CSE 211: Regular Expression

Md. Shaifur Rahman

Dept. of CSE
Bangladesh University of Engineering & Technology

Class 5-6

Regular Expression: Definition

R is a Regular Expression (RE) if R is:

1. a for some $a \in \Sigma$
2. ϵ
3. $\varnothing$
4. $(R_1 \cup R_2)$
5. $(R_1 \circ R_2)$
6. R_1^*

where, R_1 and R_2 are regular expressions

$\varnothing$ and ϵ are not the same!

$\varnothing$: for an RE that generates no string

ϵ: for an RE that generates a string of length zero

RE: Notations

Remember how numbers and operators combine to give arithmatic experssions like $(5 + 3) \times 4$
Similarly set of regular languages and regular operator $\cup, \circ, *$ combine to give regular expressions i.e. new sets of regular languages

Shorthands

- Curly brace for set notations can be omitted
- Concatenation operator $\circ$ can be omitted

Example: 0 or 1 optionally followed by 0's
$\{\{0\} \cup \{1\}\} \circ \{0\}^*$
$\Rightarrow (0 \cup 1)0^*$

RE: Notations(Contd.)

Some notations that might seem confusing:

- $(0 \cup \epsilon)1^* = 01^* \cup 1^*$ Remember, $0 \cup \epsilon \neq \{0\}$, but $0 \cup \varnothing = \{0\}$
- $(0 \cup \epsilon) \circ (1 \cup \epsilon) = \{\epsilon, 0, 1, 01\}$, but NOT $\{01\}$
- Concatenating the empty set to any set yields the empty set: $1^*\varnothing = \varnothing$
- $\varnothing^* = \{\epsilon\}$
- Adding the empty language to any other language will not change it: $R \cup \varnothing = R$
- Joining the empty string to any string will not change it: $R \circ \epsilon = R$
- $R \cup \epsilon$ may not equal R
- $R \circ \varnothing$ may not equal R

RE: Examples

The alphabet $\Sigma = \{0, 1\}$

- $0^*10^* = \{w|\ w$ contains a single $1\}$
- $\Sigma^*1\Sigma^* = \{w|\ w$ has at least one $1\}$
- $\Sigma^*001\Sigma^* = \{w|\ w$ contains the string 001 as a substring $\}$
- $(\Sigma\Sigma)^* = \{w|\ w$ is a string of even length$\}$
- $(\Sigma\Sigma\Sigma)^* = \{w|$ the length of w is a multiple of $3\}$
- $0\Sigma^*0 = \{w|\ w$ starts and ends with $0\}$
- $1\Sigma^*1 = \{w|\ w$ starts and ends with $1\}$
- $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1 = \{w|\ w$ starts and ends with the same symbol$\}$

RE: Proof

Theorm 1.54

A language is regular if and only if some regular expression describes it.

Two directions for proof:

Lemma 1.55

If a language is described by a regular expression, then it is regular.

And,

Lemma 1.60

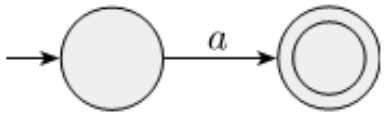
If a language is regular, then it is described by a regular expression.

RE: Equivalence with FA

If a language is described by a regular expression, then it is regular.

Proof: Convert R.E R into an NFA N. Six cases to consider from the definition of R.E.

Case 1: $R = a$ for some $a \in \Sigma$



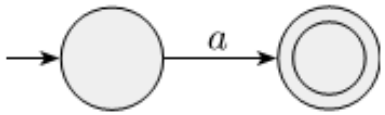
Here, $N = (\{q_1, q_2\}, \Sigma, \delta, q_1, \{q_2\})$ where $\delta(q_1, a) = q_2$ and $\delta(r, b) = \varnothing$ where $r \neq q_1$ and $b \neq a$

RE: Equivalence with FA

If a language is described by a regular expression, then it is regular.

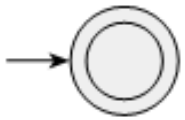
Proof: Convert R.E R into an NFA N. Six cases to consider from the definition of R.E.

Case 1: $R = a$ for some $a \in \Sigma$



Here, $N = (\{q_1, q_2\}, \Sigma, \delta, q_1, \{q_2\})$ where $\delta(q_1, a) = q_2$ and $\delta(r, b) = \varnothing$ where $r \neq q_1$ and $b \neq a$

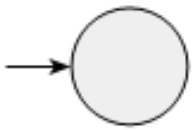
Case 2: $R = \epsilon$



Here, $N = (\{q_1\}, \Sigma, \delta, q_1, \{q_1\})$ where $\delta(r, b) = \varnothing$ for any r and b

