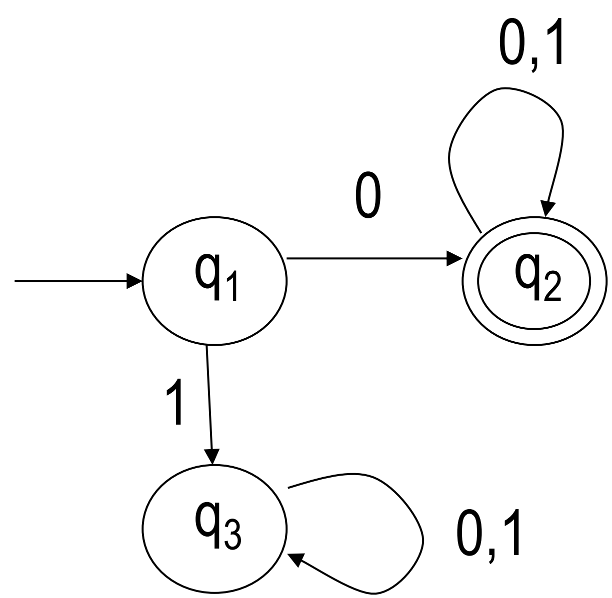


Example 3: language - - > figure

- L = Set of all strings that start with 0

  = {0, 00, 01, 000, 010, …}

q_1: ε
q_2: start with 0
q_3: start with 1



Example 4: language - - > figure

- L = Set of all strings over {0,1} that of length is 2

  = {00, 01, 10, 11}

Can anyone draw the DFA?

Example 4: language - - > figure

- L = Set of all strings over {0,1} that of length is 2

  = {00, 01, 10, 11}

q_1: ε
q_2: length is 1
q_3: length is 2
q_4: length is 3 or more

Example 4: language - - > figure

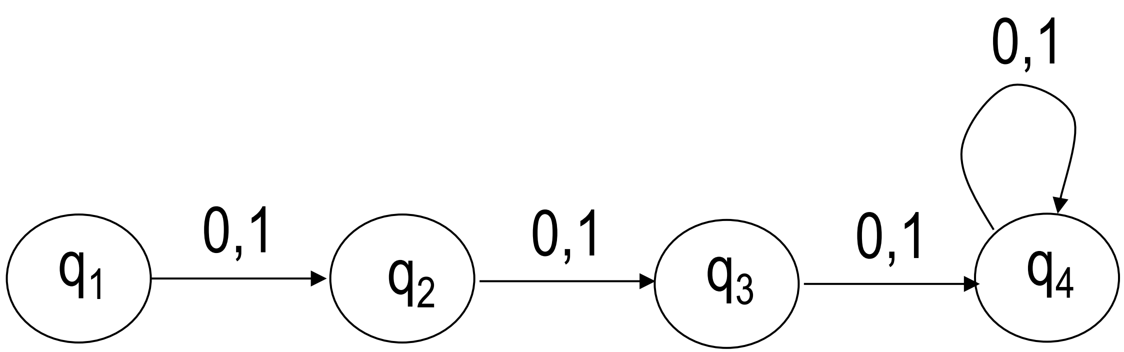
- L = Set of all strings over {0,1} that of length is 2

  = {00, 01, 10, 11}

Example 4: language - - > figure

- L = Set of all strings over {0,1} that of length is 2

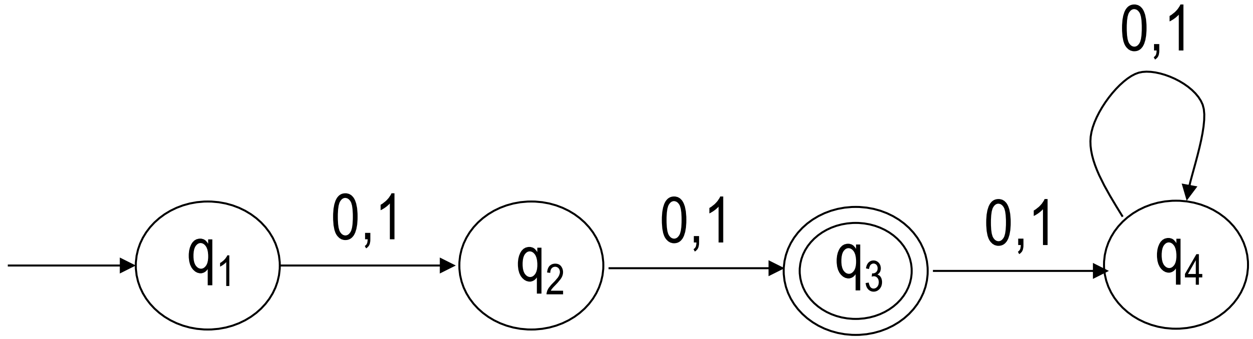
  = {00, 01, 10, 11}



Example 4: language - - > figure

- L = Set of all strings over {0,1} that of length is 2

  = {00, 01, 10, 11}



Example 5: language - - > figure

- L = Set of strings over {a,b} that contains string aabb in it

Can anyone draw the DFA?

Example 5: language - - > figure

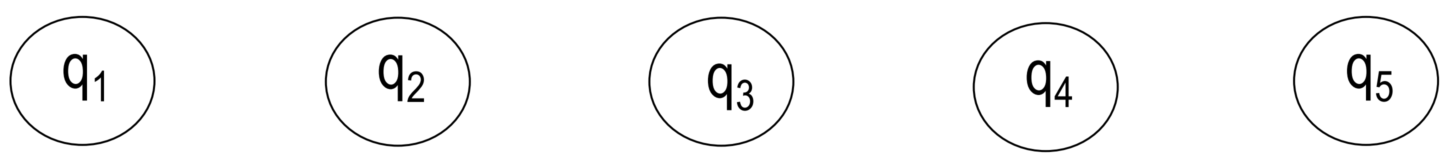
- L = Set of strings over {a,b} that contains string aabb in it

q_1: contains nothing
q_2: contains a
q_3: contains aa
q_4: contains aab
q_5: contains aabb

Example 5: language - - > figure

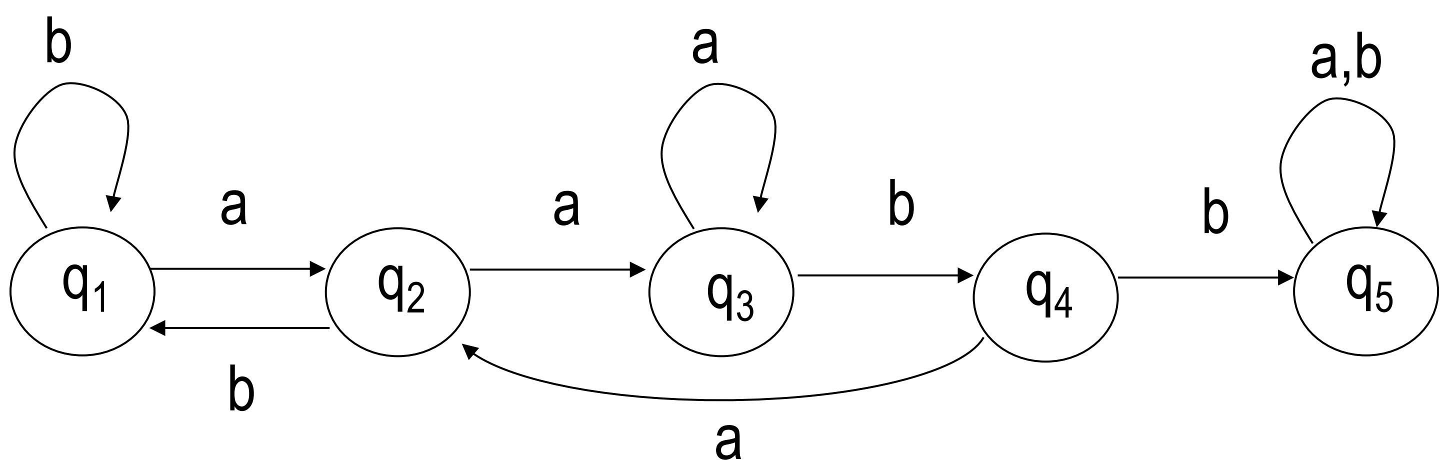
- L = Set of strings over {a,b} that contains string aabb in it

q_1: contains nothing
q_2: contains a
q_3: contains aa
q_4: contains aab
q_5: contains aabb



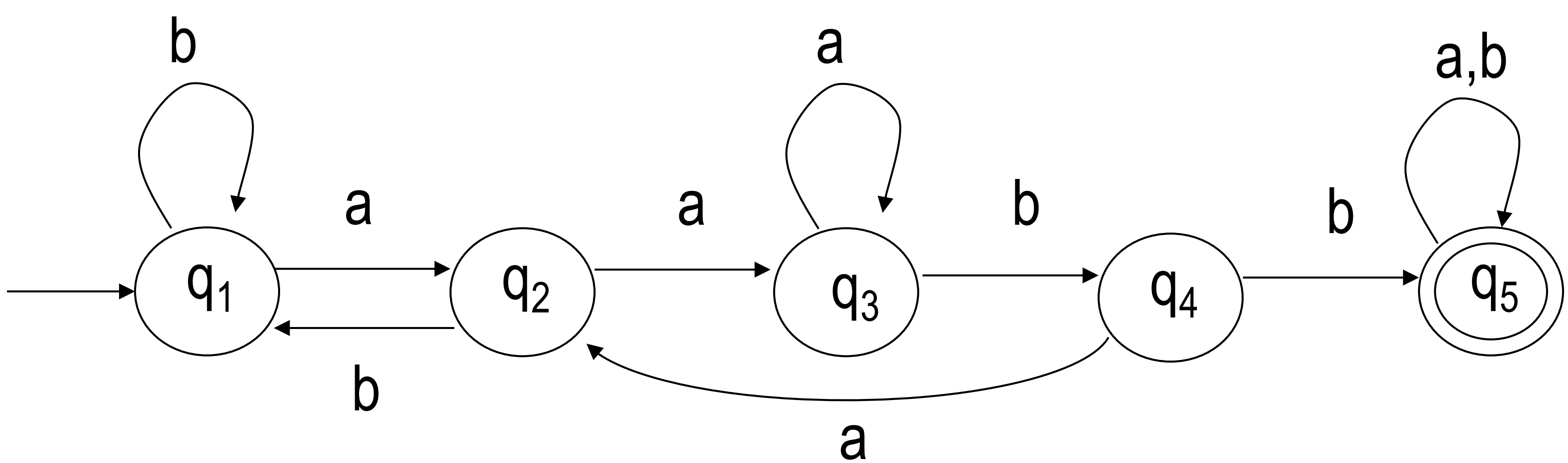
Example 5: language - - > figure

- L = Set of strings over {a,b} that contains string aabb in it



Example 5: language - - > figure

- L = Set of strings over {a,b} that contains string aabb in it



Example 6: language - - > figure

- L = Set of strings over {a,b} that does not contain string aabb in it

Can anyone draw the DFA?

Example 6: language - - > figure

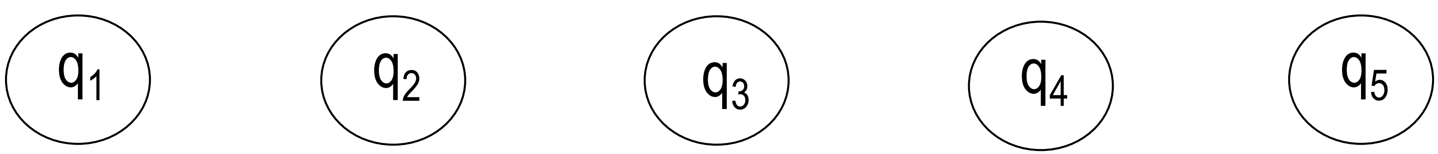
- L = Set of strings over {a,b} that does not contain string aabb in it

q_1: contains nothing
q_2: contains a
q_3: contains aa
q_4: contains aab
q_5: contains aabb

Example 6: language - - > figure

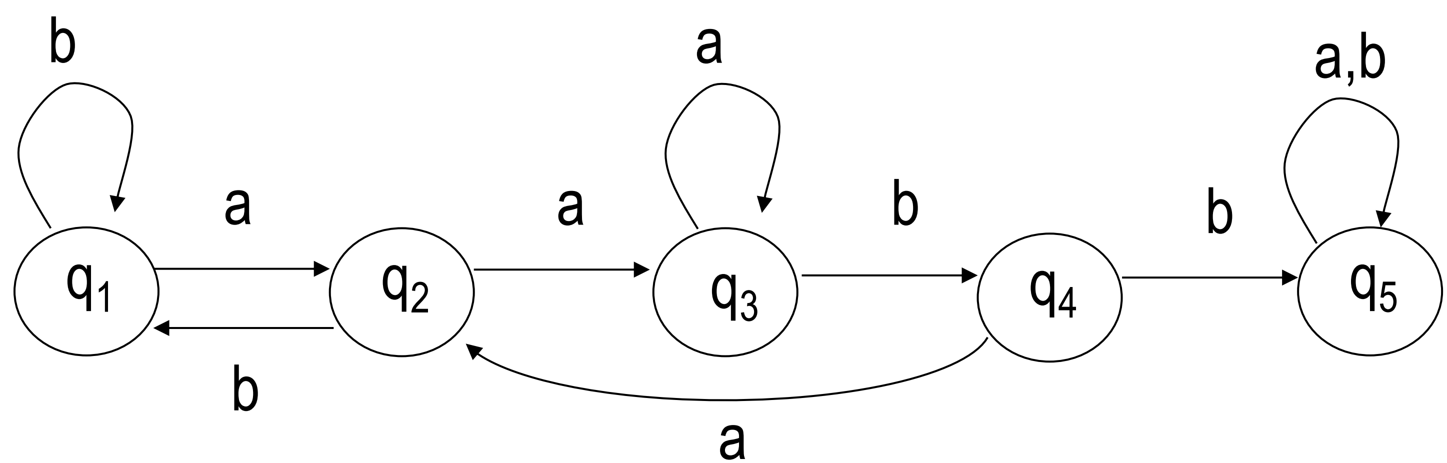
- L = Set of strings over {a,b} that does not contain string aabb in it

q_1: contains nothing
q_2: contains a
q_3: contains aa
q_4: contains aab
q_5: contains aabb



