San José State University
Department of Computer Science

Ahmad Yazdankhah
ahmad.yazdankhah@sjsu.edu
www.cs.sjsu.edu/~yazdankhah

Nondeterministic Finite Automata

(Part 1)

Lecture 09

Day 09/31

CS 154

Formal Languages and Computability

String 2018

Agenda of Day 09

- About Midterm 1
- Summary of Lecture 08
- Quiz 3
- Lecture 09: Teaching ...
  - Nondeterministic Finite Automata (Part 1)

About Midterm 1

1st Announcement

- Midterm #1 (aka Quiz+)
  - Date: Thursday 03/01
  - Value: 10%
  - Topics: Everything covered from the beginning of the semester
  - Type: Closed y ∈ Material

    Material = {Book, Notes, Electronic Devices, Chat, ... }

- The cutoff for midterm #1 is the end of this lecture.

Summary of Lecture 08: We learned ...

Deterministic Finite Automata (DFA)

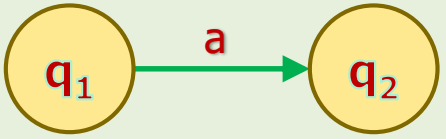
- A deterministic finite Automata (DFA) M is defined by a quintuple:

$$M = (Q, \Sigma, \delta, q_0, F)$$

- Q is ...
  - ... a finite and nonempty set of states of the transition graph.
- Σ is ...
  - ... a finite and nonempty set of symbols called input alphabet.
- δ is ...
  - ... called transition function and is defined as: $\delta: Q \times \Sigma \rightarrow Q$

δ must be total function.

- $q_0 \in Q$ is ...
  - ... the initial state of the transition graph.
- $F \subseteq Q$ is ...
  - ... the set of accepting states of the transition graph.
- Every sub-rule like $\delta(q_1, a) = q_2$ in transition graph represents a transition.



Any question?

Summary of Lecture 08: We learned ...

Deterministic Finite Automata

- Why total function?
  - ... because in some situations, the DFA does not know where to go!
- DFAs constraint ...
  - ... DFAs transition function must be total function.
- The consequence of this constraint is ...
- ... every state must have an outgoing transition for every symbol of alphabet.

- Associated language to a DFA is ...
  - ... the set of all strings that it accepts.
  - ... denoted by L(M).
- Two machines are equivalent iff ...
  - ... their associated languages are the same.

What is wrong with the following definition?

Two machines are equivalent iff both accept the same language.

Any question?

Summary of Lecture 08: We learned ...

Deterministic Finite Automata

- Computation is ...
  - ... the sequence of configurations from when the machine starts until it halts.
- Deterministic automaton ...
- A machine is called deterministic iff during any timeframe, there is NO MORE THAN ONE transition.

Any question?

NAME	Alan M. Turing		
SUBJECT	CS 154	TEST NO.	3
DATE	02/22/2018	PERIOD	1 , 2 , 3

TEST RECORD	
PART 1	123
PART 2	
TOTAL	

Quiz 3

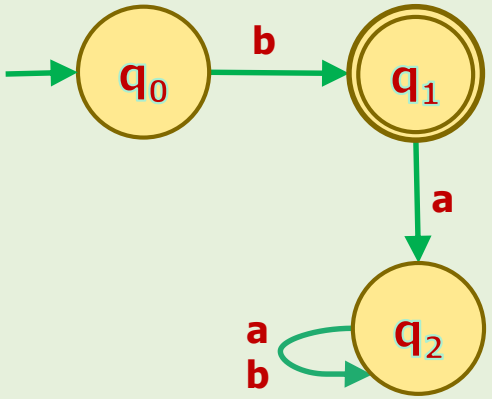
Use Scantron

Nondeterministic Finite Automata (NFA)

DFAs Constraint Violations

DFAs Constraint Violation #1

- What is the problem of the following machine? Σ = {a, b}



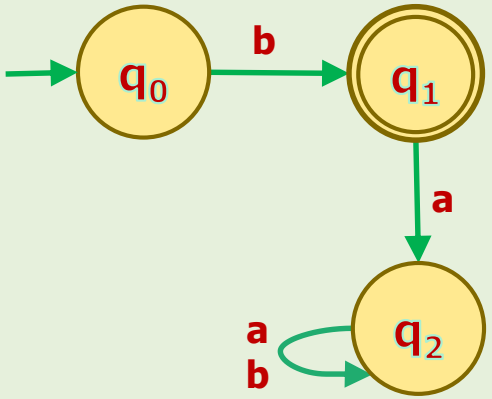
Violation

- The machine has no (zero) transition if it is in state q_0 and the input is a!
  - There is more like this in this graph, find it as exercise!

- In other words, there are some timeframes that the machine does NOT know where to go.

DFAs Constraint Violation #1

- What is the value of $\delta(q_0, a)$?
  - $\delta(q_0, a)$ = "Undefined"

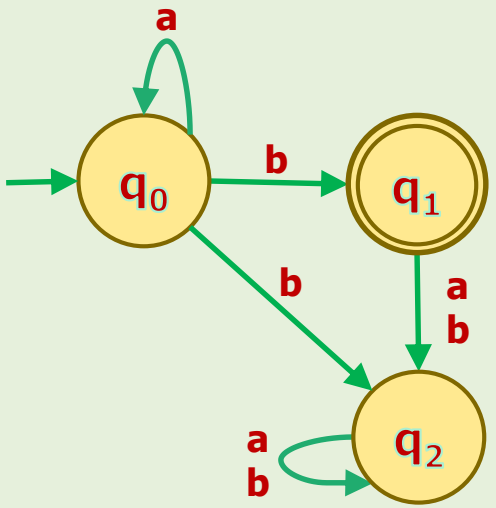


- What type of function is the transition function of this machine?
  - "Partial function"

- So, the machine is not DFA because it violates the DFA constraint!

DFAs Constraint Violation #2

- What is the problem of the following machine? Σ = {a, b}



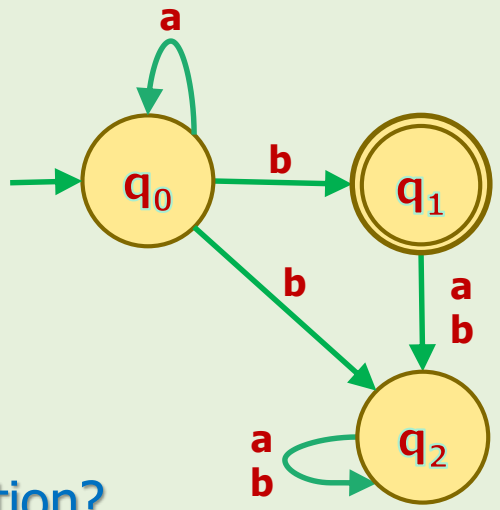
Violation

- The machine has more than one transition if it is in state q_0 and the input is b!

- In other words, there are some timeframes that the machine does NOT know where to go.
  - Because there are more than one choice!

DFAs Constraint Violation #2

- What is the value of $\delta(q_0, b)$?
  - $\delta(q_0, b) = \{q_1, q_2\}$
  - Note that the range is a set of Qs.



- What type of function is the transition function?
  - It is NOT a function because it violates the definition of function.
  - It is called "multifunction" (aka multivalued function) in math.

- So, the machine is not DFA because again, it violates the DFA constraint!

DFAs Constraint Violations Summary

- **Violation #1**: There are some timeframes that the machine has no (zero) transition.
  - The transition function is NOT total function.

- **Violation #2**: There are some timeframes that the machine has more than one transition.
  - The transition function is NOT a function.
  - It is multifunction (or multivalued function).

- Let's relax the DFAs constraint and define a new class of machines!

Let's Relax the DFAs Constraint!

- Recall that DFAs' constraint is:

$$\delta: Q \times \Sigma \rightarrow Q$$

δ must be a total function.

- To accommodate those two violations, we need to change the type of δ to "multifunction".
- In this way, the range can be zero, one, or more states.
- In other words, the range of this function is a set of Qs.
- We already know that 2^Q is the power set of Q and it contains all subsets of Q.
- Therefore, we change the range from Q to 2^Q.

$$\delta: Q \times \Sigma \rightarrow 2^Q$$

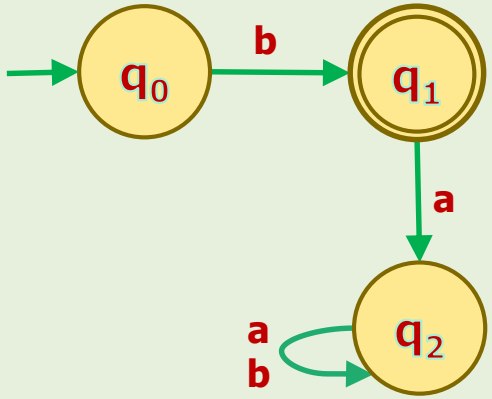
- Let's take some examples.

Relaxed δ Function Examples

Example 1

- Write the rule of the following transition graph over Σ = {a, b}.