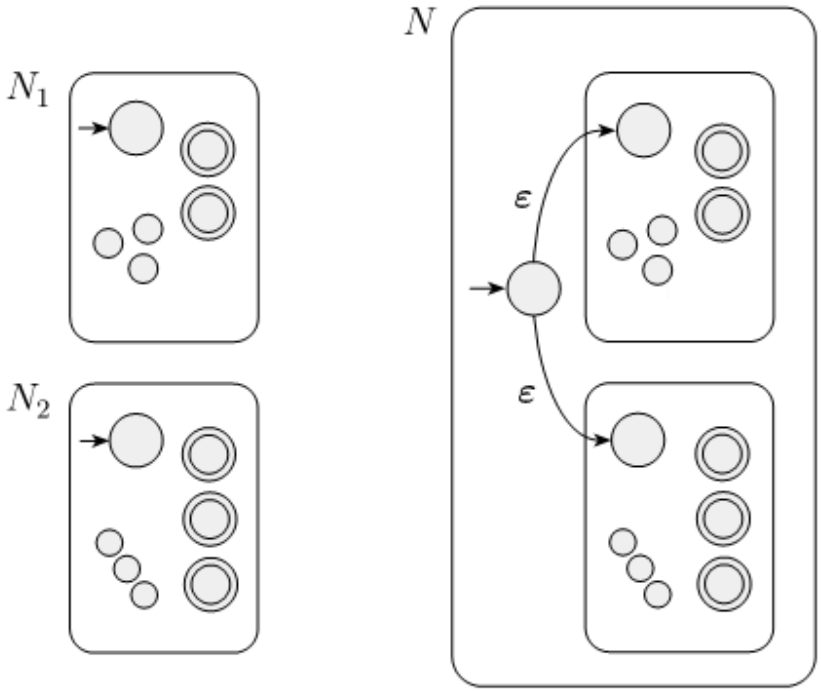
RE: Equivalence with FA (Contd.)

Case 3: $R = \varnothing$



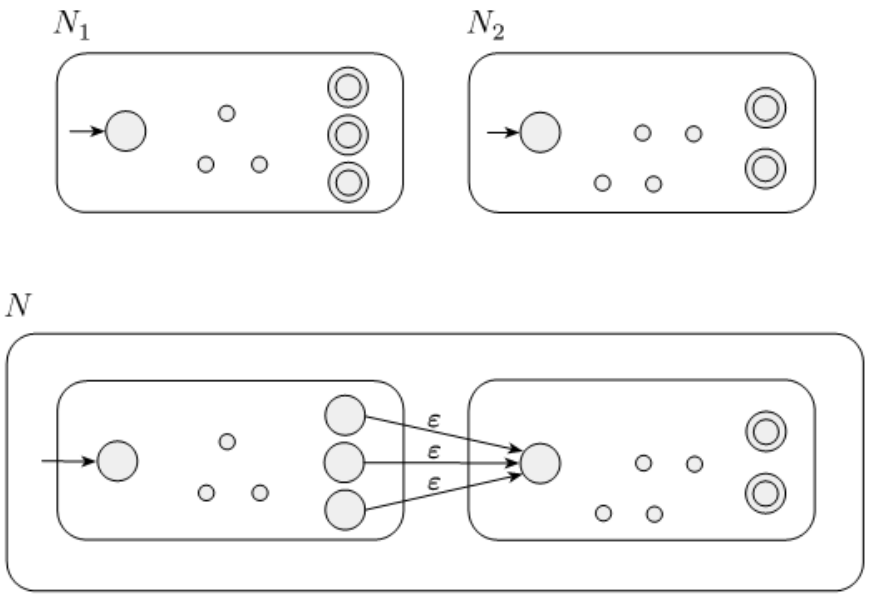
Here, $N = (\{q_1\}, \Sigma, \delta, q_1, \varnothing)$ where $\delta(r, b) = \varnothing$ for any r and b

Case 4: $R = R_1 \cup R_2$, (definition of automaton, try yourself!)

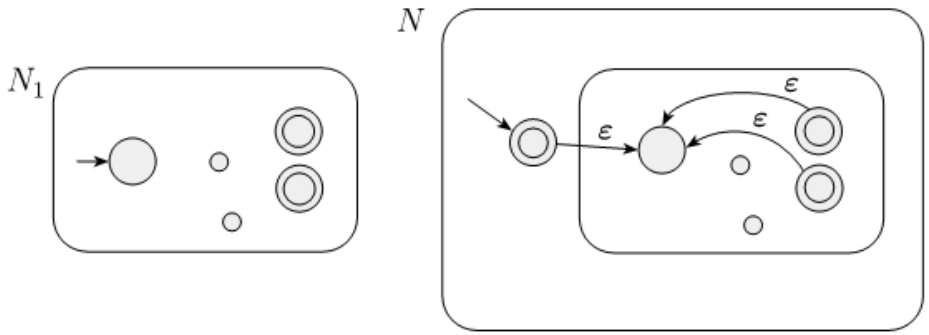


RE: Equivalence with FA (Contd.)

Case 5: $R = R_1 \circ R_2$, (definition of automaton, try yourself!)