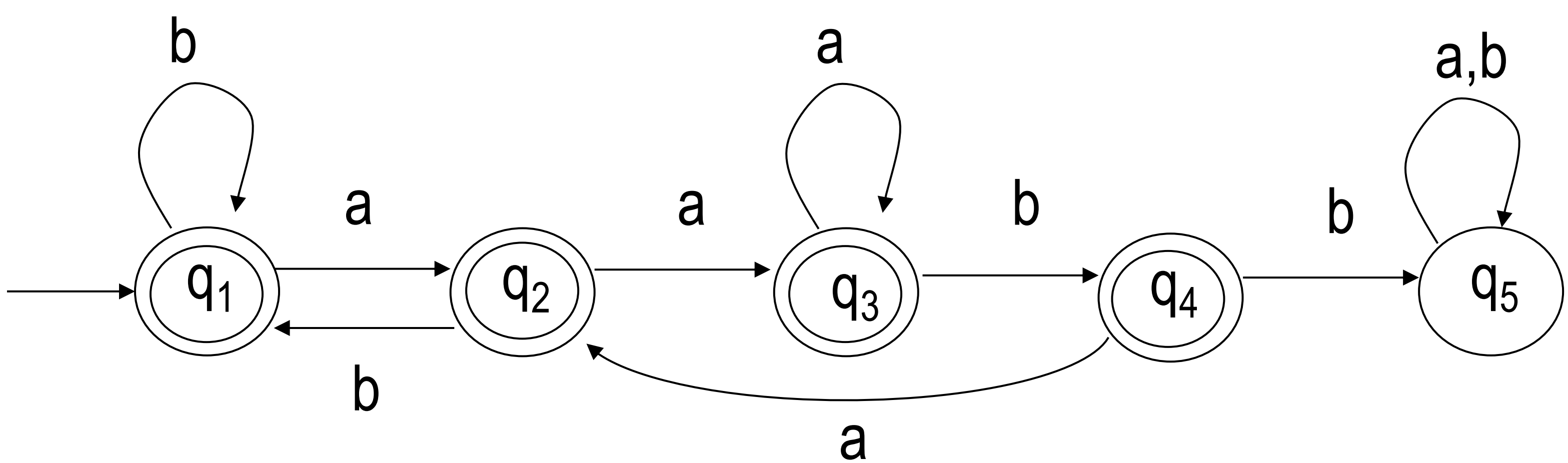
Example 6: language - - > figure

- L = Set of strings over {a,b} that does not contain string aabb in it



Example 6: language - - > figure

- L = Set of strings over {a,b} that does not contain string aabb in it



Definition of nondeterministic finite automaton

$N = (Q,\Sigma,\delta,q_0,F)$, where

- Q: finite set of states
- Σ: finite alphabet as input; ($\Sigma_\varepsilon=\Sigma\cup\{\varepsilon\}$)
- δ: $Q\times\Sigma_\varepsilon\rightarrow P(Q)$, transition function
- $q_0\in Q$: start state
- $F\subseteq Q$: accept state set

Design a NFA for a language

- Step 1: list all possible states
- Step 2: draw all the transitions between the states
- Step 3: add start and accept states

Some practices

- L_1 = {Set of all strings that end with 0}, $\Sigma = \{0,1\}$;

Can anyone draw the NFA?

Some practices

- L_1 = {Set of all strings that end with 0}, $\Sigma = \{0,1\}$;

Can anyone draw the NFA?

q1: all the strings
q2: last letter is 0

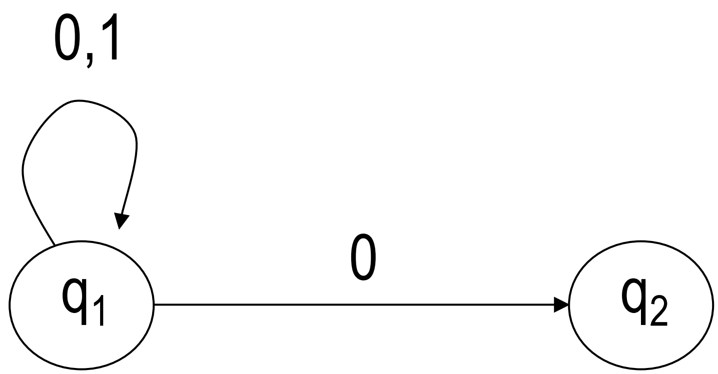
NFA of L_1

Some practices

- L_1 = {Set of all strings that end with 0}, $\Sigma = \{0,1\}$;

Can anyone draw the NFA?

q1: all the strings
q2: last letter is 0



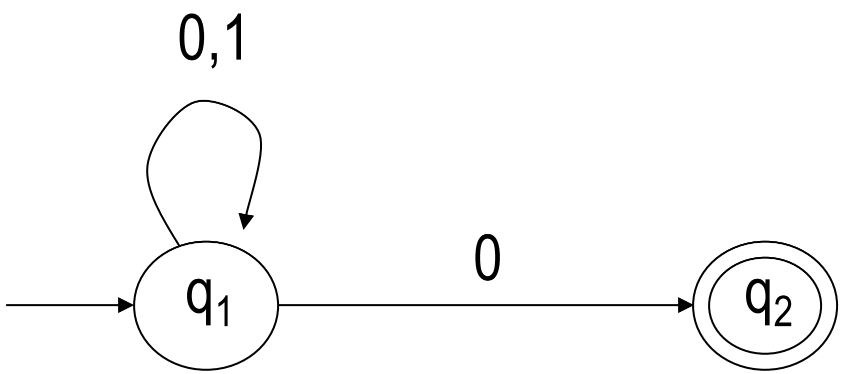
NFA of L_1

Some practices

- L_1 = {Set of all strings that end with 0}, Σ = {0,1};

Can anyone draw the NFA?

q1: all the strings
q2: last letter is 0



NFA of L_1

Some practices

- L_2 = {Set of all strings that start with 0}, $\Sigma = \{0,1\}$;

Can anyone draw the NFA?

Some practices

- L_2 = {Set of all strings that start with 0}, Σ = {0,1};

Can anyone draw the NFA?

q1: empty string
q2: first letter is 0

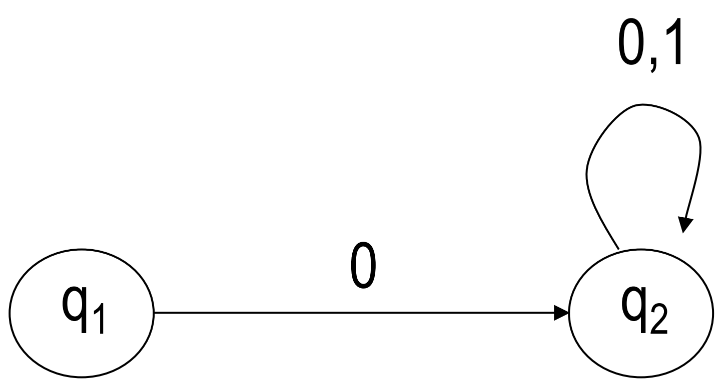
NFA of L_2

Some practices

- L_2 = {Set of all strings that start with 0}, $\Sigma = \{0,1\}$;

Can anyone draw the NFA?

q1: empty string
q2: first letter is 0



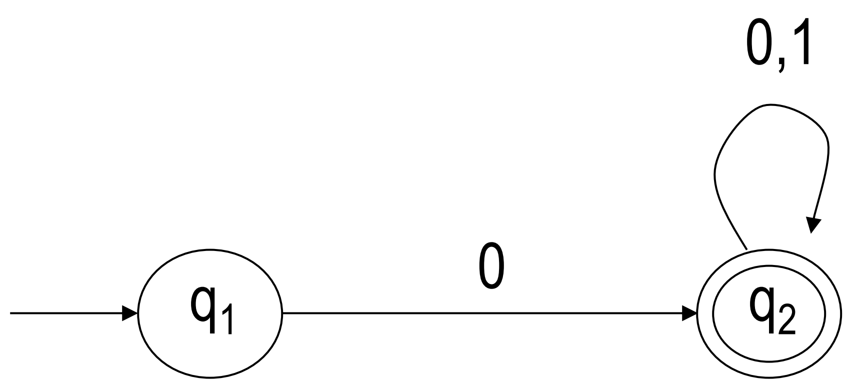
NFA of L_2

Some practices

- L_2 = {Set of all strings that start with 0}, Σ = {0,1};

Can anyone draw the NFA?

q1: empty string
q2: first letter is 0



NFA of L_2

Some practices

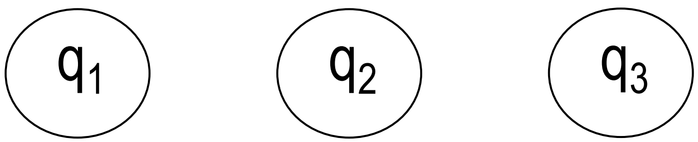
- L_3 = {Set of all strings that length is 2}, Σ = {0,1};

Can anyone draw the NFA?

Some practices

- L_3 = {Set of all strings that length is 2}, Σ = {0,1};

Can anyone draw the NFA?



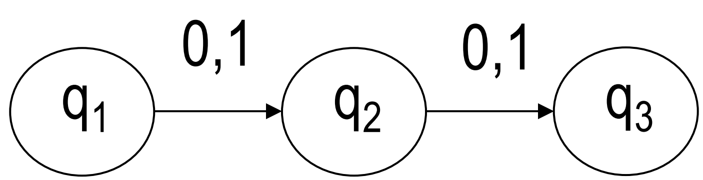
q1: strings that length is 0
q2: strings that length is 1
q3: strings that length is 2

NFA of L_3

Some practices

- L_3 = {Set of all strings that length is 2}, $\Sigma = \{0,1\}$;

Can anyone draw the NFA?



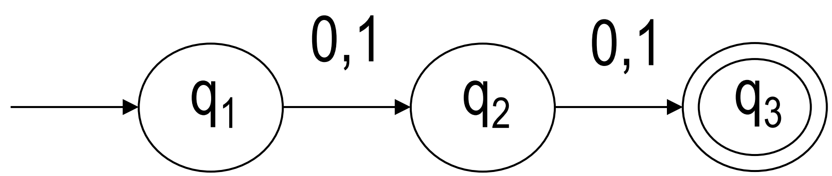
q1: strings that length is 0
q2: strings that length is 1
q3: strings that length is 2

NFA of L_3

Some practices

- L_3 = {Set of all strings that length is 2}, $\Sigma = \{0,1\}$;

Can anyone draw the NFA?



q1: strings that length is 0
q2: strings that length is 1
q3: strings that length is 2

NFA of L_3

Some practices

- L_4 = {Set of all strings that contain '0'}, $\Sigma = \{0,1\}$;

Can anyone draw the NFA?

Some practices

- L_4 = {Set of all strings that contain '0'}, $\Sigma = \{0,1\}$;

Can anyone draw the NFA?

q1: all strings
q2: strings that contain 0

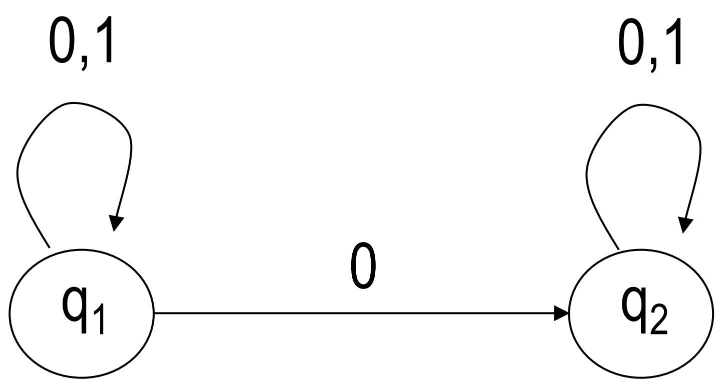
NFA of L_4

Some practices

- L_4 = {Set of all strings that contain ‘0’}, Σ = {0,1};

Can anyone draw the NFA?

q1: all strings
q2: strings that contain 0



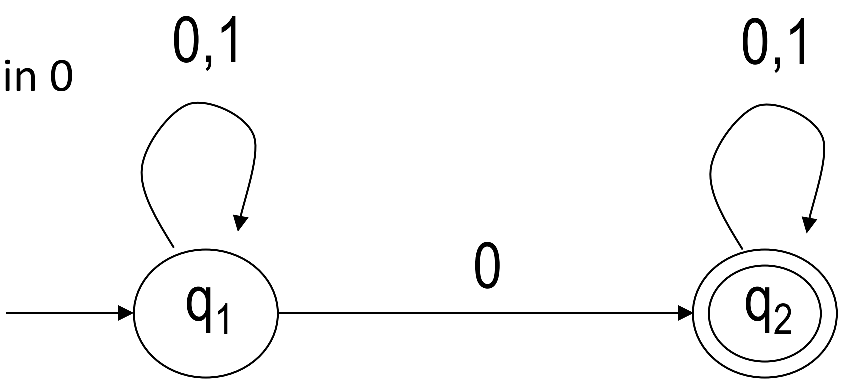
NFA of L_4

Some practices

- L_4 = {Set of all strings that contain '0'}, $\Sigma = \{0,1\}$;

Can anyone draw the NFA?

q1: all strings
q2: strings that contain 0



NFA of L_4

Some practices

- L_5 = {Set of all strings that starts with '10'}, Σ = {0,1};

Can anyone draw the NFA?