- δ: $\begin{cases} \delta(q_0, a) = \{\} \\ \delta(q_0, b) = \{q_1\} \\ \delta(q_1, a) = \{q_2\} \\ \delta(q_1, b) = \{\} \\ \delta(q_2, a) = \{q_2\} \\ \delta(q_2, b) = \{q_2\} \end{cases}$

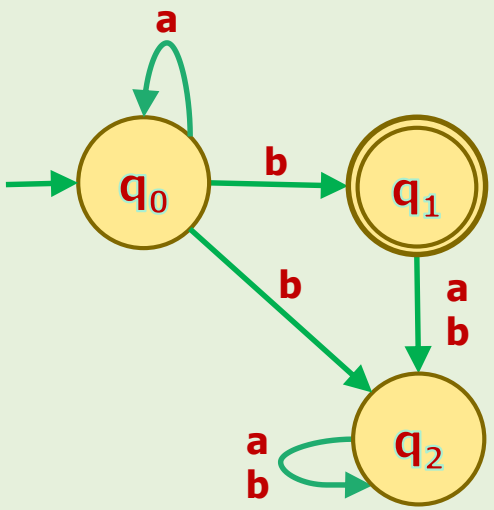


Relaxed δ Function Examples

Example 2

- Write the rule of the following transition graph over Σ = {a, b}.

- δ:
$$\begin{cases} \delta(q_0, a) = \{q_0\} \\ \delta(q_0, b) = \{q_1, q_2\} \\ \delta(q_1, a) = \{q_2\} \\ \delta(q_1, b) = \{q_2\} \\ \delta(q_2, a) = \{q_2\} \\ \delta(q_2, b) = \{q_2\} \end{cases}$$



Let's Construct a New Class of Automata

Template for Constructing a New Class of Automata

- To construct a new class of automata, we need to respond the following questions:
  1. Why do we need a new class of machines? (Justification)
  2. Name of the new class
  3. Building blocks of the new class
  4. How they work
     1. What is the "starting configuration"?
     2. What would happen during a timeframe?
     3. When would the machines halt (stop)?
     4. How would a string be Accepted/Rejected?
  5. Formal Definition
  6. Their power: this class versus previous class
  7. What would be the next possible class?

Why We Need a New Class

- The goal of introducing a new class is always having "more powerful" machines.
  - To understand the meaning of "power", we need more knowledge about formal languages.
  - So, we should wait until then.

- For now, let's claim that ...
- ... we relaxed the DFAs constraint to have simpler transition graph.

- We'll see this fact shortly.

Name of the New Class

- To figure out what to call this new class, let's review its features.

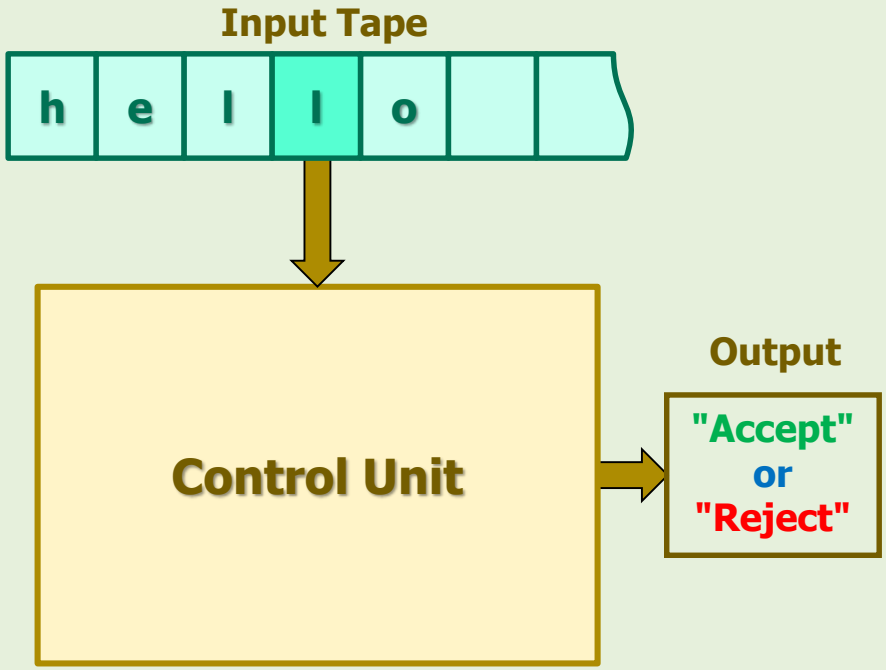
1. The second violation violates the definition of determinism.
   - In other words, during any timeframe, there might be more than one transitions.
   - So, it should be "nondeterministic".
2. The number of states is still "finite".

- Therefore, this new class automata is called:

"Nondeterministic Finite Automata (NFA)"

NFAs Building Blocks

- NFAs have the same building blocks as DFAs:
  1. Input Tape
  2. Control unit
  3. Output



- As usual, we don't need to show the output.

How NFAs Work

How NFAs Work

- To understand the operations of NFAs (and any other class of machines), we should clearly respond to the following questions:

1. What is the "starting configuration"?
2. What would happen during a timeframe?
3. When would the machine halt (stop)?
4. How would a string be Accepted/Rejected?

- The starting configuration of NFAs is the same as DFAs'.

- So, we need to respond to the other three questions.

How NFAs Work

- DFAs' and NFAs' have the same building blocks.
- So, we expect their behavior be the same ...
- … **except for those two violations**.

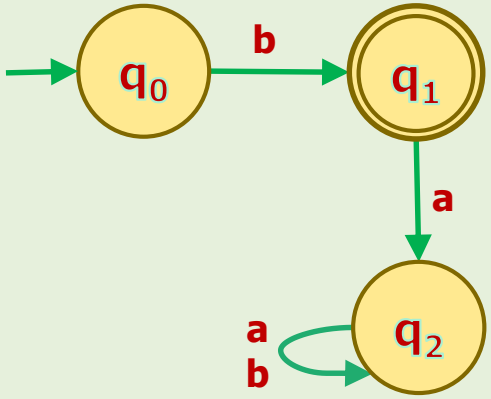
- Therefore, it makes sense if we know what NFAs would do when they encounter those two violations.

- Also we should know what would be the effect(s) of those violations

NFAs' Behavior For the Violation #1

Violation #1

- There are some timeframes that the machine has zero transition.
  - e.g.: $\delta(q_0, a) = \{\}$



NFAs' Behavior

- They halt. ≡ **h**

- So, NFAs halt iff:

  All input symbols are consumed. ≡ **c**

  OR

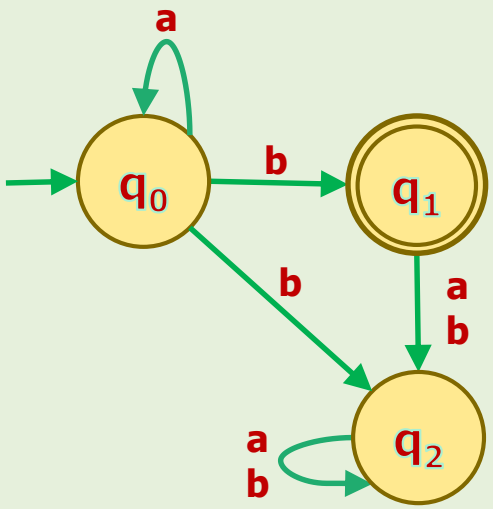
  It has zero transition. ≡ **z**

$$(c \vee z) \leftrightarrow h$$

NFAs' Behavior For the Violation #2

Violation #2

- There are some timeframes that the machine has more than one transition.
  - e.g.: $\delta(q_0, b) = \{q_1, q_2\}$



NFAs' Behavior

- It checks all possibilities by "parallel processing". How?
- It initiates another process. How?
  - It replicates itself.
  - It initializes the new process with the current configuration.
  - The new process independently continues processing the rest of the input string.

Basic Questions of "How NFAs Work"

- So far, we've responded three out of four questions:

#	Question	Answer
1	What is the "starting configuration"?	Same as DFAs
2	What would happen during a timeframe?	Halting (Violation #1) Parallel processing (Violation #2) Same as DFAs for the rest
3	When would the machine halt?	$(c \vee z) \leftrightarrow h$
4	How would a string be Accepted/Rejected?	???

- Before responding the last question, let's take some examples and see NFAs in Action!

NFAs in Action

Input Tapes of NFAs Operation

- Input Tapes of NFAs have one more task to do.

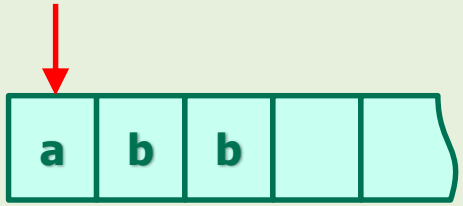
- Here is what an input tape does:

  1. It reads the symbol at which it is pointing and sends it to the control unit.
  2. If the control unit consume it, then the head moves one cell to the right.