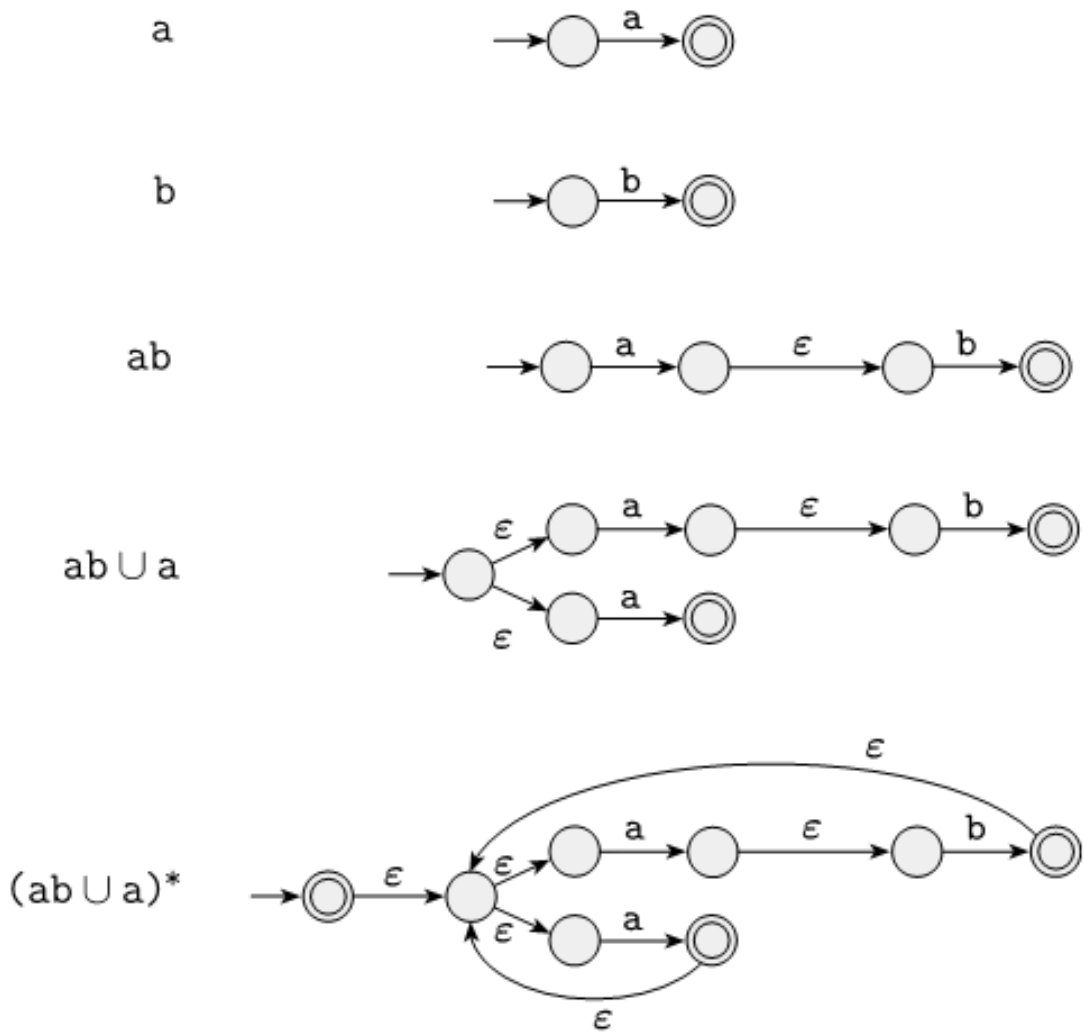


Case 6: $R = R_1^*$, (definition of automaton, try yourself!)