NFA of L_5

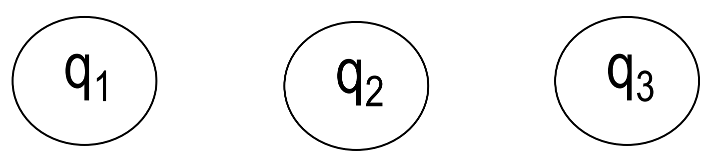
Some practices

- L_5 = {Set of all strings that starts with '10'}, Σ = {0,1};

Can anyone draw the NFA?

q1: all strings
q2: strings that start with 1
q3: strings that start with 10

NFA of L_5



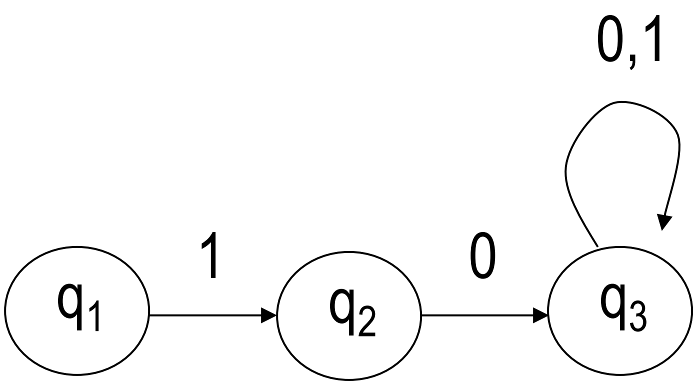
Some practices

- L_5 = {Set of all strings that starts with '10'}, Σ = {0,1};

Can anyone draw the NFA?

q1: all strings
q2: strings that start with 1
q3: strings that start with 10

NFA of L_5



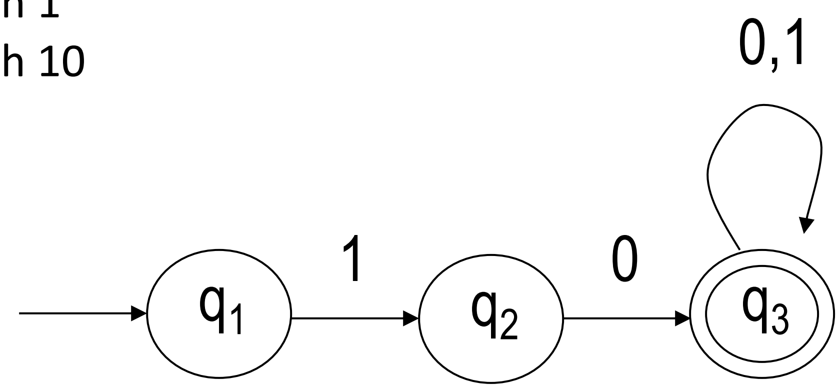
Some practices

- L_5 = {Set of all strings that starts with '10'}, Σ = {0,1};

Can anyone draw the NFA?

q1: all strings
q2: strings that start with 1
q3: strings that start with 10

NFA of L_5



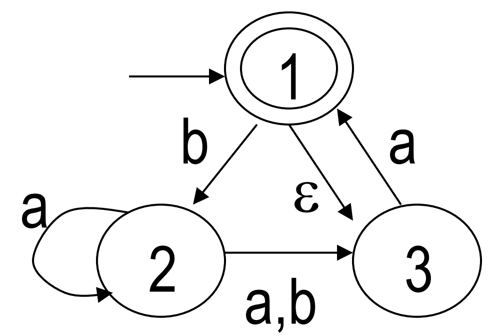
Equivalence of NFAs and DFAs

- Step 1: Draw all the states in DFAs
- Step 2: Define the transitions in DFAs based on the NFAs
- Step 3: Define the start state and accept state in DFAs
- Step 4: Remove all inaccessible states

Example

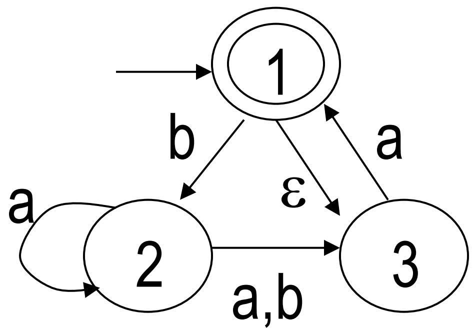
- NFA $N_4=(\{1,2,3\},\{a,b\},\delta,1,\{1\})$

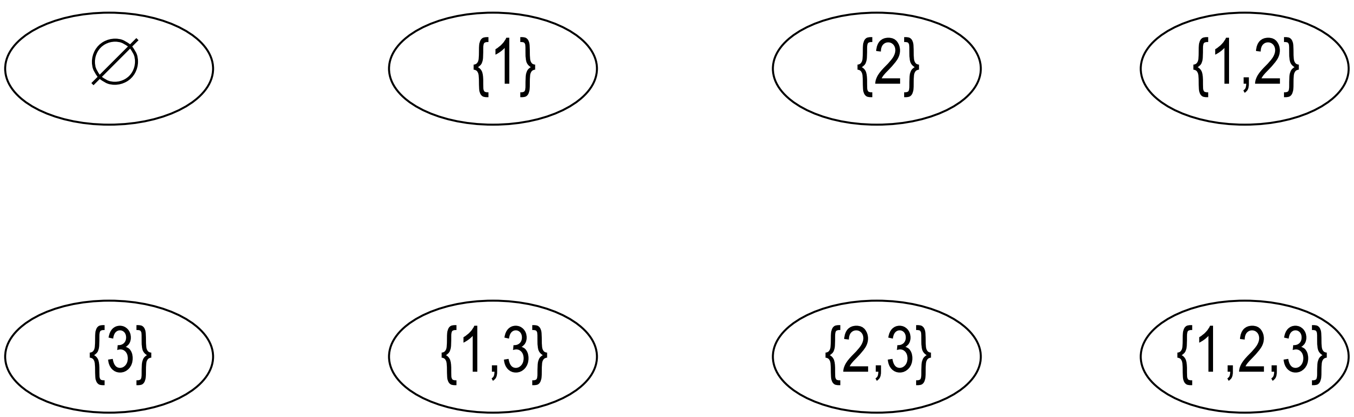
  What is its equivalent DFA?



Example

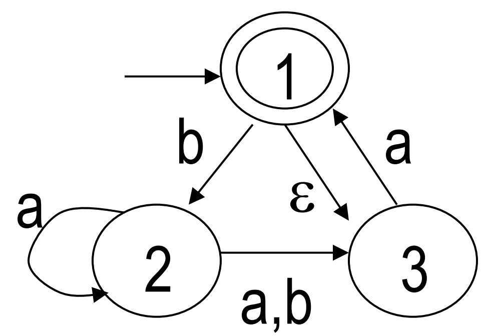
- List all subset states

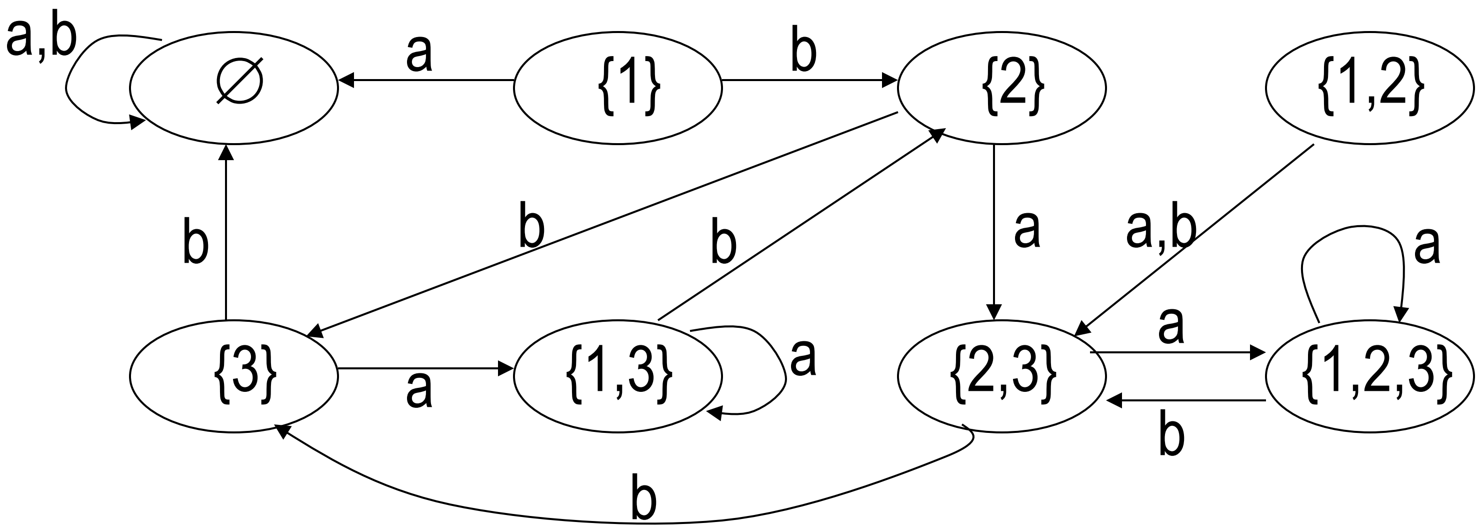




Example

- Add transitions

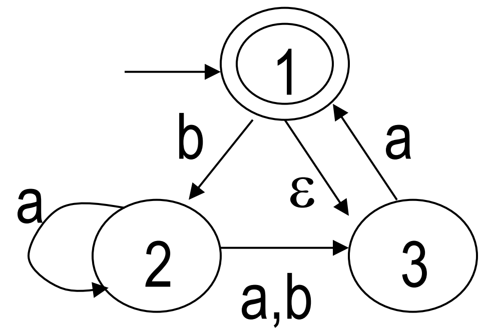


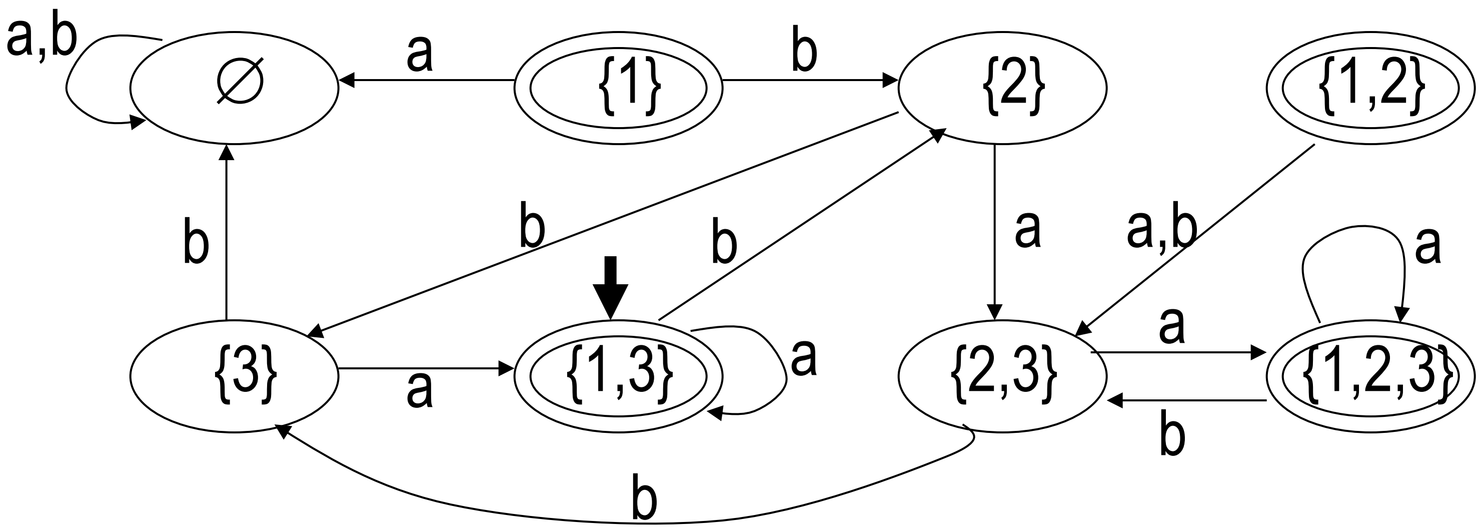


Example

Closure E(R) on ε
Collection of states that can be reached from members of R by going only along ε arrows, including the members of R themselves.