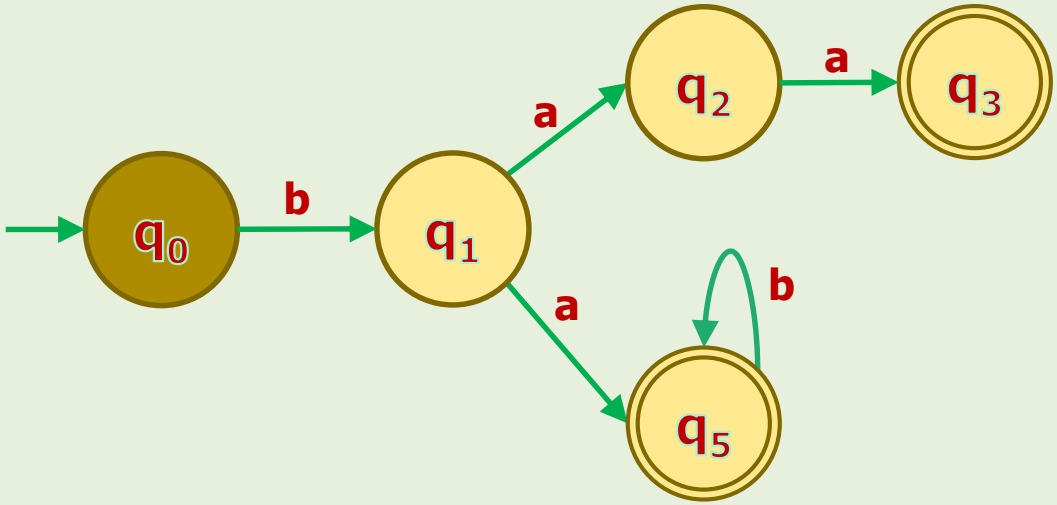
- So, for NFAs, the meaning of "consuming" is NOT equal to "reading" and "scanning" any longer.
- Consuming = reading (or scanning) a symbol + moving the head

NFAs in Action

Example 3: Starting Configuration



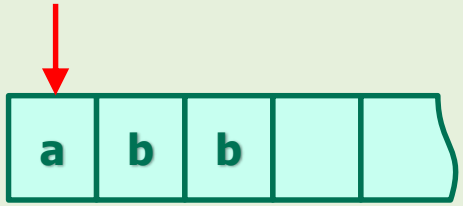
- $\Sigma = \{a , b\}$
- w = abb



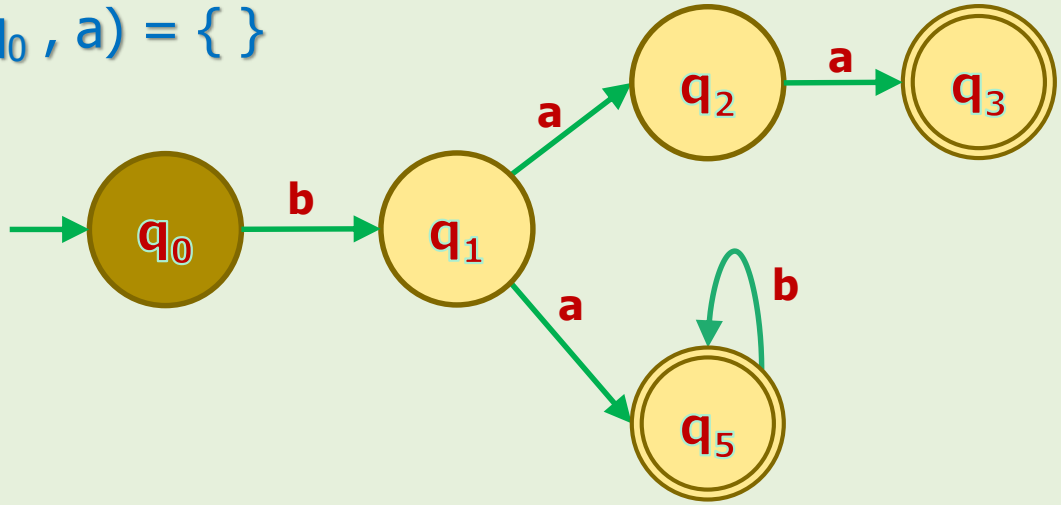
- Process #1 starts normally.

NFAs in Action

Example 3: Process #1 (main)



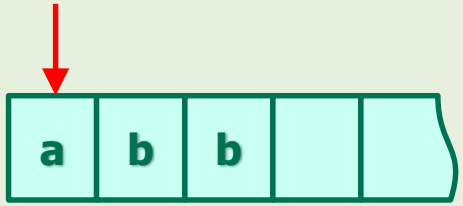
- $\Sigma = \{a , b\}$
- $w = abb$
- $\delta (q_0 , a) = \{ \}$



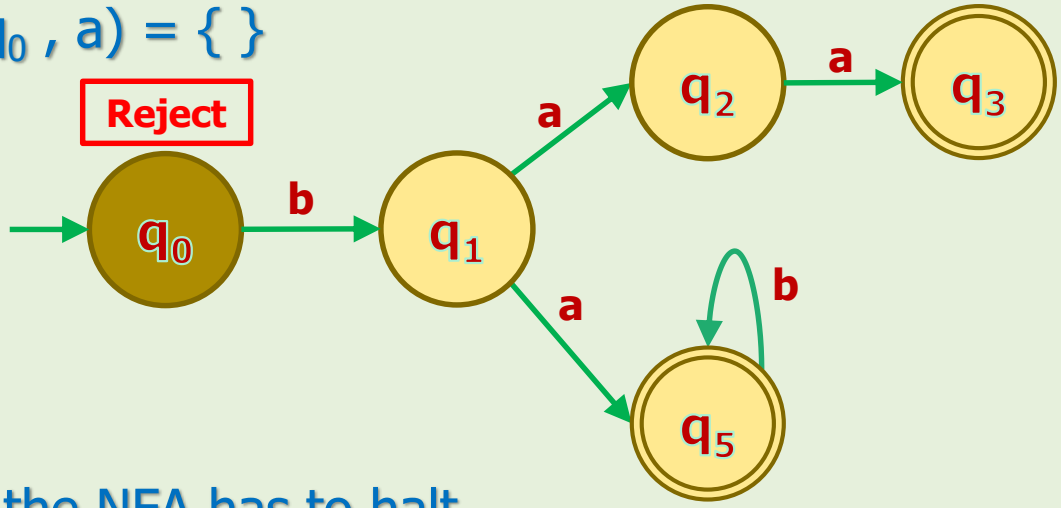
- Input tape reads 'a' and sends it to the control unit.
- The control unit cannot consume it because it has no choice for 'a'.
- The head does not move because control unit did not consume it.

NFAs in Action

Example 3: Process #1 (main)



- $\Sigma = \{a , b\}$
- $w = abb$
- $\delta (q_0 , a) = \{ \}$

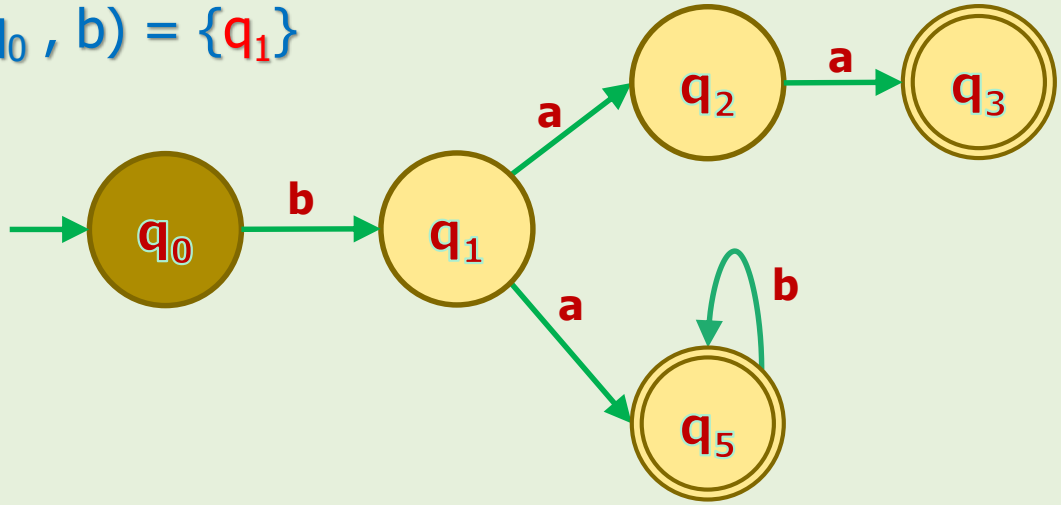


- So, the NFA has to halt.
- The string w is rejected because the machine halts in a non-accepting state.
- Also, all symbols are not consumed. (One reason is enough.)