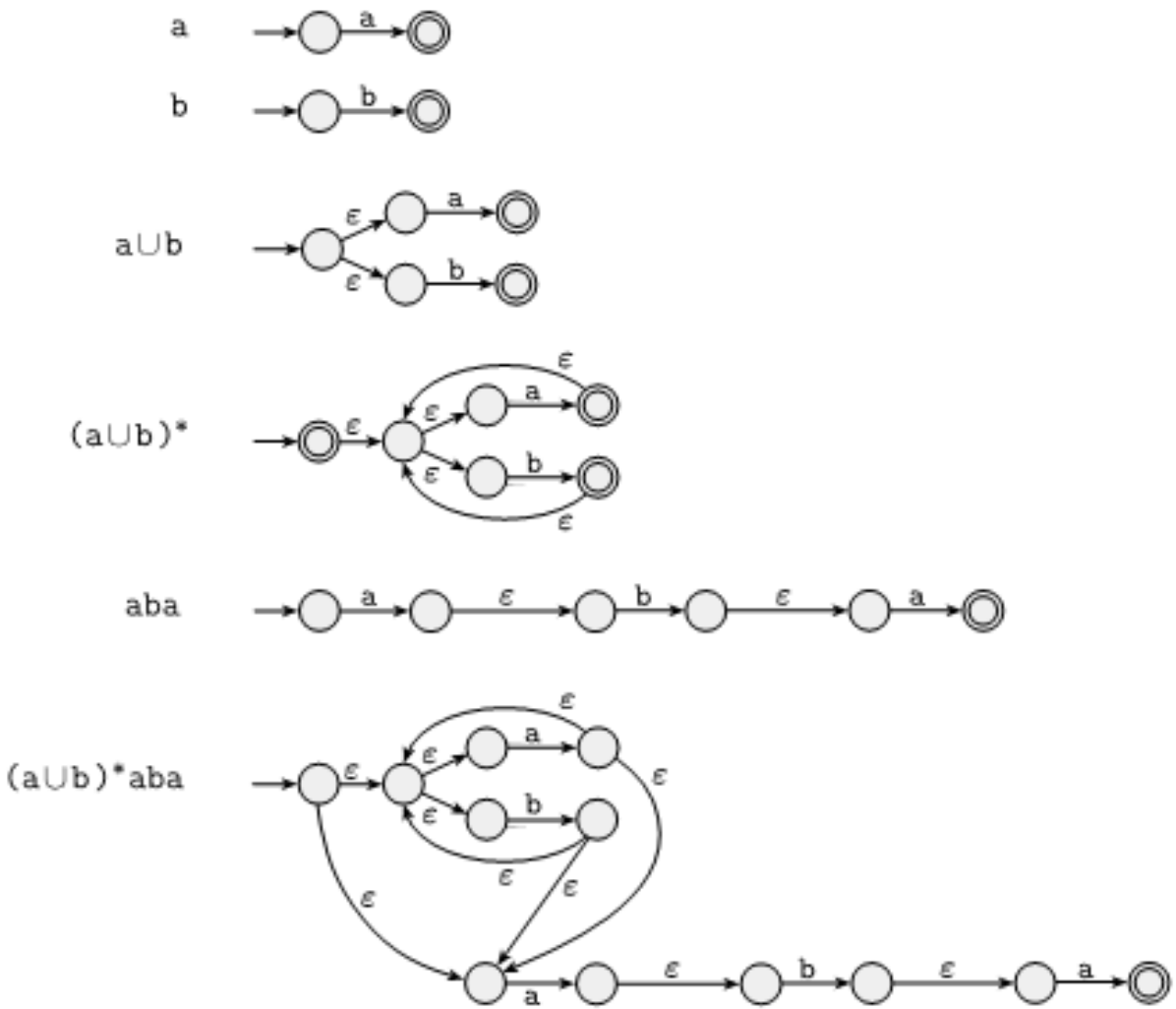
Example of Converting RE into NFA

Convert the RE $(ab \cup a)^*$ into an NFA



Example of Converting RE into NFA (Contd.)

Convert the RE $(a \cup b)^* aba$ into an NFA



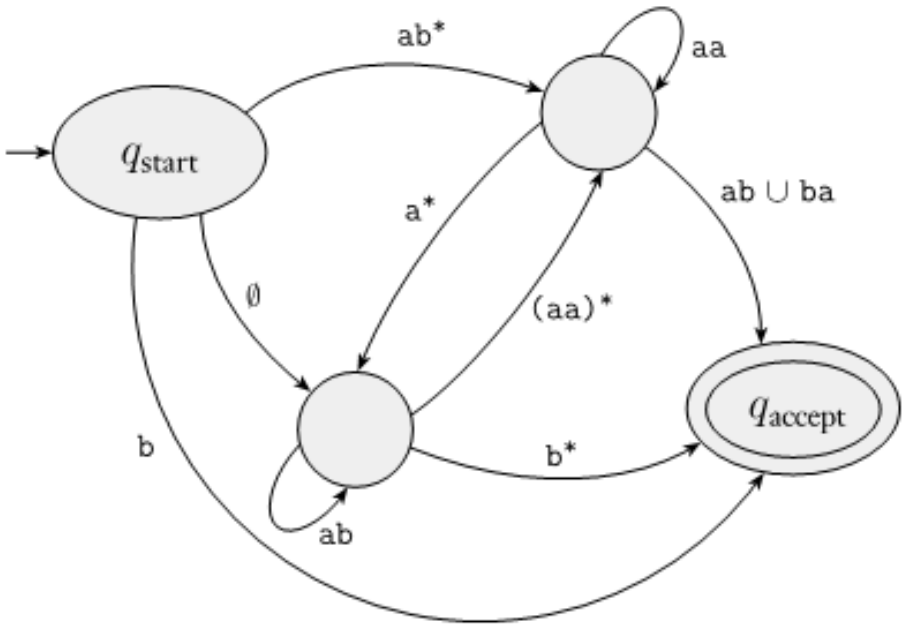
RE: Equivalence with FA (Contd.)

If a language is regular, then it is described by a regular expression.

Proof Technique:

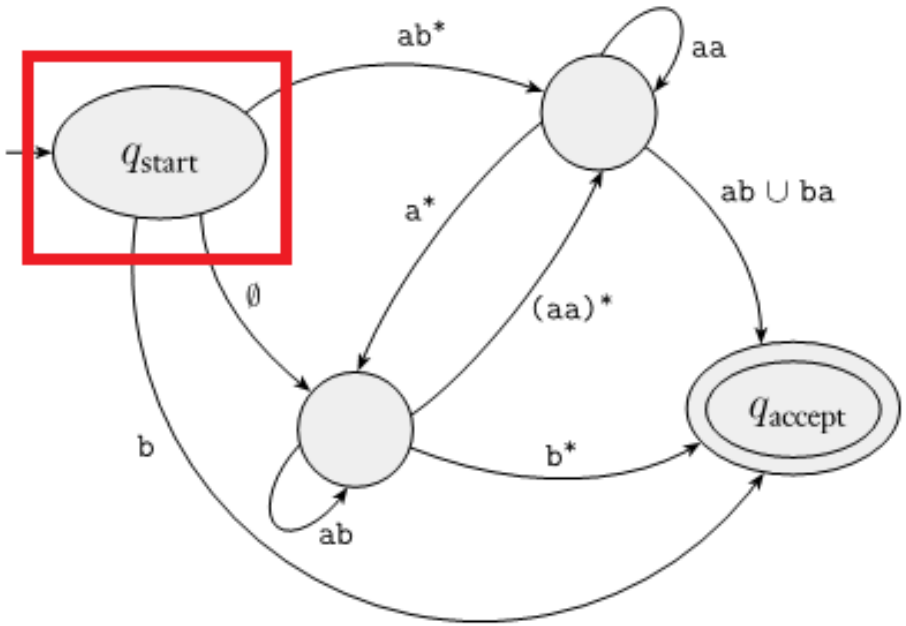
- We convert a given DFA into a Generalized Non-deterministic Finite Automaton (GNFA)
- Then, we generate RE from the GNFA

RE: Equivalence with FA (Contd.)



Characteristic of GNFA

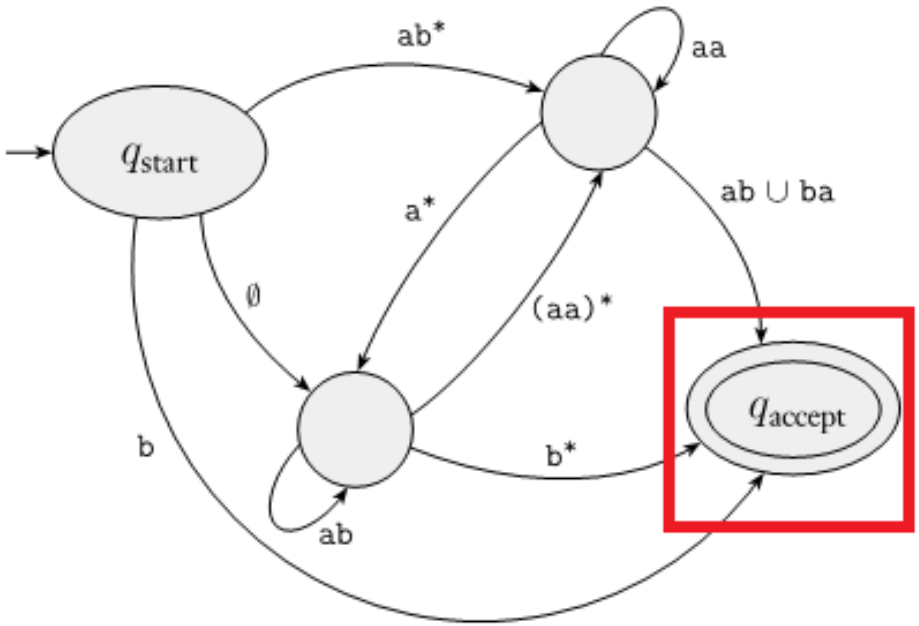
RE: Equivalence with FA (Contd.)



Characteristic of GNFA

- Start state has outgoing arrow to every other state but no incoming arrow

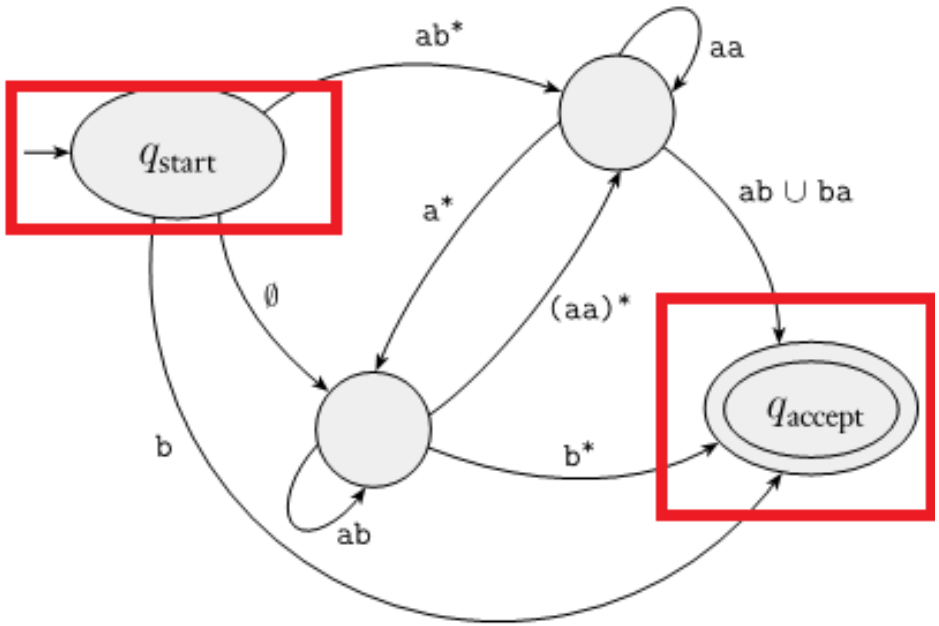
RE: Equivalence with FA (Contd.)