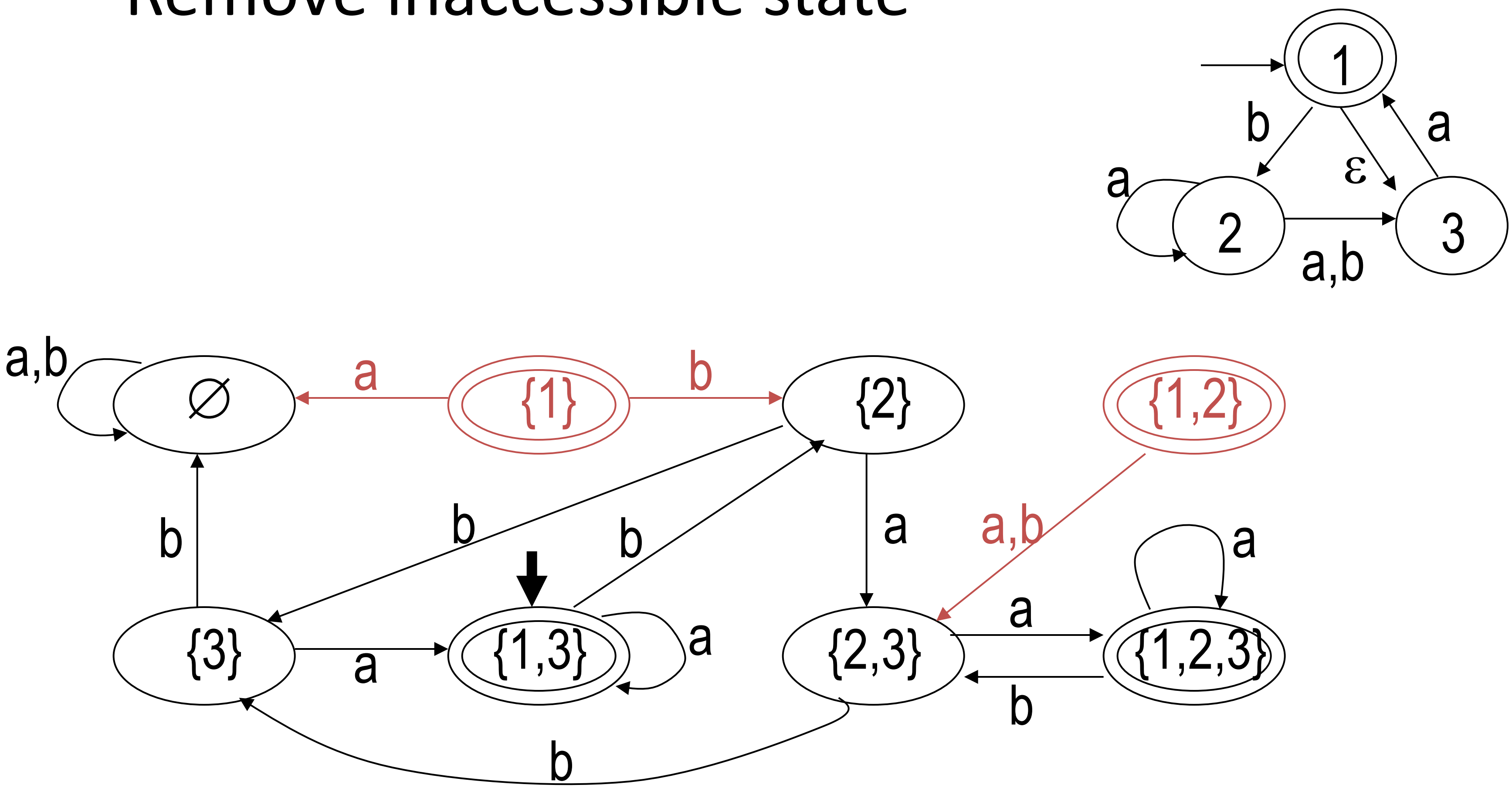
- Start state: E({1}) = {1,3}
- Accept state: all states with 1





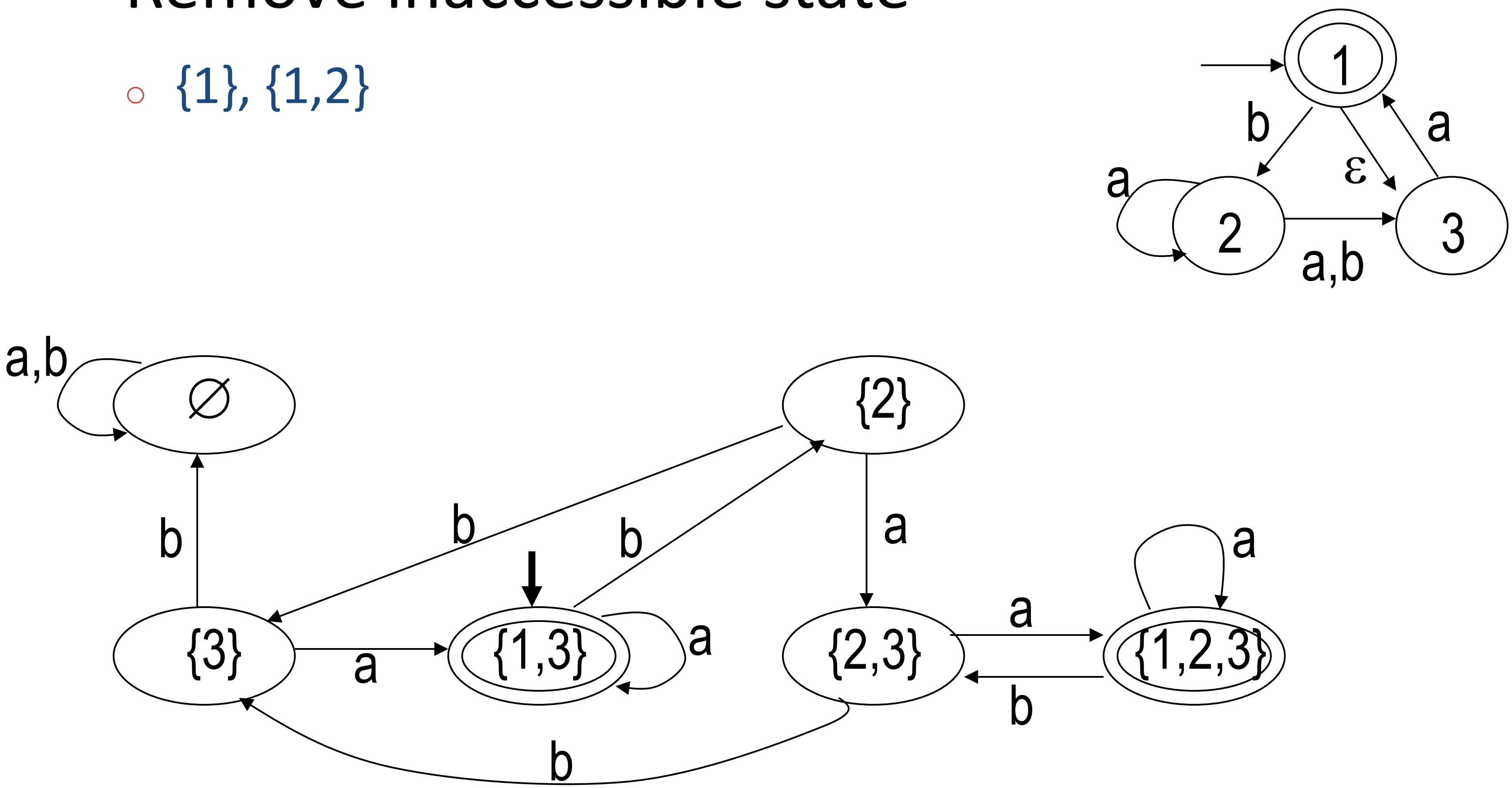
Example

- Remove inaccessible state



Example

- Remove inaccessible state
  - {1}, {1,2}



Regular operations

	DFA/NFA	PDA	TM
Union	close	?	?
Concatenation	close	?	?
Star	close	?	?
Complement	close	?	?
Boolean operation	close	?	?

Definition of regular expression

- R is regular expression if R is
  - a, where $a \in \Sigma$, length is 1;
  - ε;
  - $\emptyset$;
  - Union: $(R_1 \cup R_2)$, where R_1 and R_2 are all regular expressions;
  - Concatenation: $(R_1 R_2)$, where R_1 and R_2 are all regular expressions;
  - Star: $(R_1{}^*)$, where R_1 is regular expression.

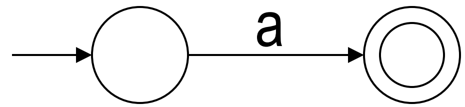
Priority:
$* > \circ > \cup$

- L(R): the language of R
  - $L(1\Sigma^*)$: language that starts with 1

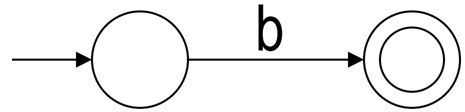
Examples: create NFAs for regular expression

- Create (ab∪a)*

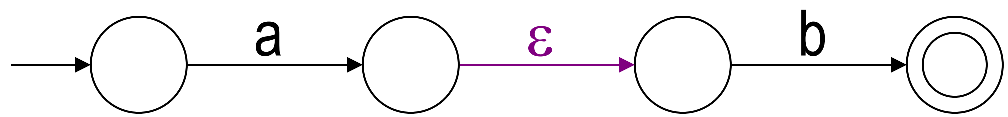
1. a



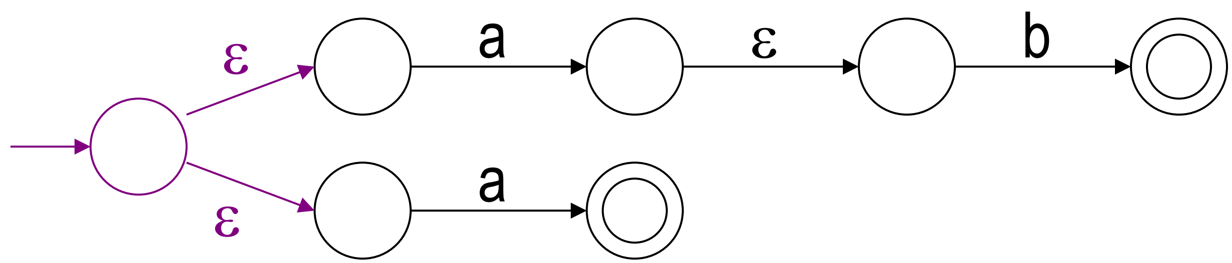
2. b



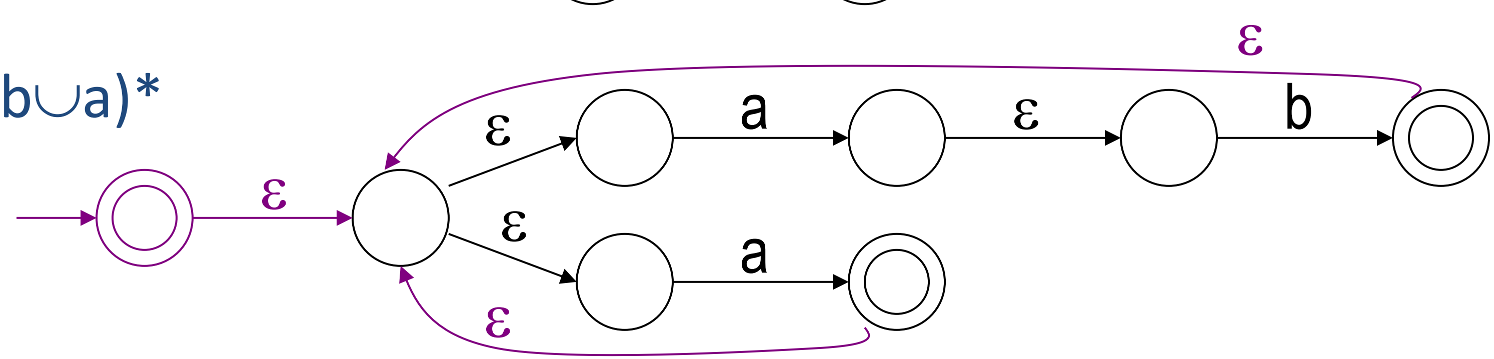
3. ab



4. ab∪a

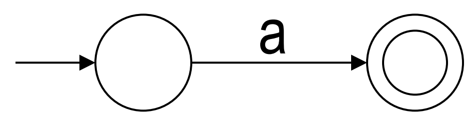


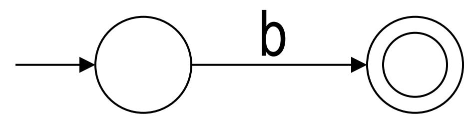
5. (ab∪a)*



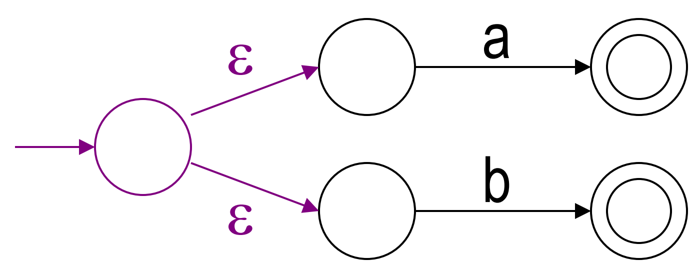
Examples: create NFAs for regular expression