NFAs in Action

Example 4: Starting Configuration

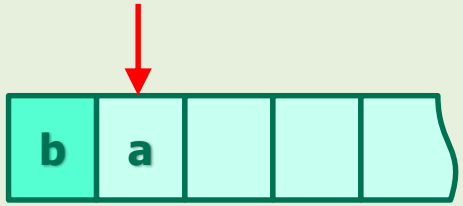
- $\Sigma = \{a , b\}$
- $w = ba$
- $\delta (q_0 , b) = \{q_1\}$



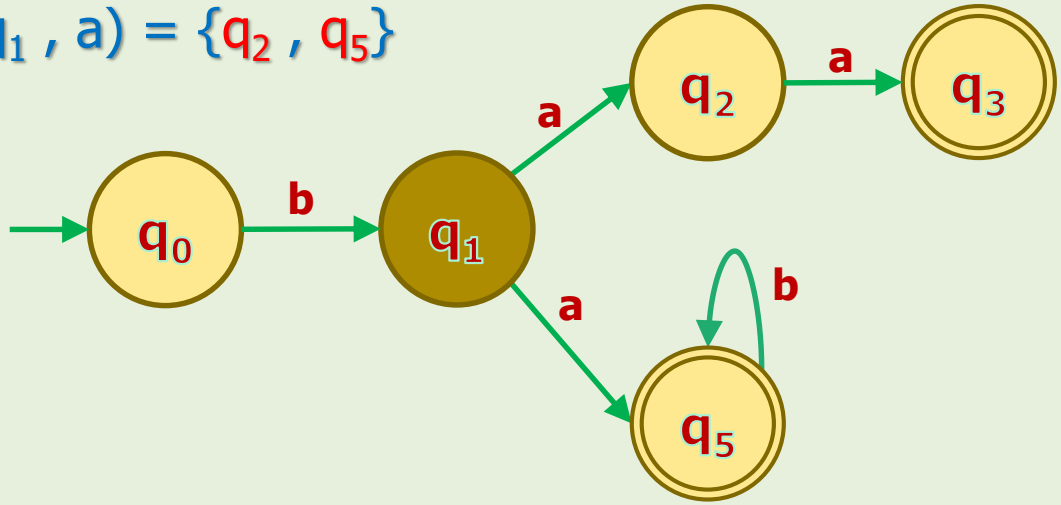
- Process #1 starts normally.

NFAs in Action

Example 4: Process #1 (main)



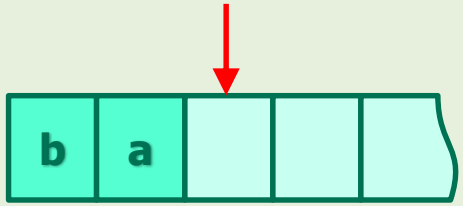
- $\Sigma = \{a , b\}$
- $w = ba$
- $\delta(q_1 , a) = \{q_2 , q_5\}$



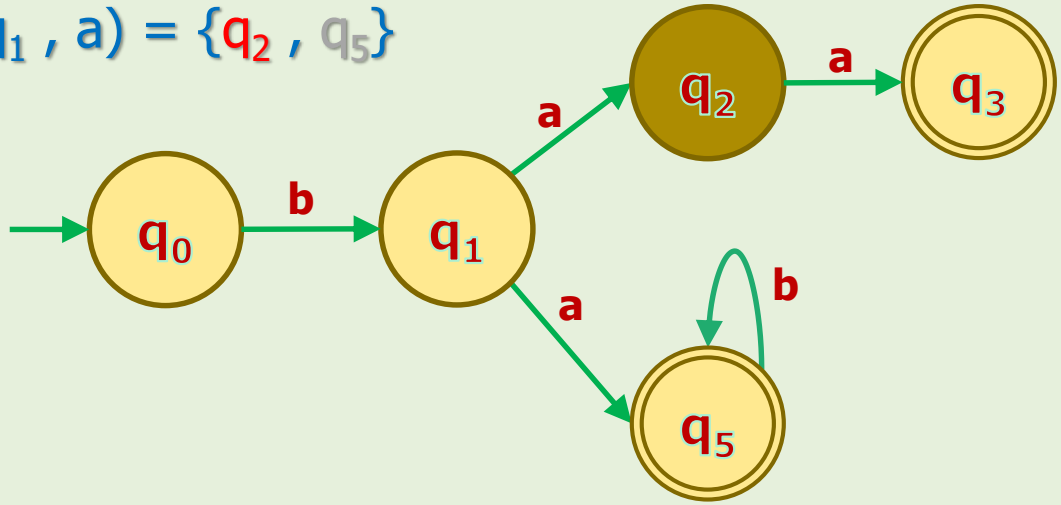
- In the next timeframe, it reads 'a' and encounters two possibilities.
- So, parallel procession starts!

NFAs in Action

Example 4: Process #1 (main)



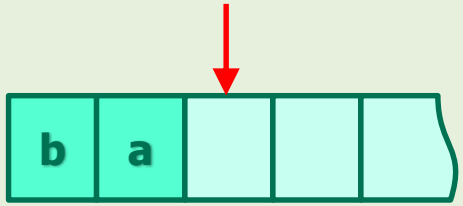
- $\Sigma = \{a , b\}$
- $w = ba$
- $\delta(q_1 , a) = \{q_2 , q_5\}$



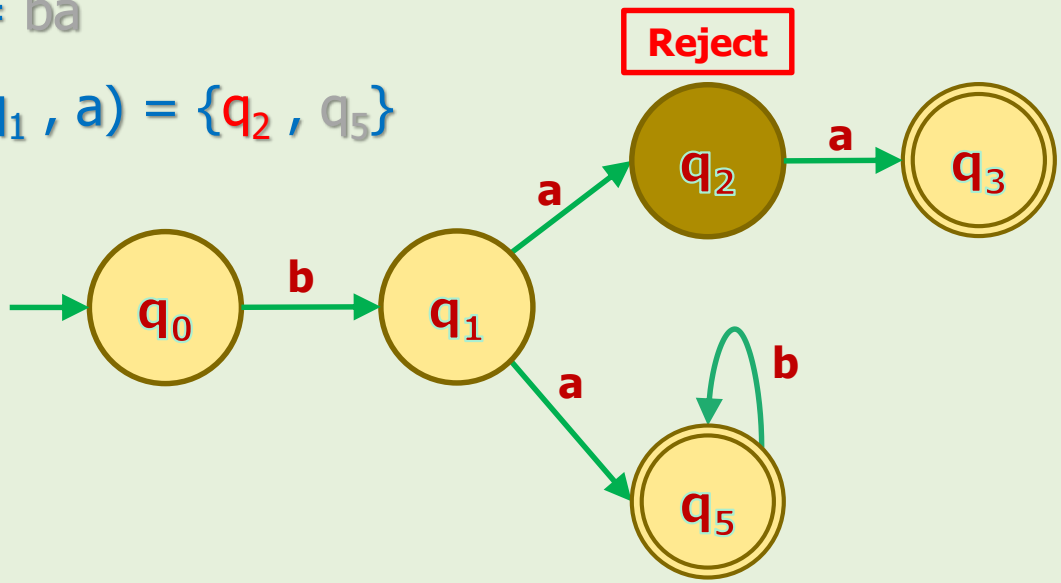
- The parent process (Process #1) will continue with q_2.
- It initiates another process (Process #2) starting from q_5.
- What if there were 3 or more outgoing transitions for 'a'?

NFAs in Action

Example 4: Process #1 (main)



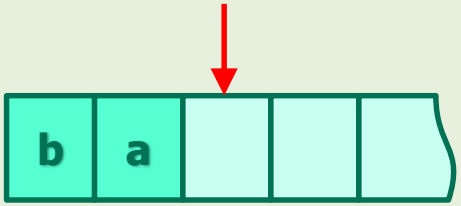
- $\Sigma = \{a , b\}$
- $w = ba$
- $\delta(q_1 , a) = \{q_2 , q_5\}$



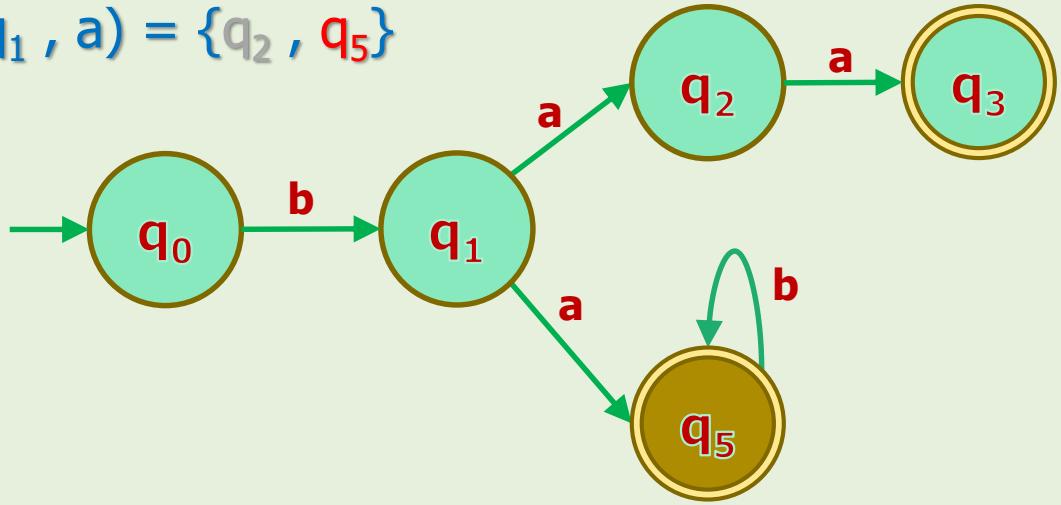
- Process #1 is out of symbol. It consumed all.
- It halts in the non-accepting state q_2.
- So, process #1 rejects w.