Characteristic of GNFA

- Start state has outgoing arrow to every other state but no incoming arrow
- Final State has incoming arrows from every other state but no outgoing arrow

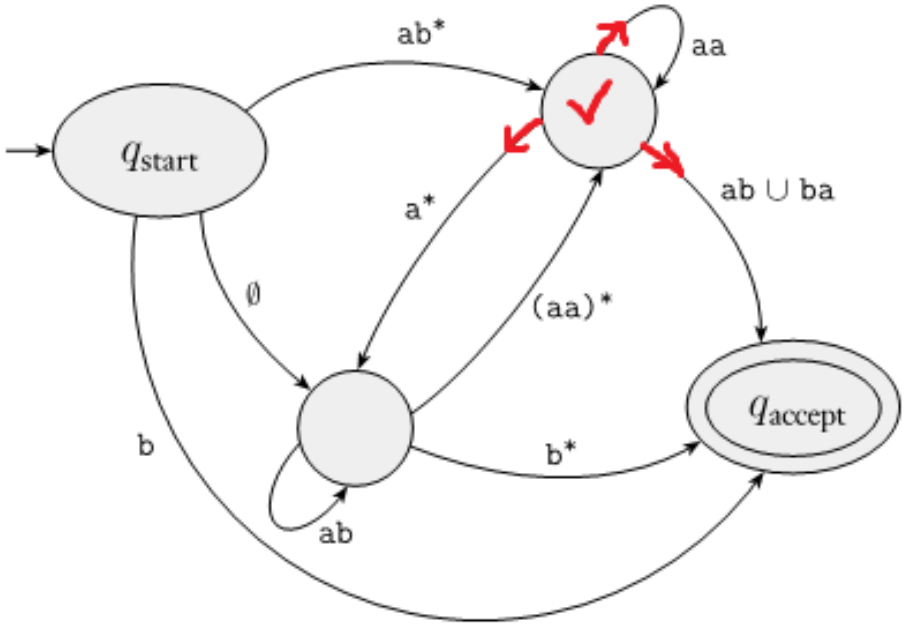
RE: Equivalence with FA (Contd.)



Characteristic of GNFA

- Start state has outgoing arrow to every other state but no incoming arrow
- Final State has incoming arrows from every other state but no outgoing arrow
- Only one Start and one Final state; Start state $\neq$ Final state

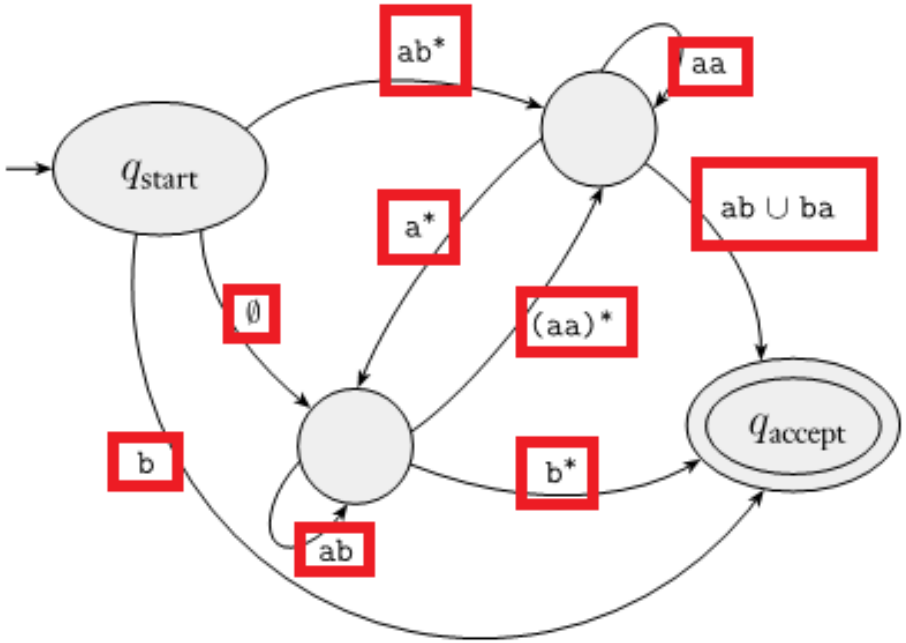
RE: Equivalence with FA (Contd.)



Characteristic of GNFA

- One arrow goes from every state to every other state (except the Start state) and also from each state (except the Final state) to itself

RE: Equivalence with FA (Contd.)