- Create (a∪b)*aba
  - a

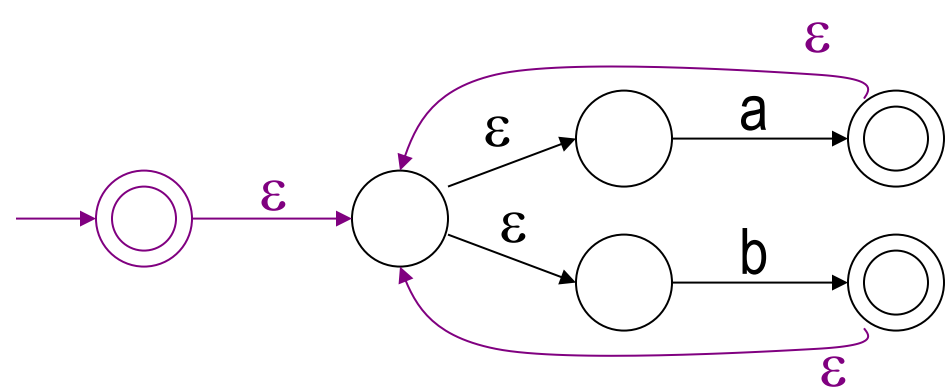




  - b

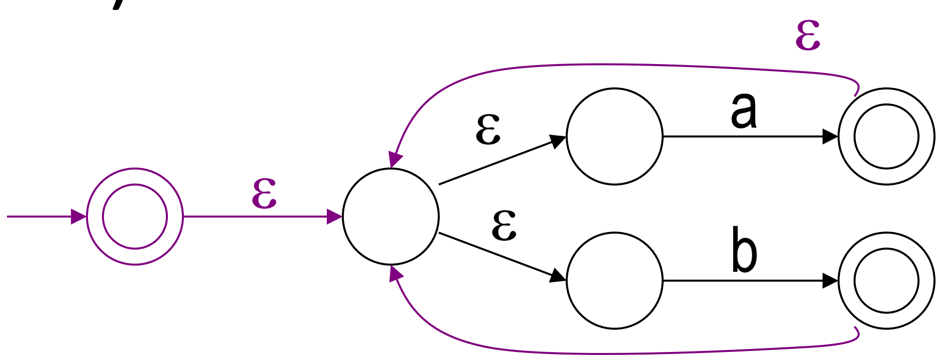


  - a∪b
  - (a∪b)*

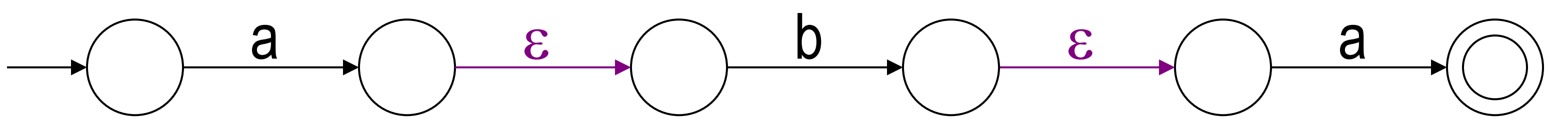


Examples: create NFAs for regular expression

- Create (a∪b)*aba
  - (a∪b)*

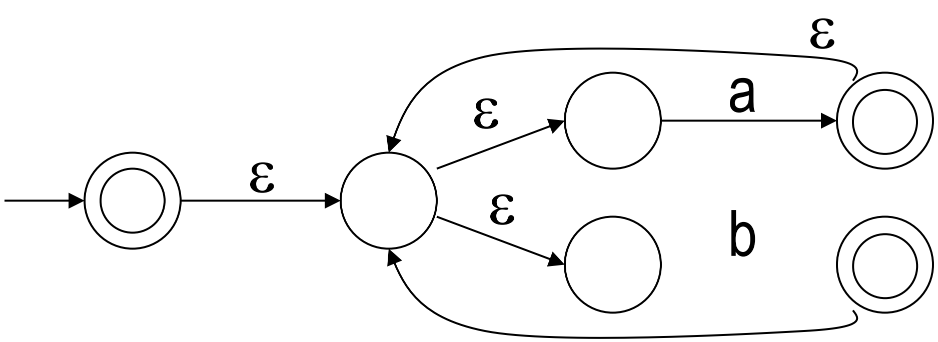


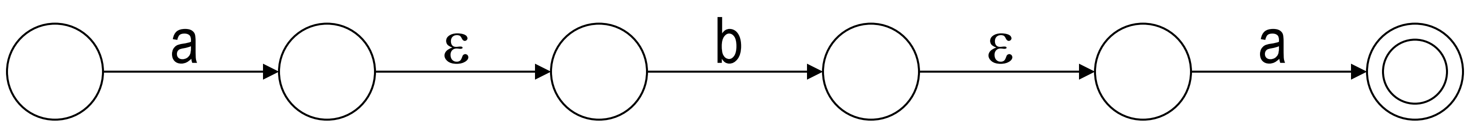
  - aba



Examples: create NFAs for regular expression

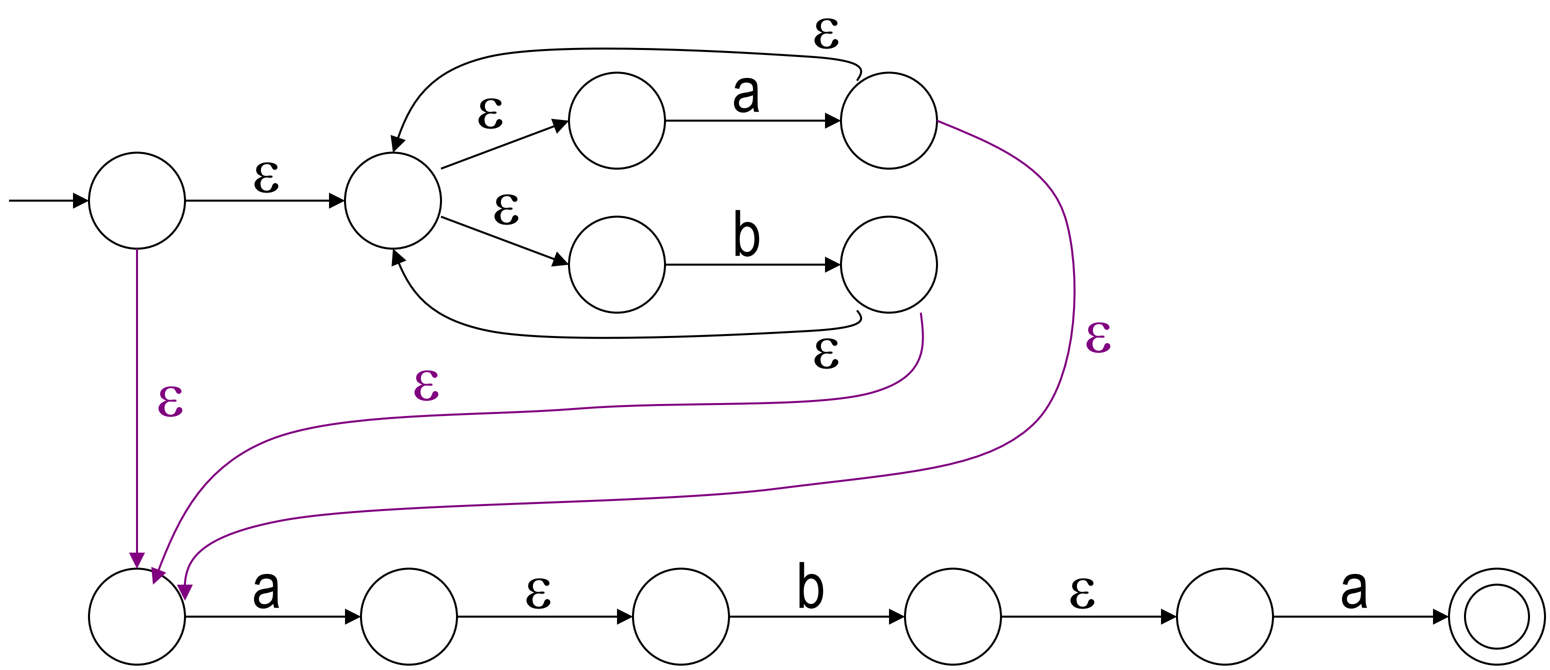
- Create (a∪b)*aba





Examples: create NFAs for regular expression

- Create (a∪b)*aba



Design RE based on description

- Let $\Sigma=\{0,1\}$,
- w starts with 1
  - $1\Sigma^*$

Design RE based on description

- Let $\Sigma=\{0,1\}$,
- w has an even length
  - $(\Sigma\Sigma)^*$

Design RE based on description

- Let $\Sigma=\{0,1\}$,
- w has an odd length
  - $\Sigma(\Sigma\Sigma)^*$

Design RE based on description

- Let $\Sigma=\{0,1\}$,
- w contains the substring 111
  - $\Sigma^*111\Sigma^*$

Design RE based on description

- Let $\Sigma=\{0,1\}$,
- w contains at least one 1
  - $\Sigma^*1\Sigma^*$

Design RE based on description

- Let $\Sigma=\{0,1\}$,
- w contains at most one 1
  - $0^*10^* \cup 0^*$

Design RE based on description

- Let $\Sigma=\{0,1\}$,
- w contains at least two 1s
  - $\Sigma^*1\Sigma^*1\Sigma^*$

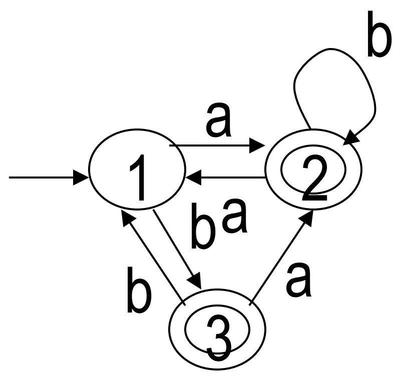
Design RE based on description

- Let $\Sigma=\{0,1\}$,
- w contains at most two 0s
  - $1^*01^*01^* \cup 1^*01^* \cup 1^*$

Design RE based on description

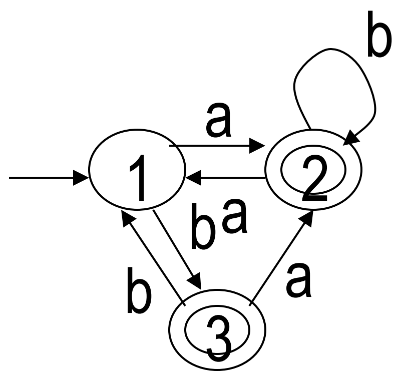
- Let $\Sigma=\{0,1\}$,
- w contains at least two 1s or contains at most two 0s
  - $\Sigma^*1\Sigma^*1\Sigma^* \cup 1^*01^*01^*$