NFAs in Action

Example 4: Process #2



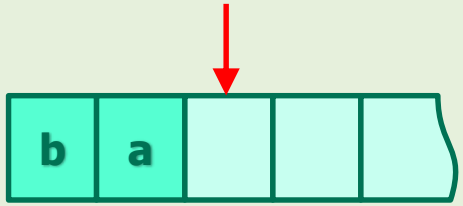
- $\Sigma = \{a , b\}$
- $w = ba$
- $\delta(q_1 , a) = \{q_2 , q_5\}$



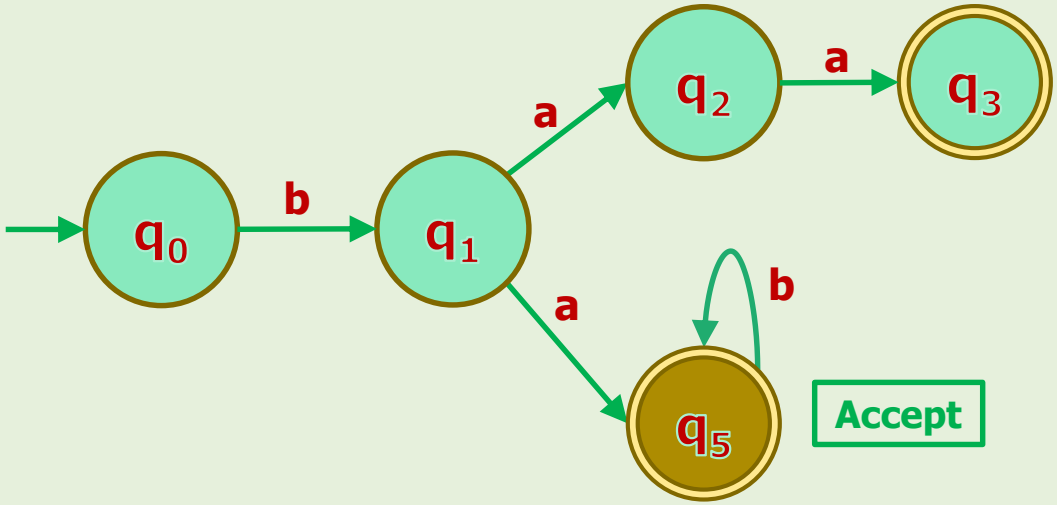
- Process #2 starts from where it was initiated.

NFAs in Action

Example 4: Process #2



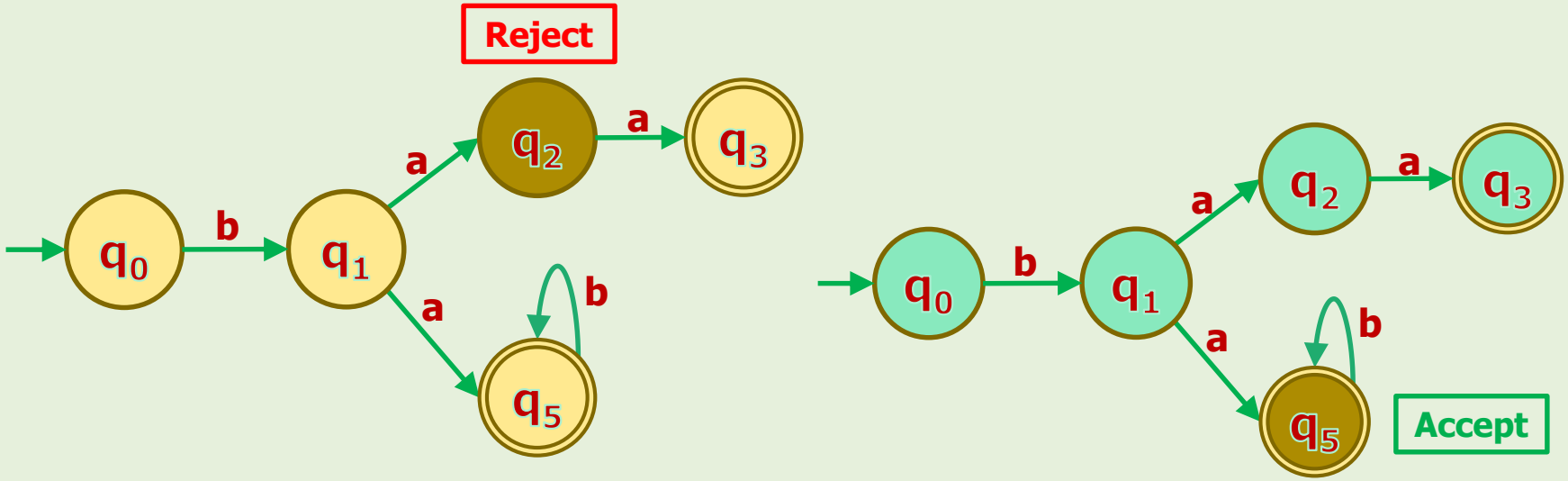
- $\Sigma = \{a , b\}$
- w = ba



- Process #2 is out of symbol too.
- It halts in the accepting state q_5.
- So, process #2 accepts w.

NFAs in Action

Example 4: Overall Result



Process #1 REJECTED w = ba

Process #2 ACCEPTED w = ba

- Overall, the string was ACCEPTED because at least one process (#2) accepted it.

How NFAs Accept/Reject Strings

Accepting Strings

- A string is accepted by an NFA iff at least one process accepts it.

- Note that for NFAs, **(h ∧ c ∧ f) ↔ a** is valid for accepting strings by one process.
  - Because **h** and **c** might have different values.

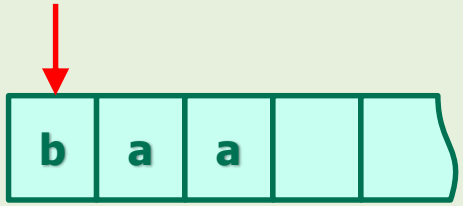
Rejecting Strings

- A string is rejected by an NFA iff all processes reject it.

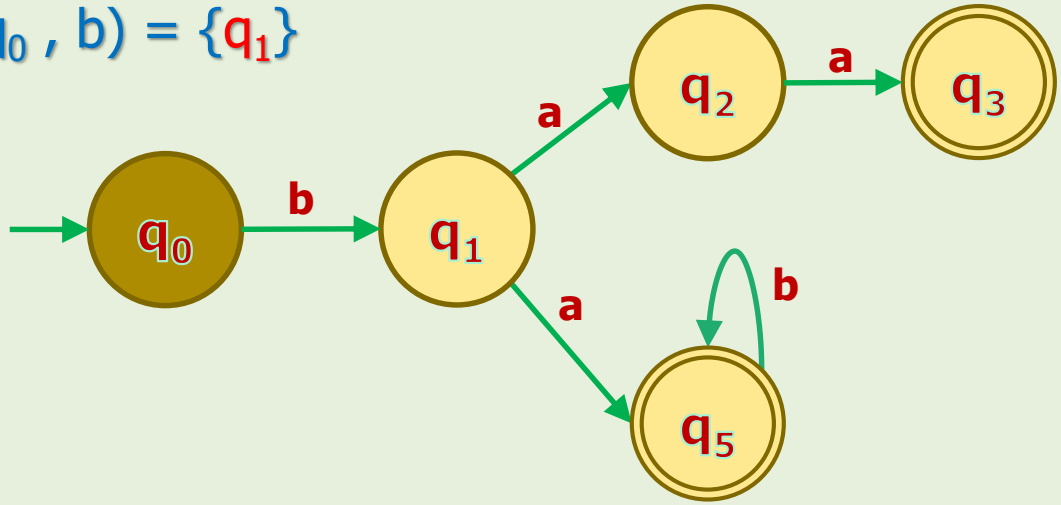
- Let's take more examples.

NFAs in Action

Example 5: Starting Configuration



- $\Sigma = \{a , b\}$
- $w = baa$
- $\delta (q_0 , b) = \{q_1\}$

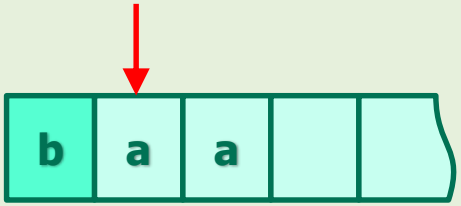


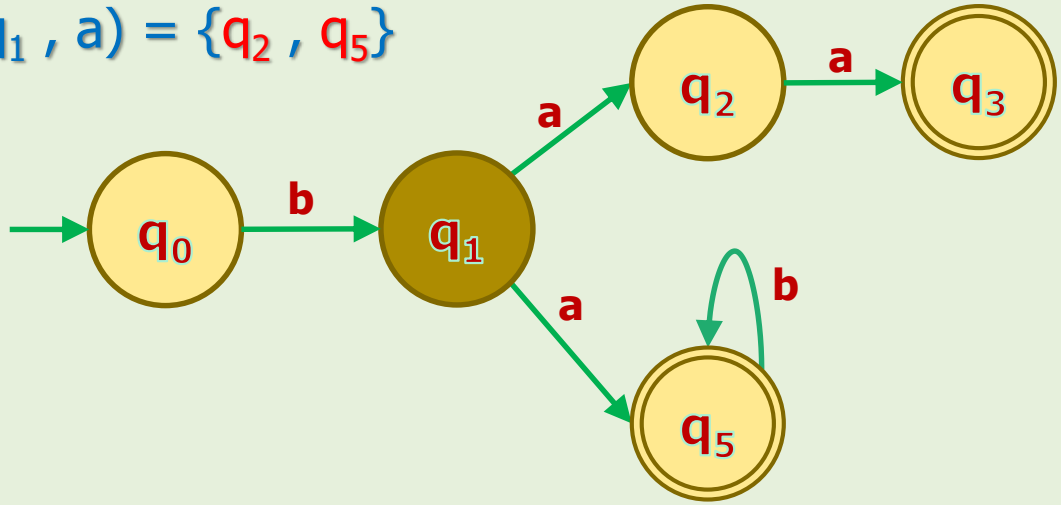
- Process #1 starts normally.