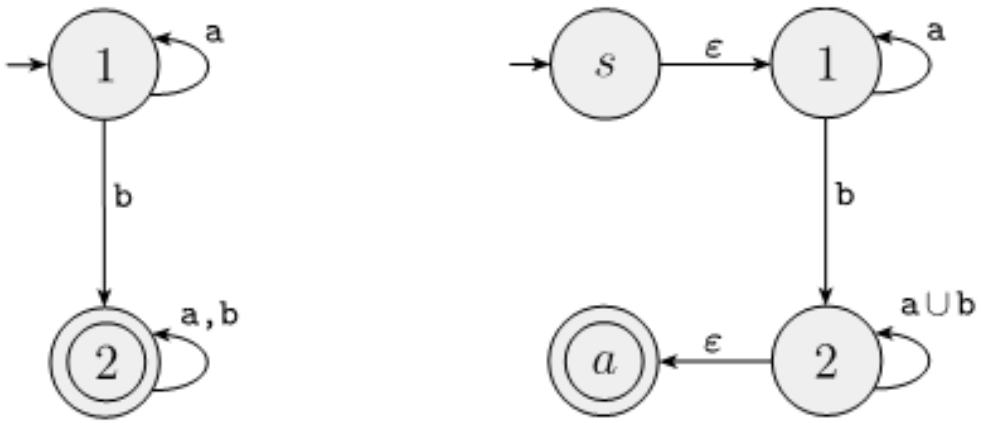


Characteristic of GNFA

- One arrow goes from every state to every other state (except the Start state) and also from each state (except the Final state) to itself
- The transition arrows may have any regular expressions as labels

RE: Equivalence with FA (Contd.)

DFA $\Rightarrow$ GNFA:



- Add a new Start state with an ϵ arrow to the old Start state
- Add a new Final state with ϵ arrows from the old Final states

RE: Equivalence with FA (Contd.)

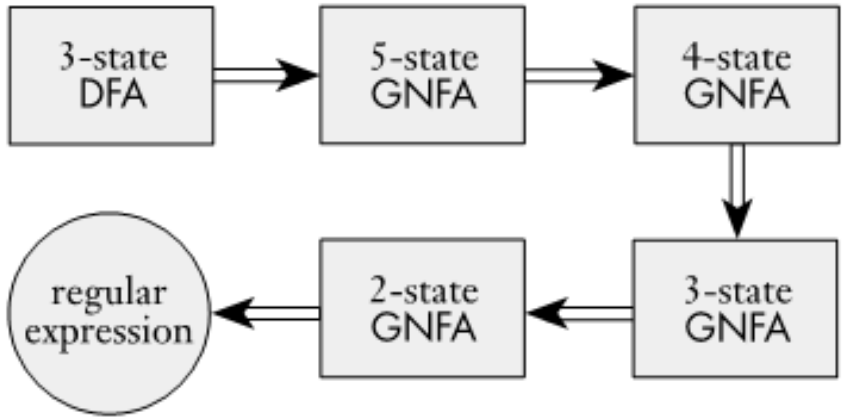
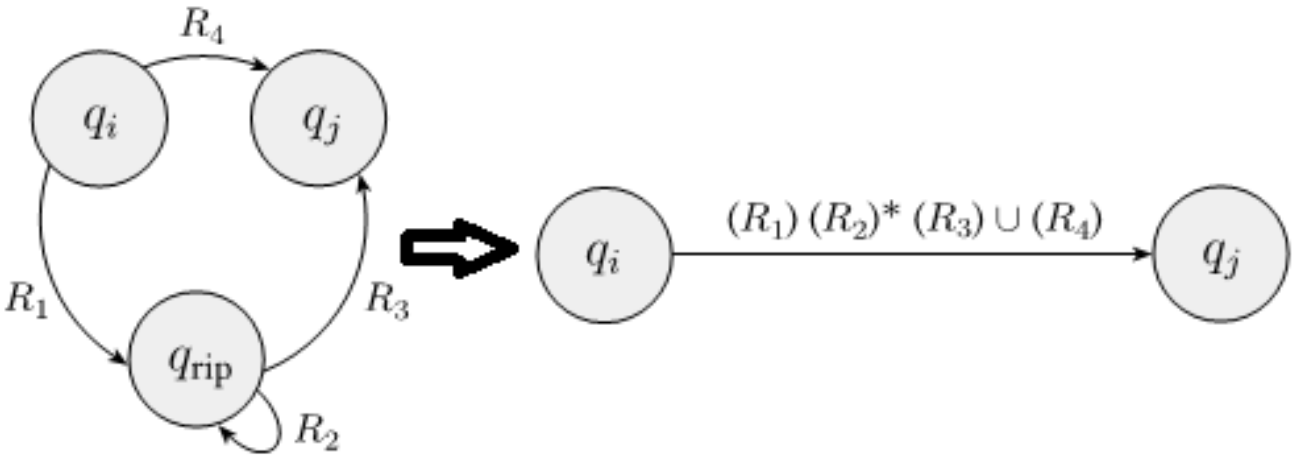


Figure : 3-state DFA to 2-state GNFA and Finally, RE

- k-state DFA $\Rightarrow$ $(k+2)$-state GNFA
- $(k+2)$-state GNFA $\Rightarrow$ $(k+1)$-state GNFA $\Rightarrow \ldots \Rightarrow$ 2-state GNFA
- 2-state GNFA's arrow label is the final RE

RE: Equivalence with FA (Contd.)