DFA - -> Regular expression

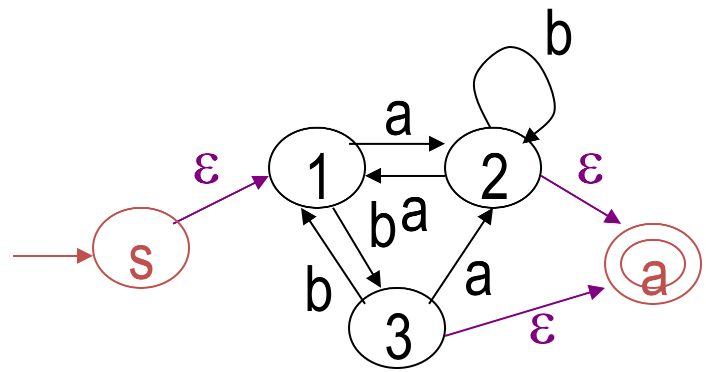


DFA

DFA - -> Regular expression

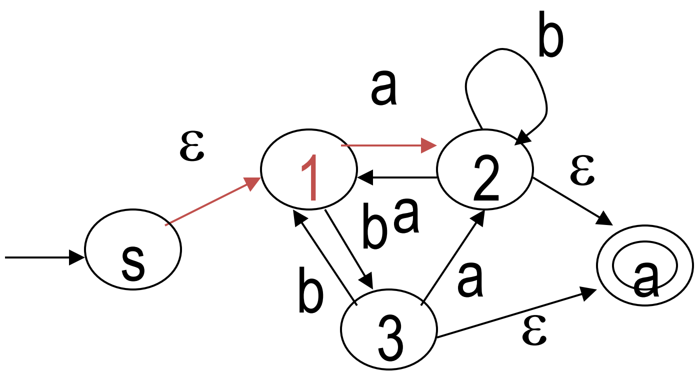


DFA

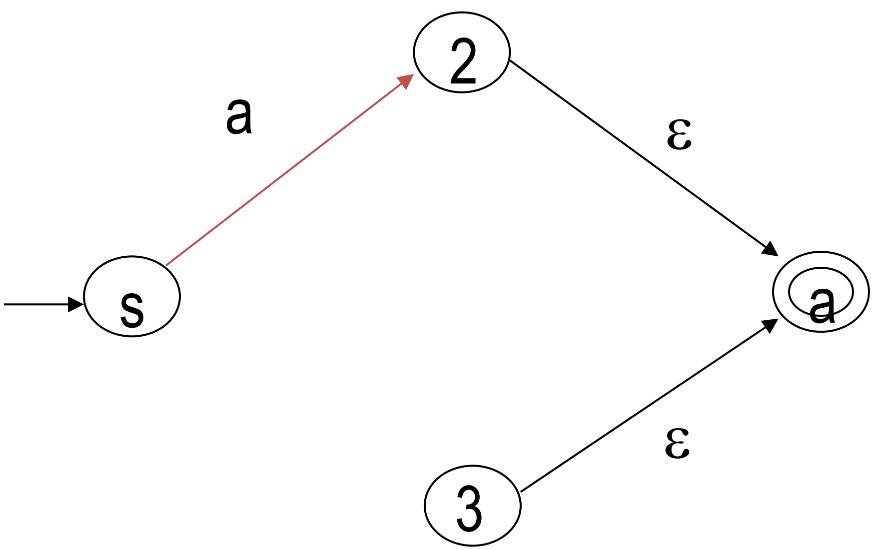


GNFA

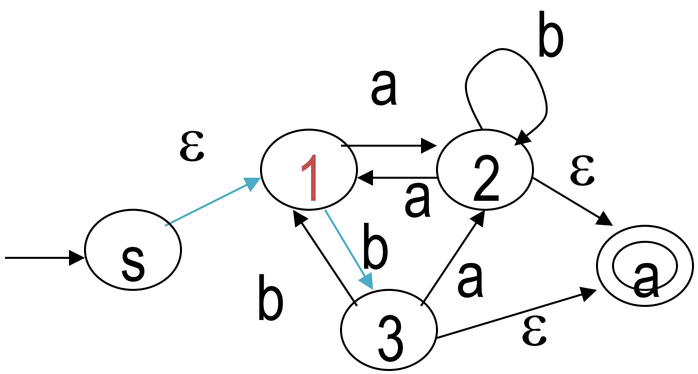
DFA - -> Regular expression



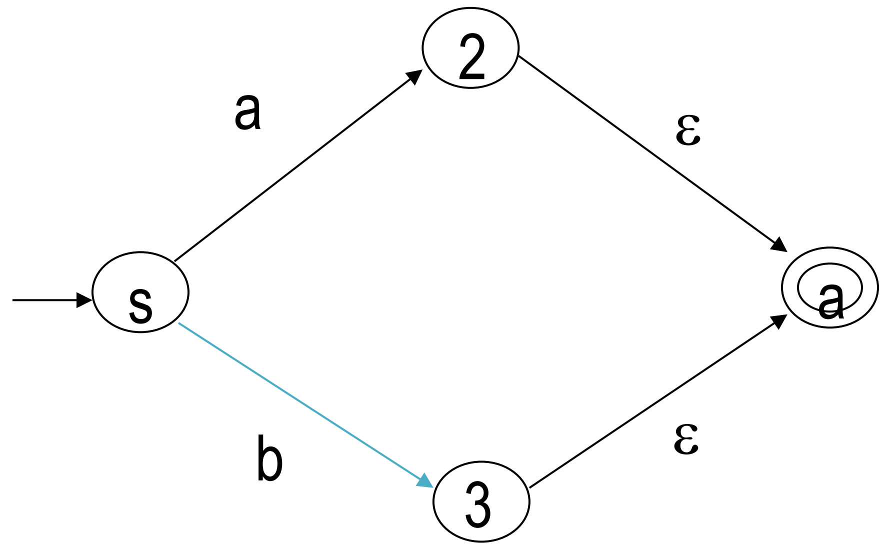
GNFA



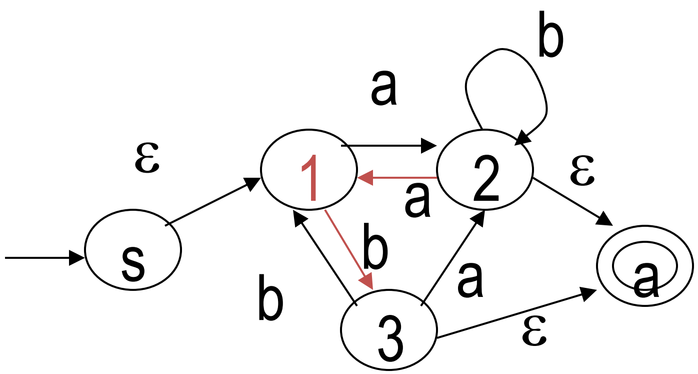
DFA - -> Regular expression



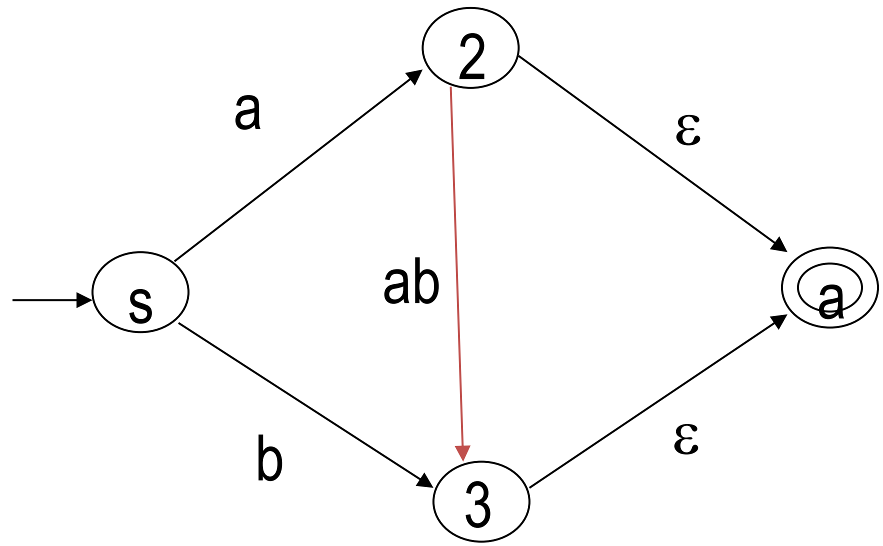
GNFA



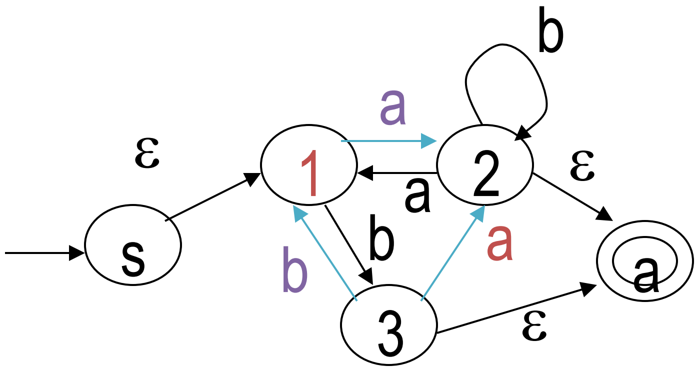
DFA - -> Regular expression



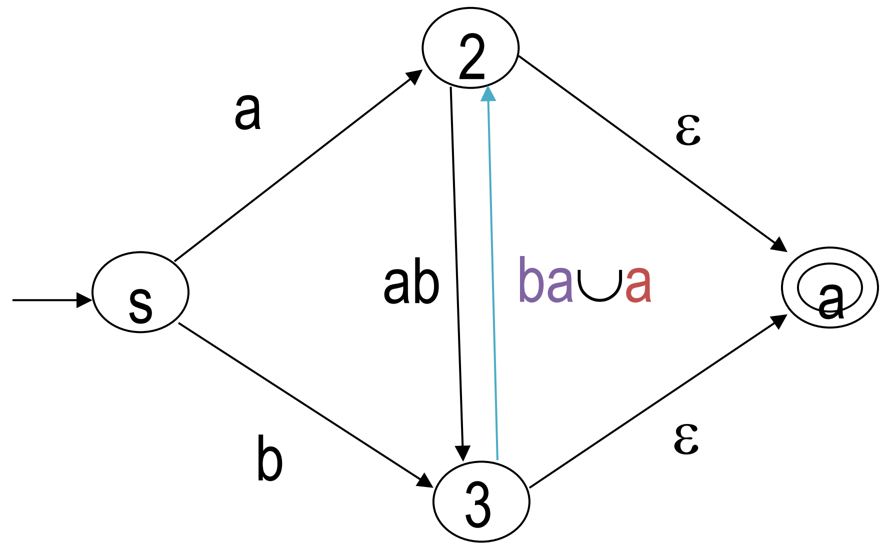
GNFA



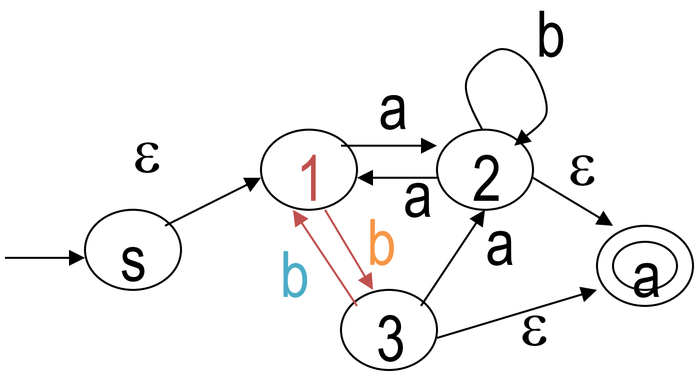
DFA - -> Regular expression



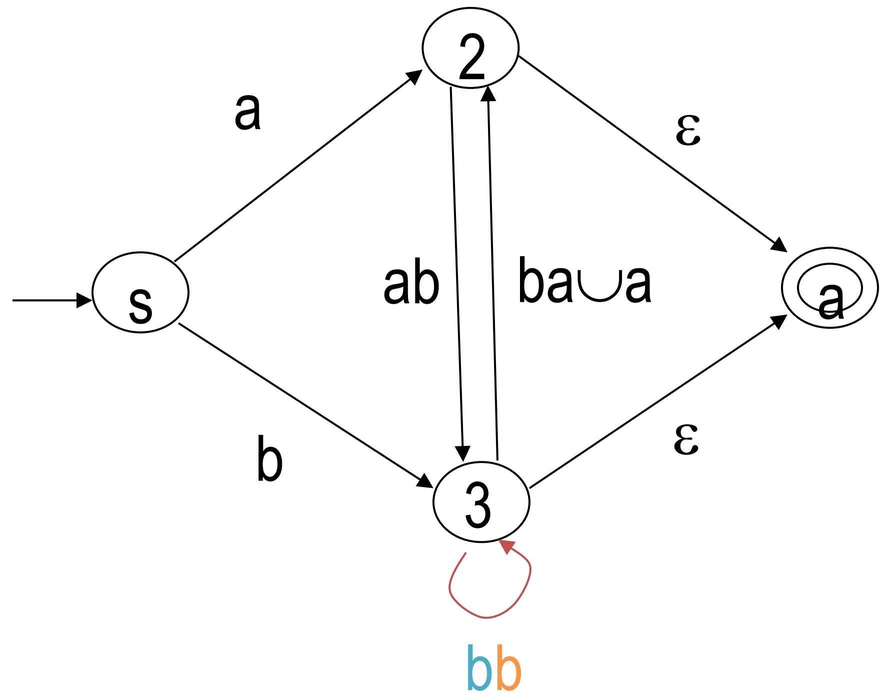
GNFA



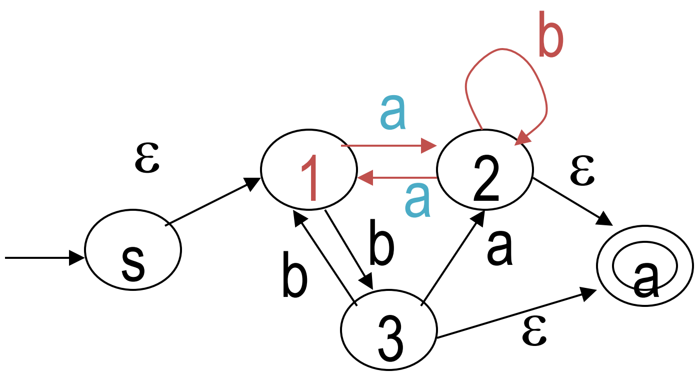
DFA - -> Regular expression



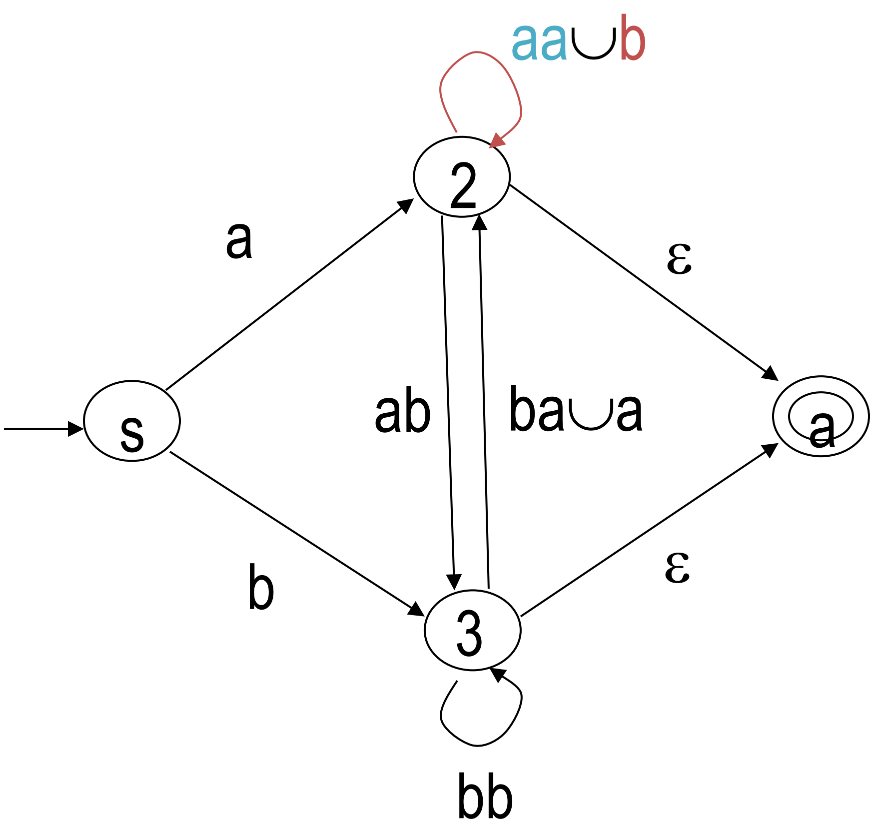
GNFA



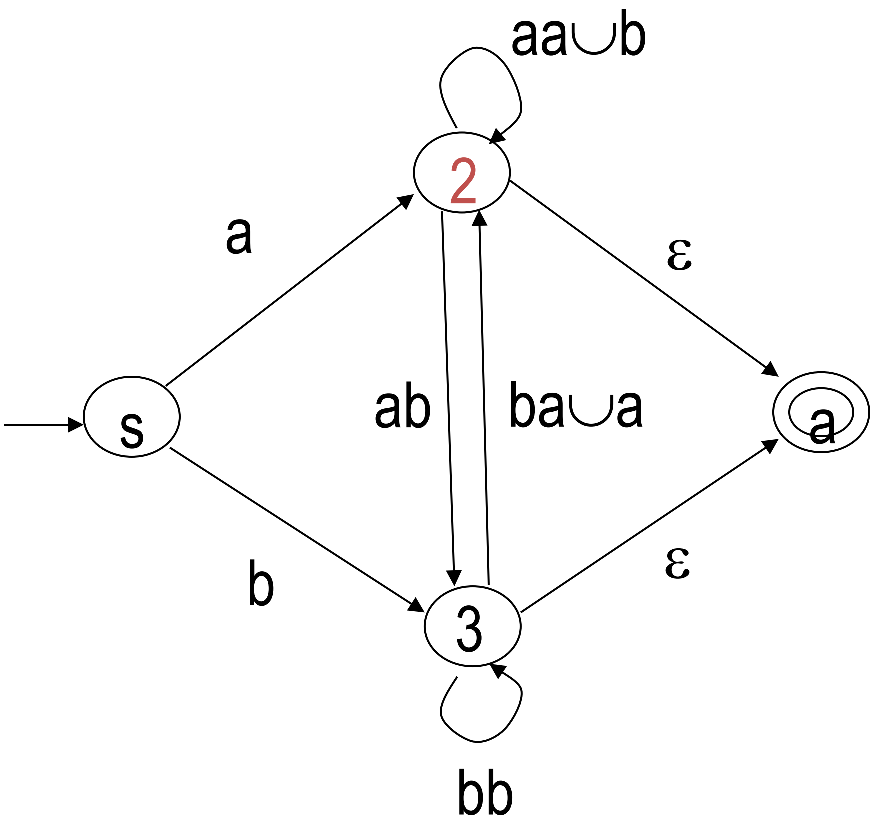
DFA - -> Regular expression

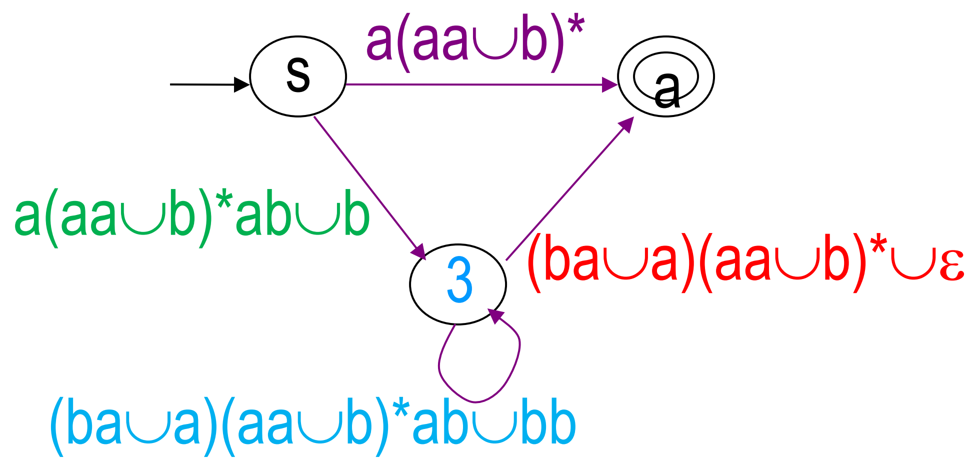


GNFA

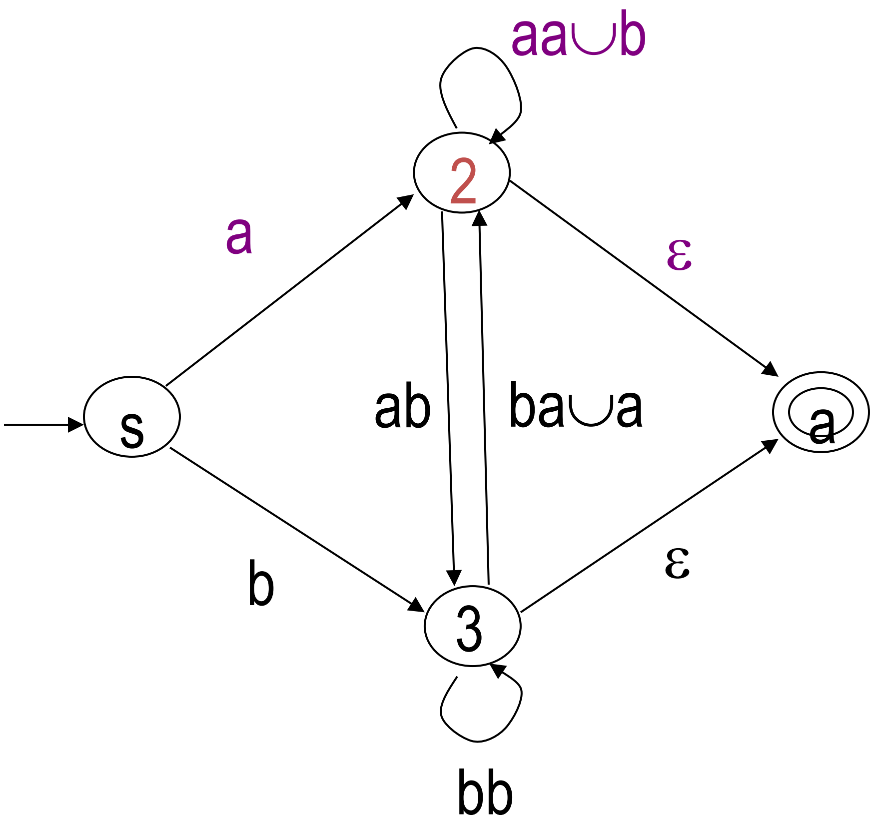


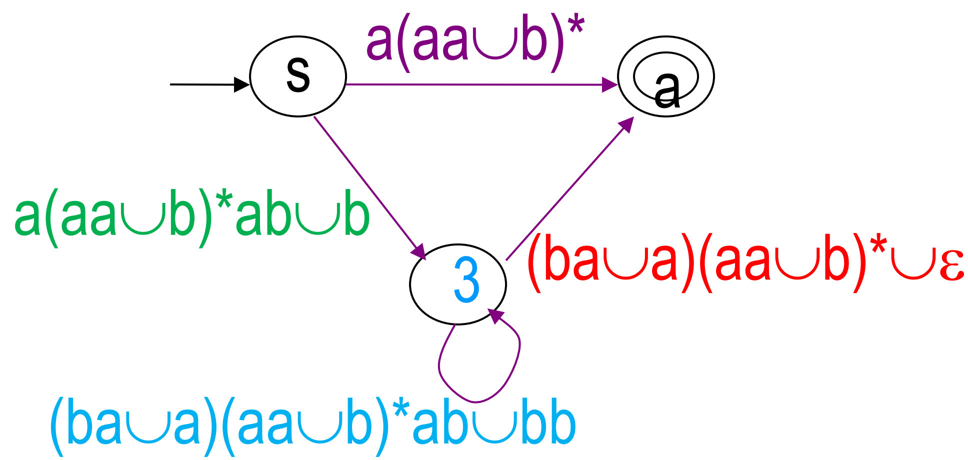
DFA - -> Regular expression



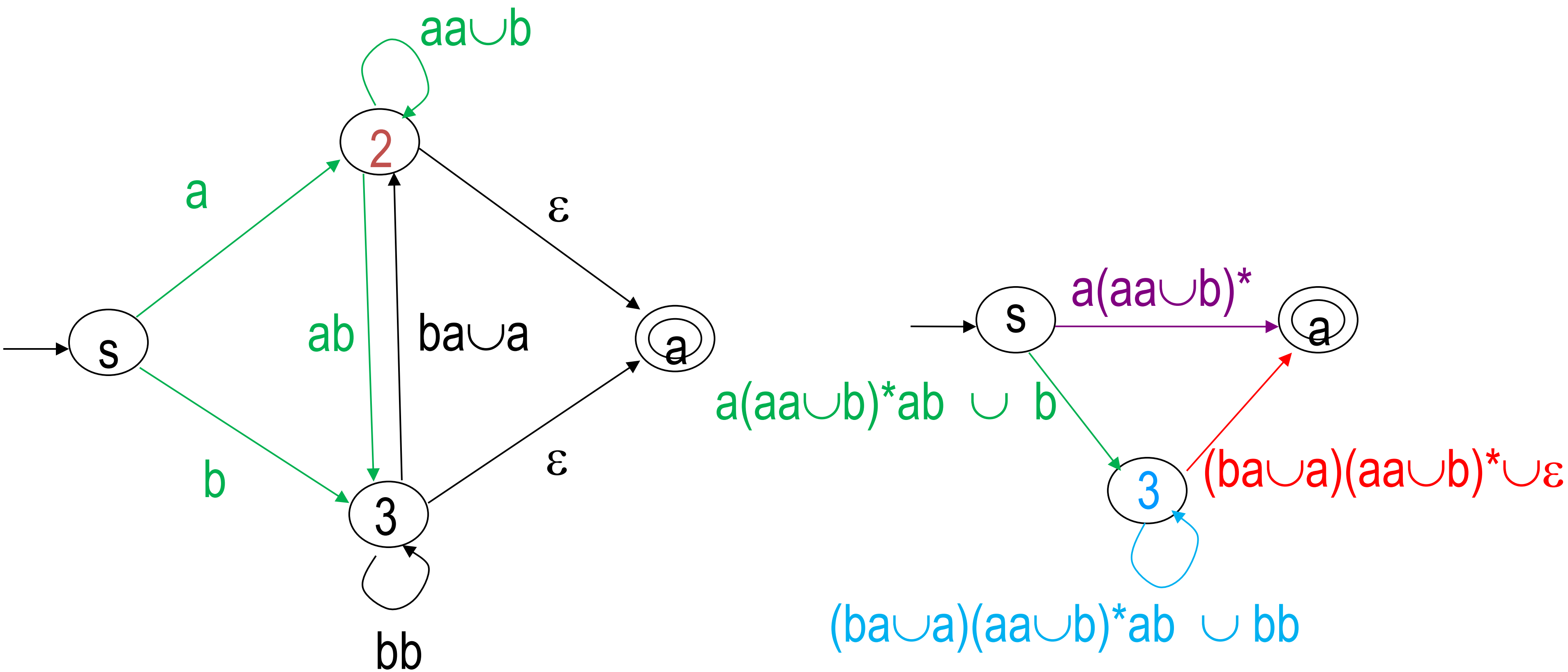


DFA - -> Regular expression

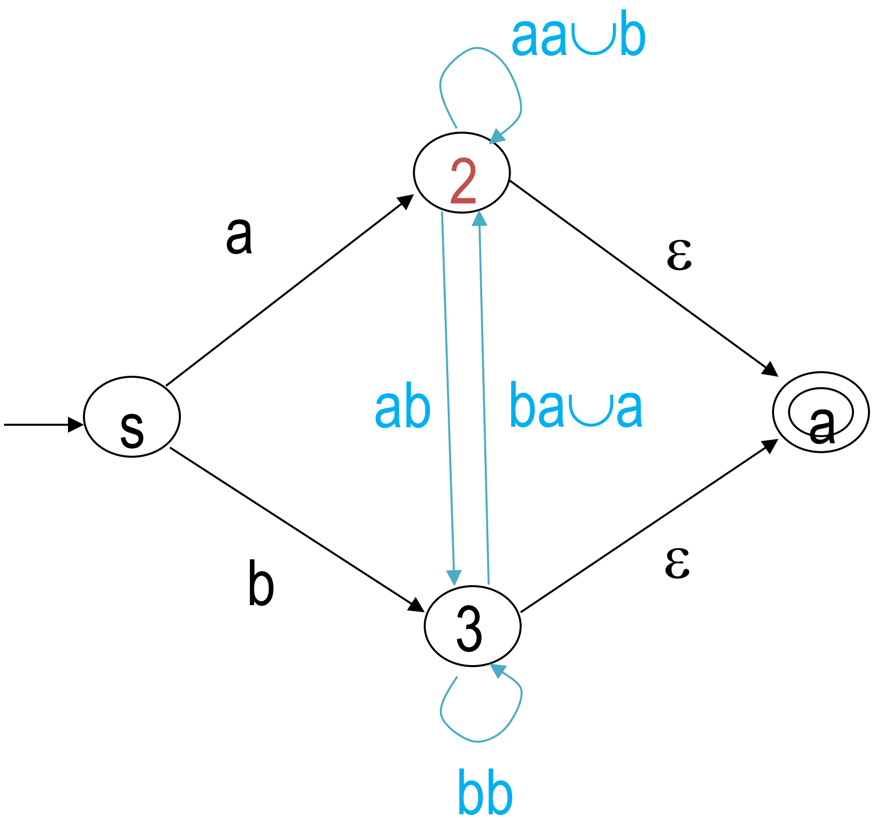


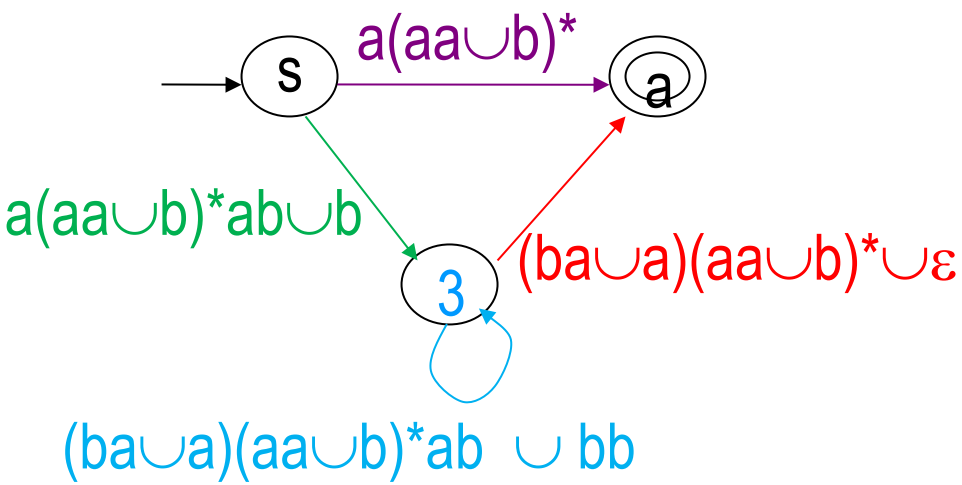


DFA - -> Regular expression

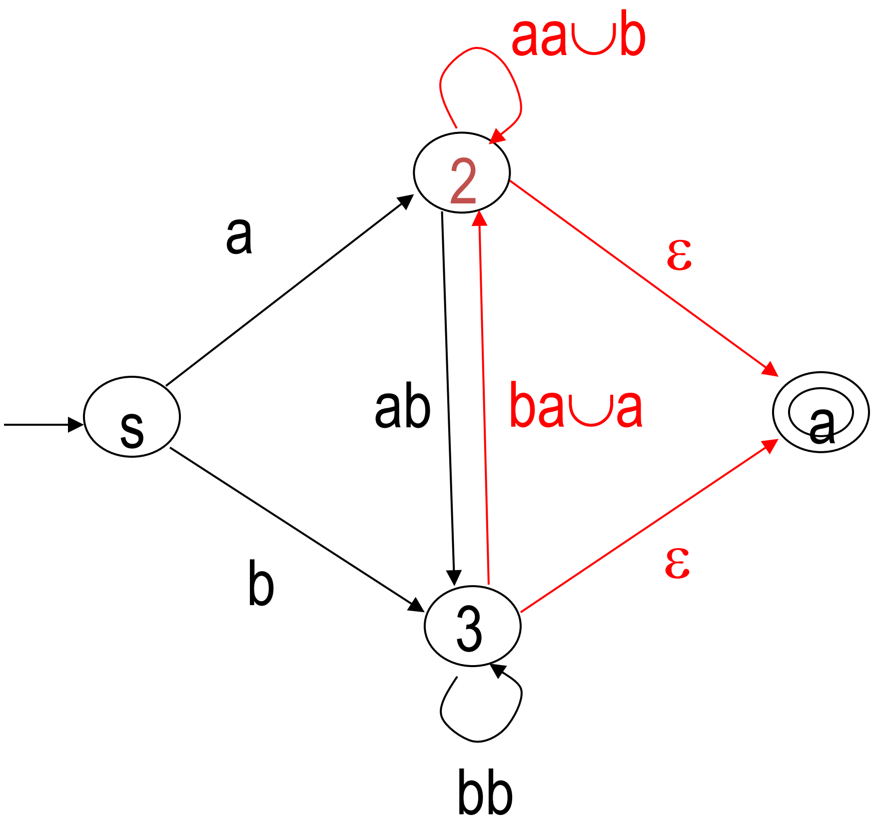


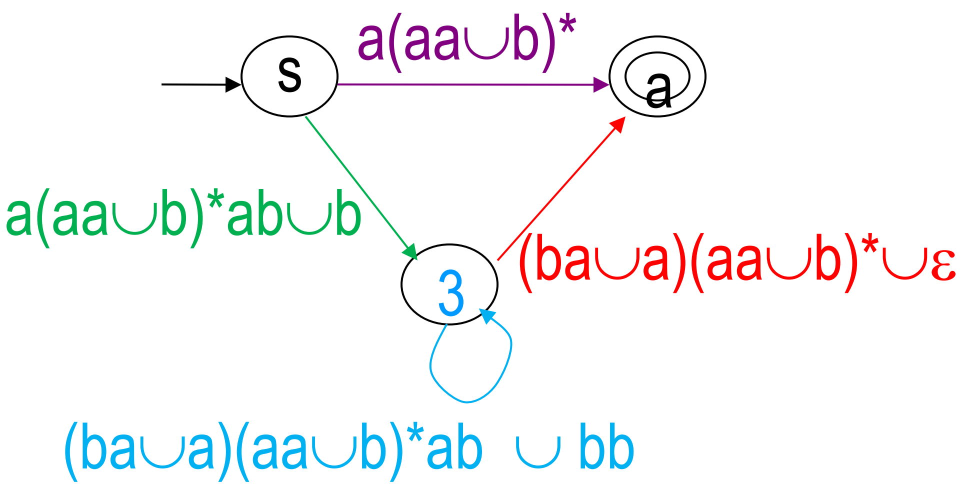
DFA - -> Regular expression



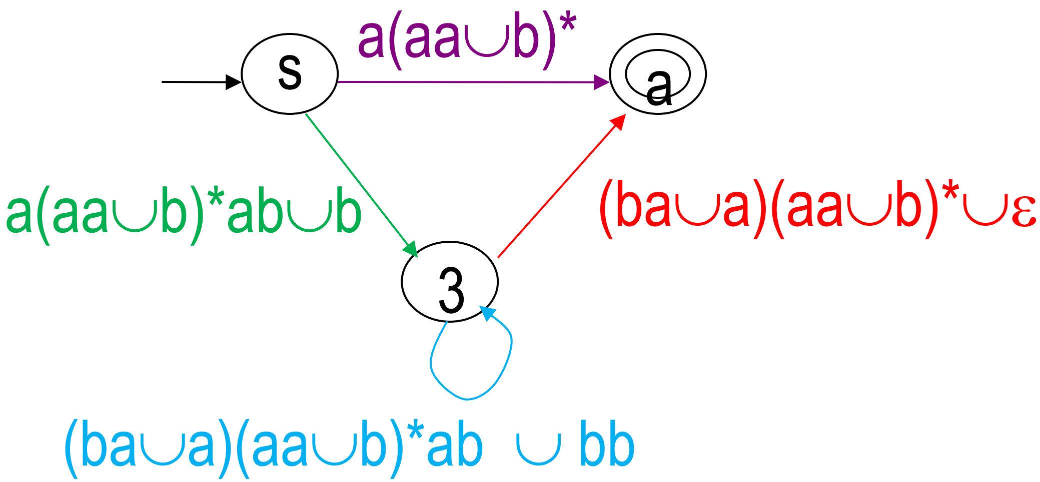


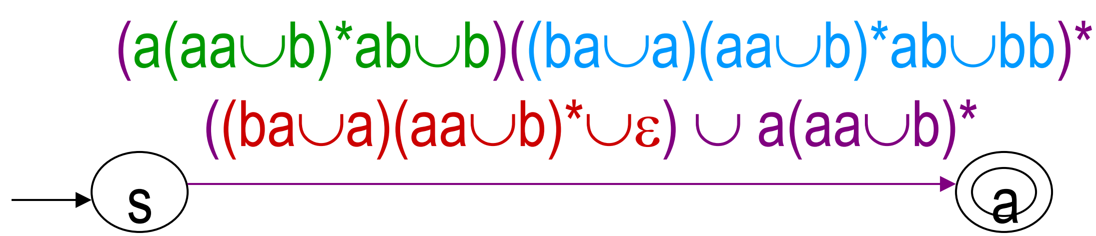
DFA - -> Regular expression



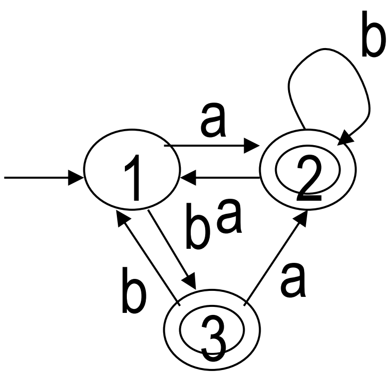


DFA - -> Regular expression

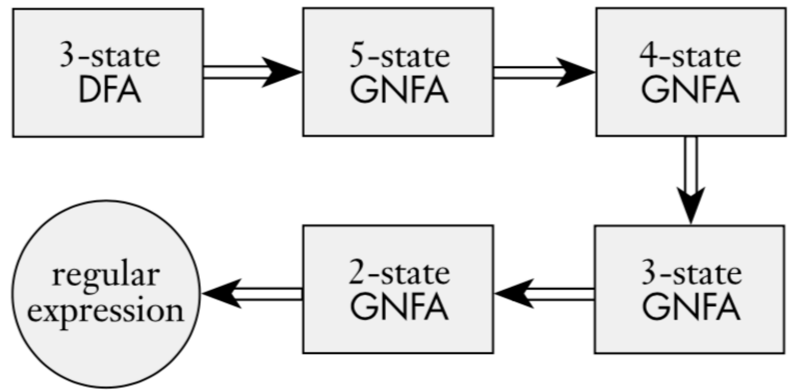