NFAs in Action

Example 5: Process #1 (main)

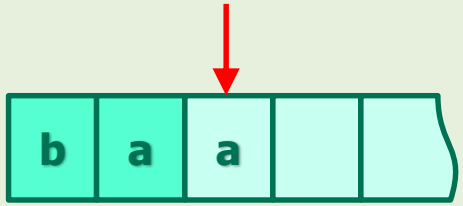
- $\Sigma = \{a , b\}$
- $w = baa$
- $\delta(q_1 , a) = \{q_2 , q_5\}$



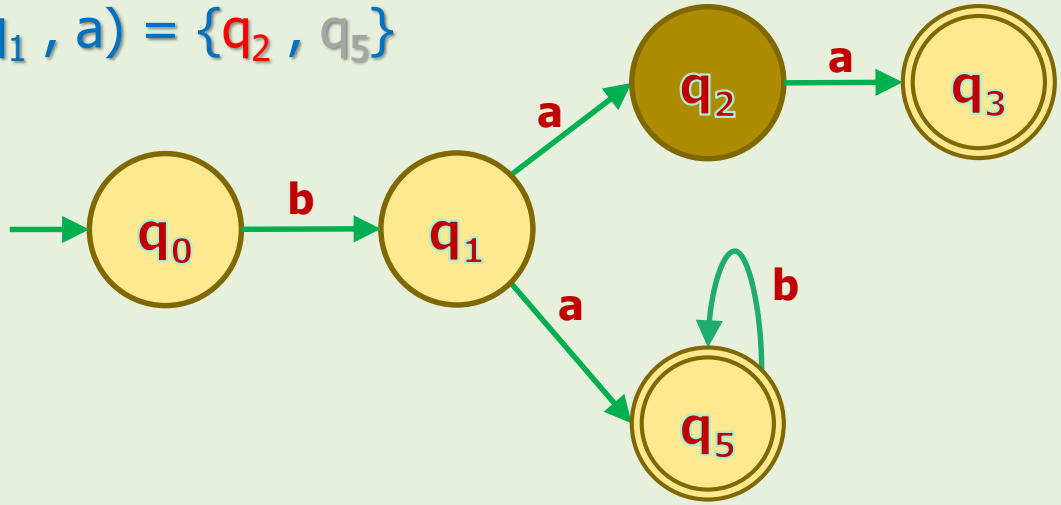


NFAs in Action

Example 5: Process #1 (main)



- $\Sigma = \{a , b\}$
- $w = baa$
- $\delta(q_1 , a) = \{q_2 , q_5\}$

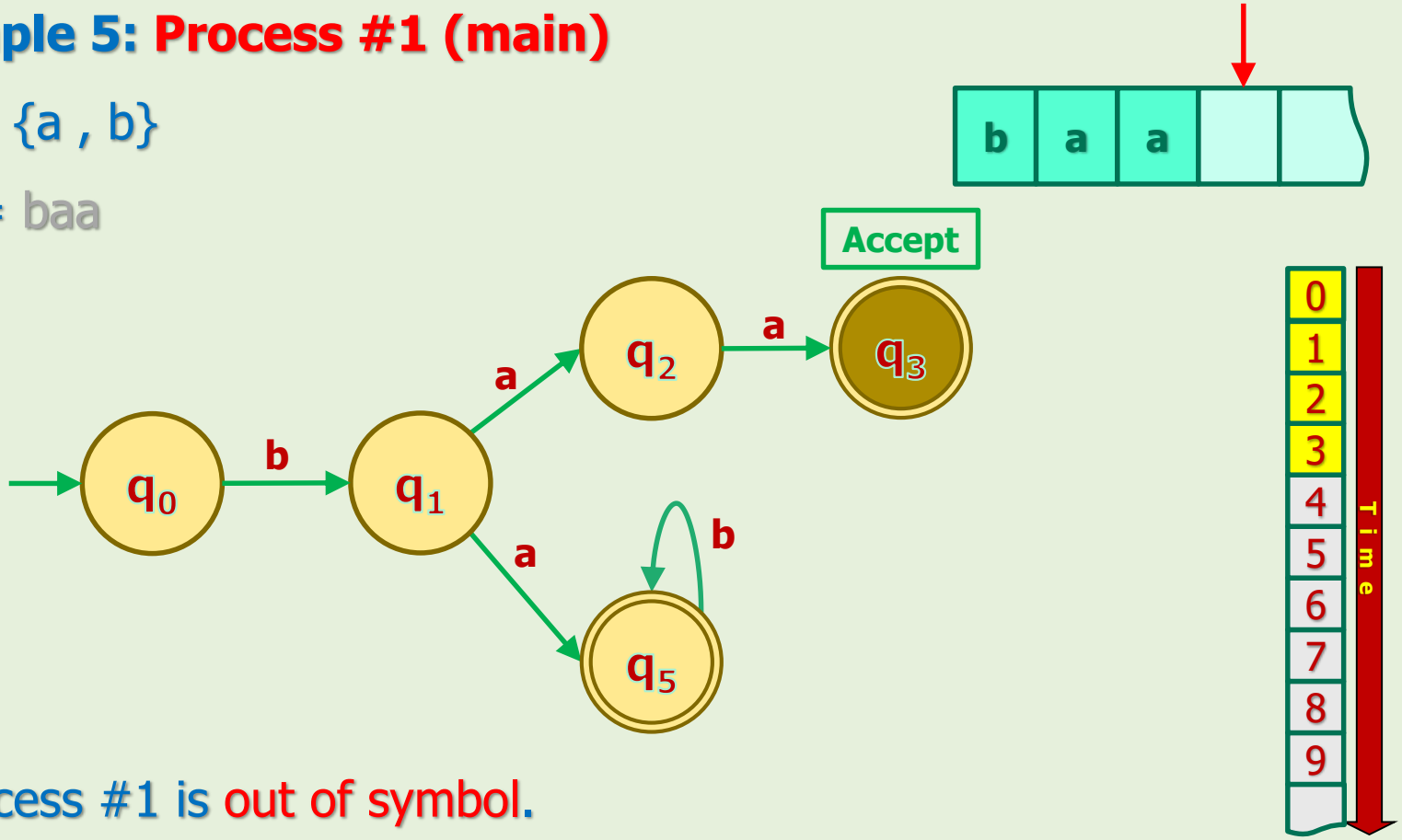


- The parent process (Process #1) will continue with q_2.
- It initiates another process (Process #2) starting from q_5.

NFAs in Action

Example 5: Process #1 (main)

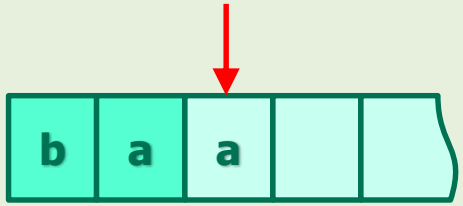
- $\Sigma = \{a , b\}$
- w = baa



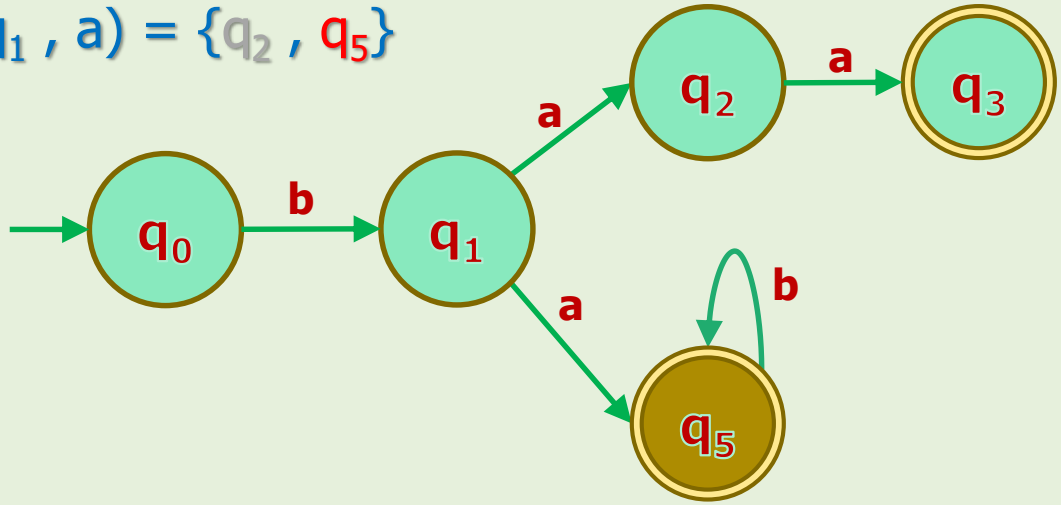
- Process #1 is out of symbol.
- It halts in the accepting state q_3.
- So, process #1 accepts w.