In old GNFA, state q_{rip} is to be dropped. If:

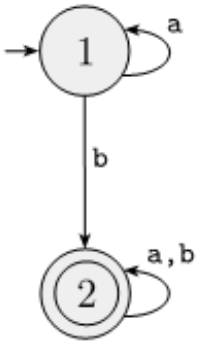
1. q_i goes to q_{rip} with arrow label R_1
2. q_{rip} goes to q_{rip} with arrow label R_2
3. q_{rip} goes to q_3 with arrow label R_3
4. q_i goes to q_j with arrow label R_4

then in the new GNFA, arrow from q_i to q_j gets a label

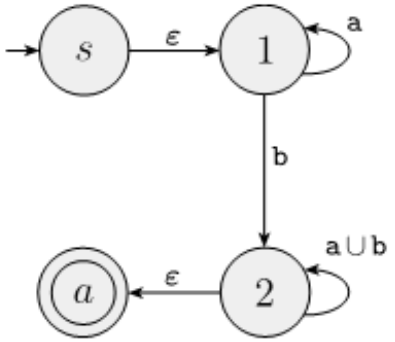
$$(R_1)(R_2)^*(R_3) \cup (R_4)$$

RE: Equivalence with FA (Contd.)

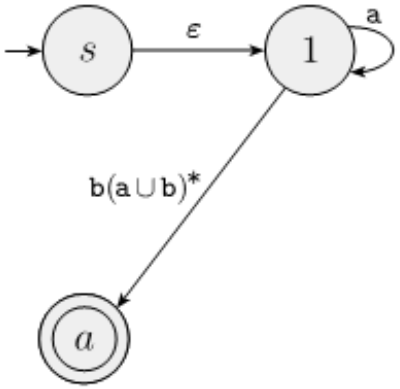
Find the RE for the DFA given in Figure(a).



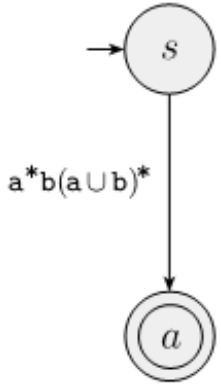
(a)



(b)



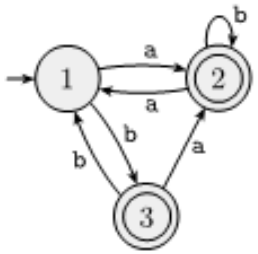
(c)



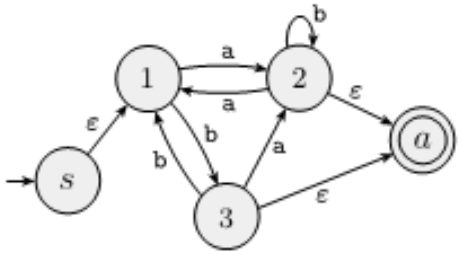
(d)

RE: Equivalence with FA (Contd.)

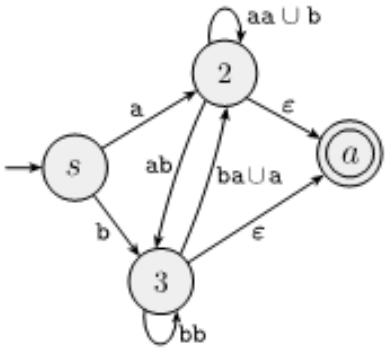
Find the RE for the DFA given in Figure(a).



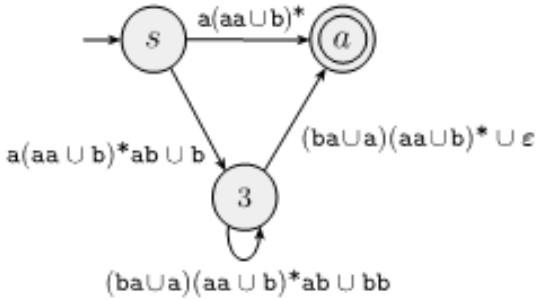
(a)



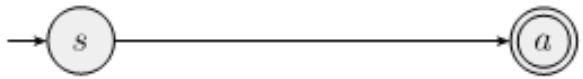
(b)



(c)



(d)



$(a(aa \cup b)^*ab \cup b)((ba \cup a)(aa \cup b)^*ab \cup bb)^*((ba \cup a)(aa \cup b)^* \cup \varepsilon) \cup a(aa \cup b)^*$

RE: Equivalence with FA (Contd.)

If a language is regular, then it is described by a regular expression.

We convert DFA into a k-state GNFA. Now, we prove that:

All k-state GNFA are equivalent to each other for $k = k, k-1, \ldots, 2$

Proof Technique: Proof by induction on the number of states k
Basis: True for $k = 2$.
Induction Step: Prove that if true for $(k-1)$ state, then true for k state.

A Caveat on Proof by Induction

Is there anything wrong with the following proof??

All horses have the same color.

Proof by Induction on the number of horses:

Basis: True for $k = 1$

Induction Step: Assume that n horses always are the same color. Let us consider a group consisting of $n + 1$ horses. First, exclude the last horse and look only at the first n horses; all these are the same color since n horses always are the same color.

Likewise, exclude the first horse and look only at the last n horses. These too, must also be of the same color.

Therefore, the first horse in the group is of the same color as the horses in the middle, who in turn are of the same color as the last horse. Thus, we have proven that:

If n horses have the same color, then $n + 1$ horses will also have the same color.

Question?