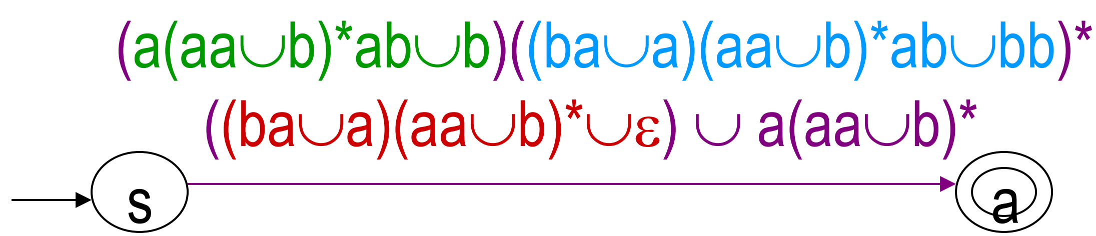


DFA - -> Regular expression



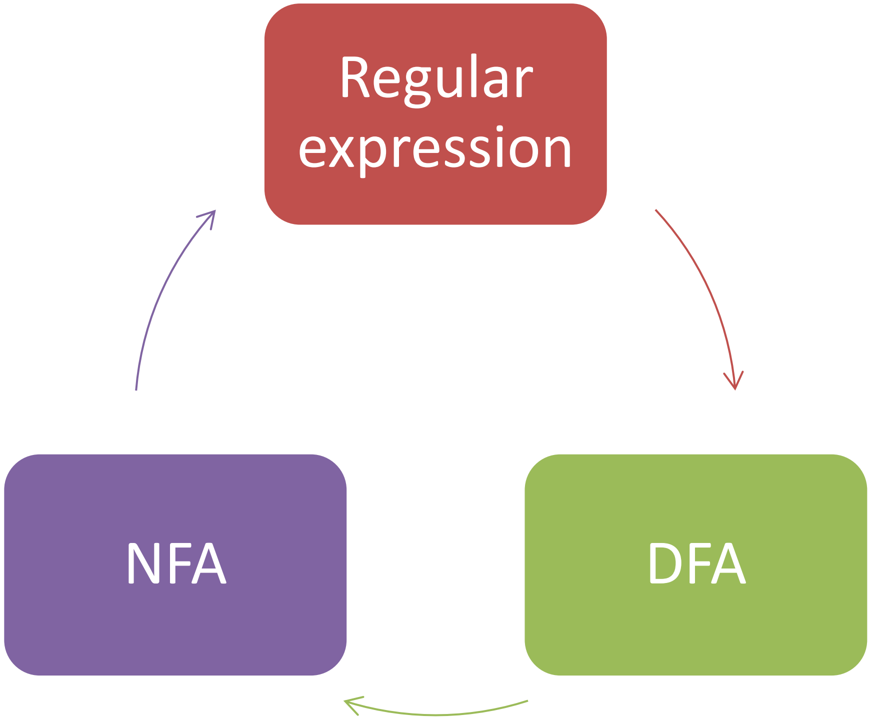
DFA





Regular language: DFA, NFA, Regular expression

- A language is regular if some deterministic finite automaton recognizes it
- A language is regular if and only if some nondeterministic finite automaton recognizes it
- A language is regular if and only if some regular expression describes it



Non-regular languages

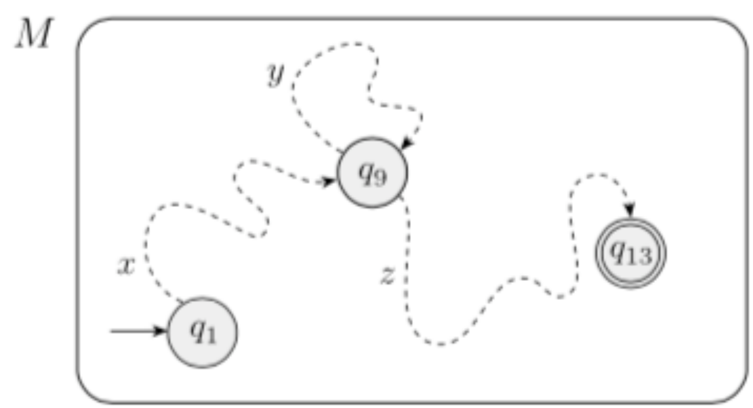
- If a language is regular, we can create a deterministic finite automaton (DFA), or nondeterministic finite automaton (NFA), or regular expression for it

- How to determine a language is nonregular?
  - Pumping lamma

Pumping lemma

- All regular languages have a special property:

- A is RL, then there is a number p (pumping length), where if $s \in A$ and $|s| \geq p$, then s=xyz, satisfying the following:
  1) $\forall i \geq 0$, $xy^iz \in A$;
  2) $|y|>0$;
  3) $|xy| \leq p$.



Pumping lemma example

1) $\forall i \geq 0, xy^iz \in A$;

2) $|y|>0$;

3) $|xy| \leq p$.

- $B = \{0^n1^n \mid n \geq 0\}$ is not regular
- Prove:

Suppose B is regular and p is the pumping length, let $s = 0^p1^p$,

Because $s \in B$ and $|s| > p$,

So $s=xyz = 0^p1^p$ and for each $i \geq 0$, that $xy^iz \in B$

(1) If y only has 0s, then xyyz has more 0s than 1s, so $xyyz \notin B$

(2) If y only has 1s, something happens

(3) If y has 0s and 1s, for xyyz, we will has "1…0" in the substring yy, so $xyyz \notin B$

Contradiction happens. So B is not regular.

Pumping lemma example

1) $\forall i \geq 0, xy^iz \in A$;