NFAs in Action

Example 5: Process #2



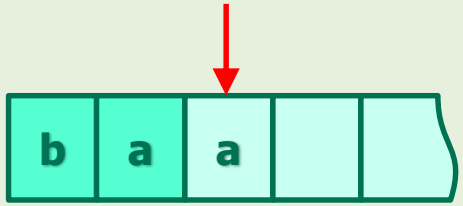
- $\Sigma = \{a , b\}$
- $w = baa$
- $\delta(q_1 , a) = \{q_2 , q_5\}$



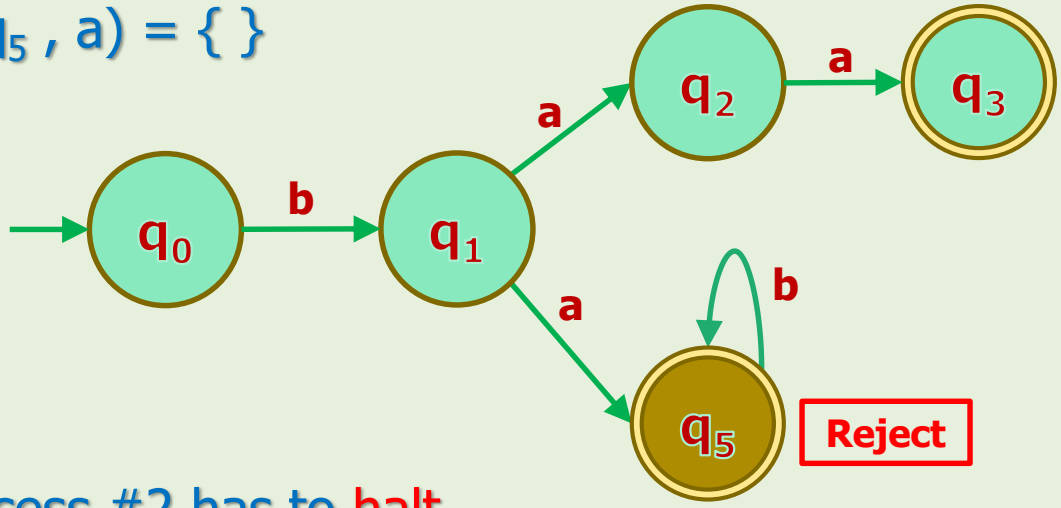
- Process #2 starts from where it was initiated.

NFAs in Action

Example 5: Process #2



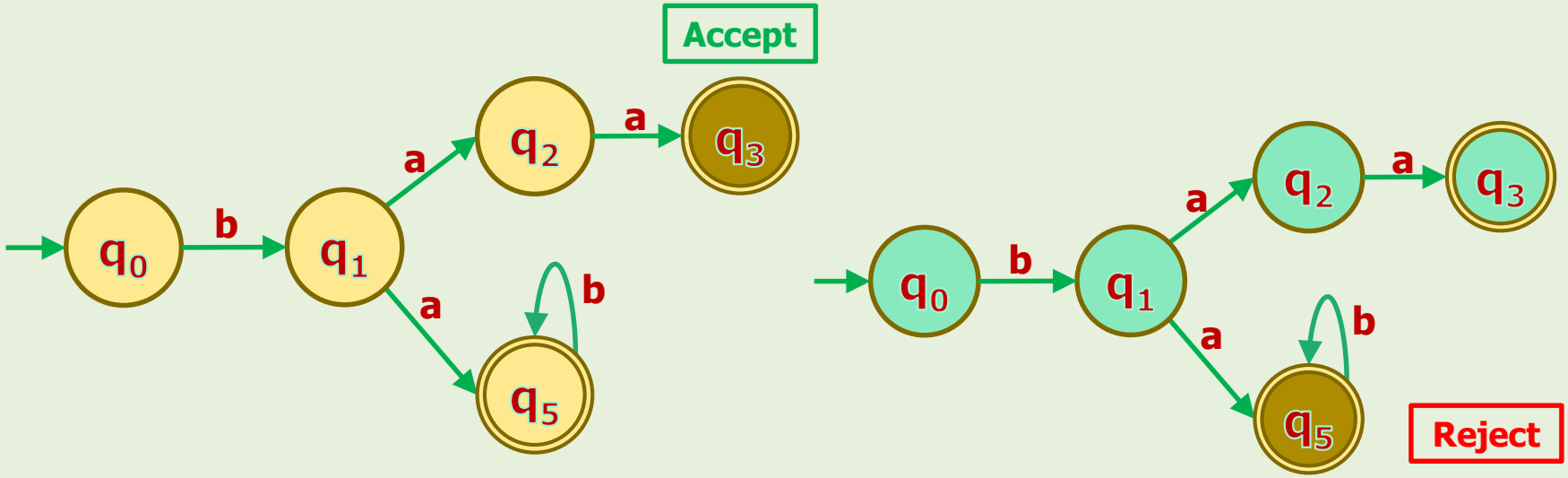
- $\Sigma = \{a , b\}$
- w = baa
- $\delta (q_5 , a) = \{ \}$



- Process #2 has to halt
  because for input 'a', it has no transition.
- All input symbols are NOT consumed.
- So, process #2 rejects w.

NFAs in Action

Example 5: Overall Result



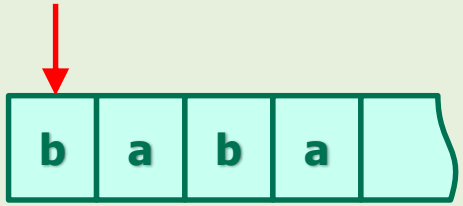
Process #1 ACCEPTED w = baa

Process #2 REJECTED w = baa

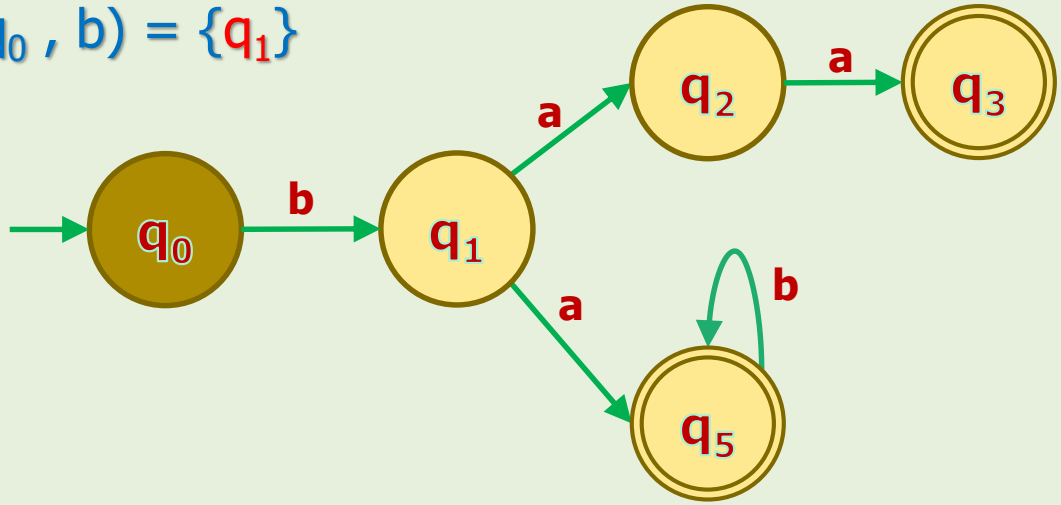
- Overall, the string was ACCEPTED because at least one process (#1) accepted it.

NFAs in Action

Example 6: Starting configuration



- $\Sigma = \{a , b\}$
- w = baba
- $\delta (q_0 , b) = \{q_1\}$

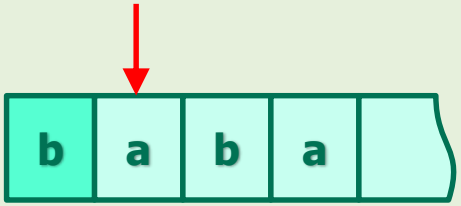


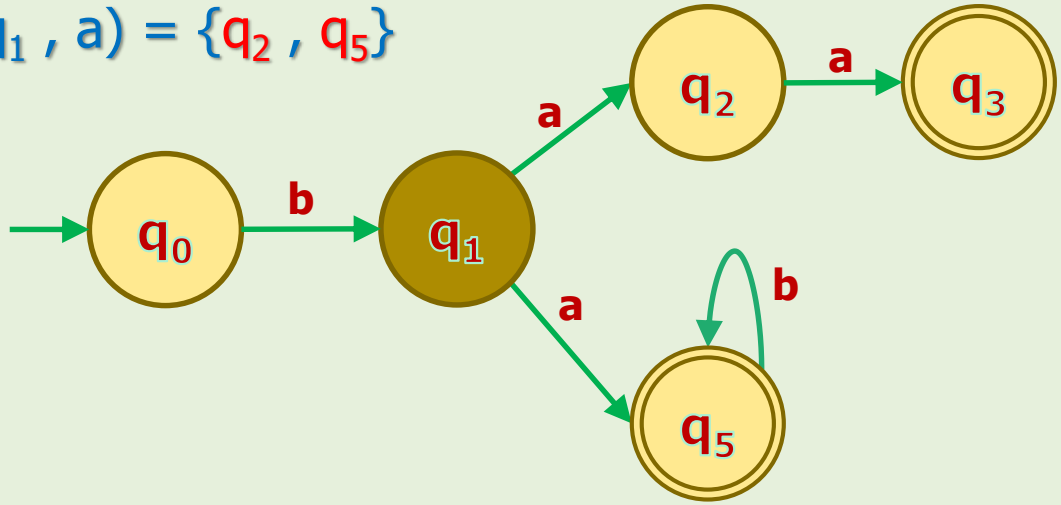
- Process #1 starts normally.