2) $|y|>0$;

3) $|xy| \leq p$.

- C = {w| w has an equal number of 0s and 1s} is not regular
- Prove:

Suppose C is regular and p is the pumping length,

let $s = 0^p1^p = xyz$,

Because $s \in C$ and $|s| > p$,

so that each $i \geq 0$, that $xy^iz \in C$ and $|xy| \leq p$

If $|xy| \leq p$, then y only has 0s.

Based on the previous prove in language B, we can get $xyyz \notin C$

Contradiction happens. So C is not regular.

Pumping lemma example

1) $\forall i \geq 0, xy^iz \in A$;

2) $|y|>0$;

3) $|xy| \leq p$.

- Let F = {ww| w ∈ $\{0,1\}^*$}. We show that F is nonregular
- Prove:

Suppose F is regular and p is the pumping length,

let s = 0^p10^p1 = xyz,

Because s∈F and |s|> p, so that each $i \geq 0$,

that xy^iz ∈ F and $|xy| \leq p$

If $|xy| \leq p$, then y only has 0s. Then we can get xyyz ∉ F

Contradiction happens. So F is not regular.

1) $\forall i \geq 0, xy^iz \in A$;

2) $|y|>0$;

3) $|xy| \leq p$.

Pumping lemma example

- $E = \{0^i1^j \mid i > j\}$ is not regular
- Prove:

Suppose E is regular and p is the pumping length,

let $s = 0^{p+1}1^p = xyz$,

Because $s \in F$ and $|s| > p$, so that each $i \geq 0$,

that $xy^iz \in F$ and $|xy| \leq p$

If $|xy| \leq p$, then y only has 0s. We let i=0, then we have xz

Because in s, the number of 0s is only one more than the number of 1s, then in xz, the number of 0s cannot be more than 1s, therefore $xz \notin E$

Contradiction happens. So E is not regular.

Exam 1

- 10 True/False question
  - 2 points each
- 4 short answer question
  - 20 points each
- $100 = 2*10 + 4*20$