NFAs in Action

Example 6: Process #1 (main)

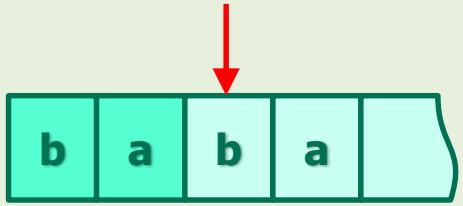
- $\Sigma = \{a , b\}$
- w = baba
- $\delta (q_1 , a) = \{q_2 , q_5\}$



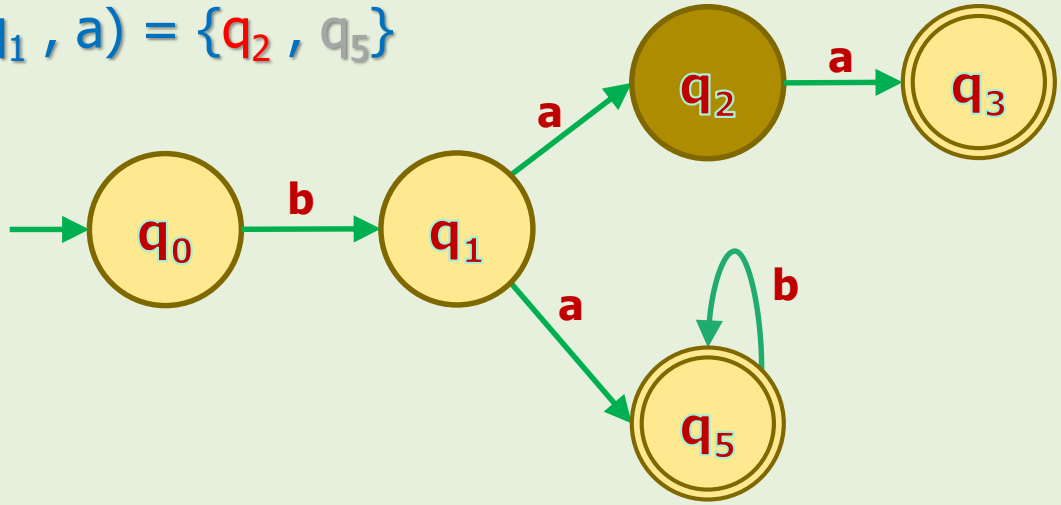


NFAs in Action

Example 6: Process #1 (main)



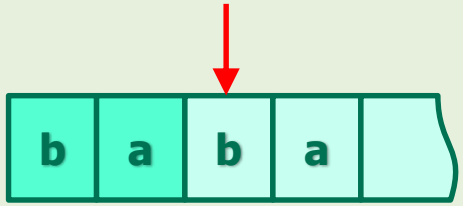
- $\Sigma = \{a , b\}$
- w = baba
- $\delta(q_1 , a) = \{q_2 , q_5\}$



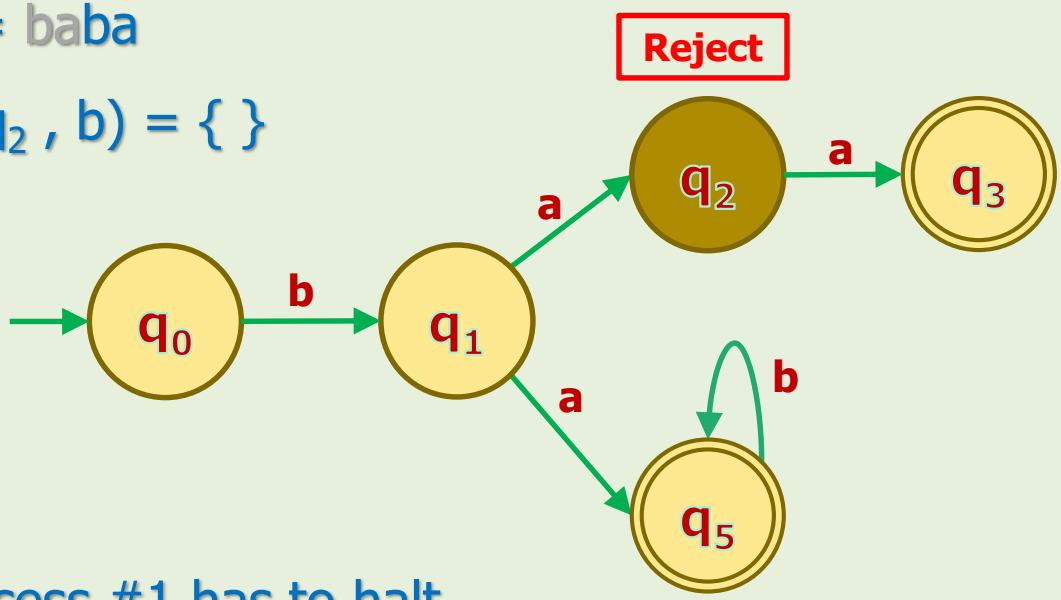
- The parent process (Process #1) will continue with q_2.
- It initiates another process (Process #2) starting from q_5.

NFAs in Action

Example 6: Process #1 (main)



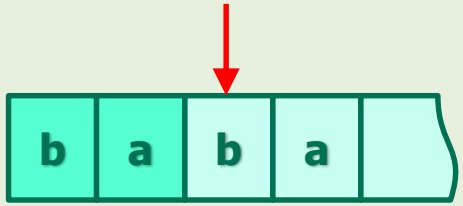
- $\Sigma = \{a , b\}$
- w = baba
- $\delta(q_2 , b) = \{ \}$



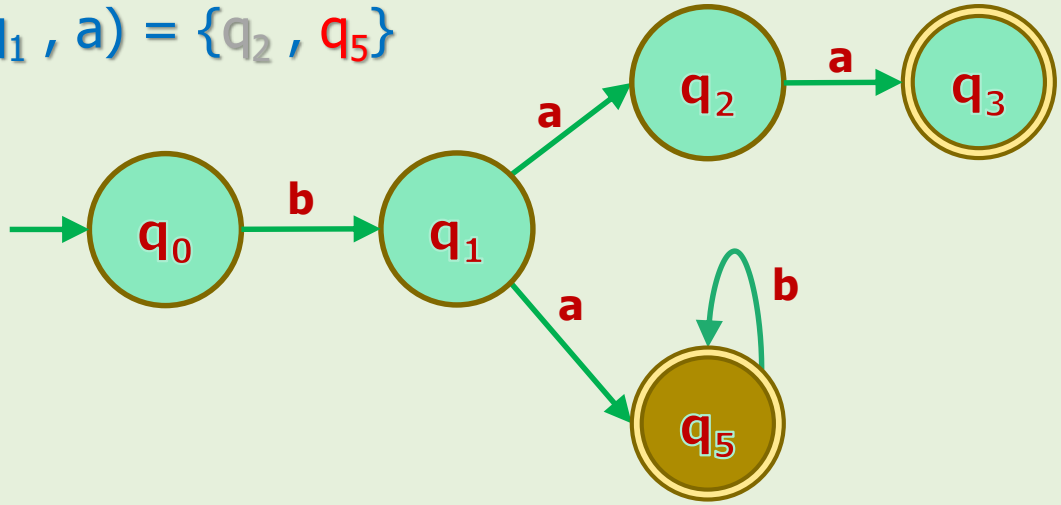
- Process #1 has to halt
  because for input 'b', it has NO transition.
- Also, all input symbols are NOT consumed. (One reason is enough.)
- So, process #1 rejects w.

NFAs in Action

Example 6: Process #2



- $\Sigma = \{a , b\}$
- $w = baba$
- $\delta(q_1 , a) = \{q_2 , q_5\}$

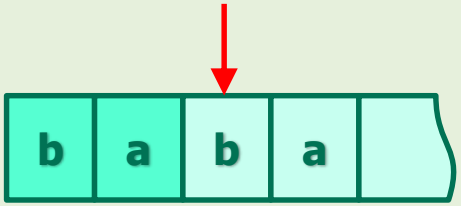


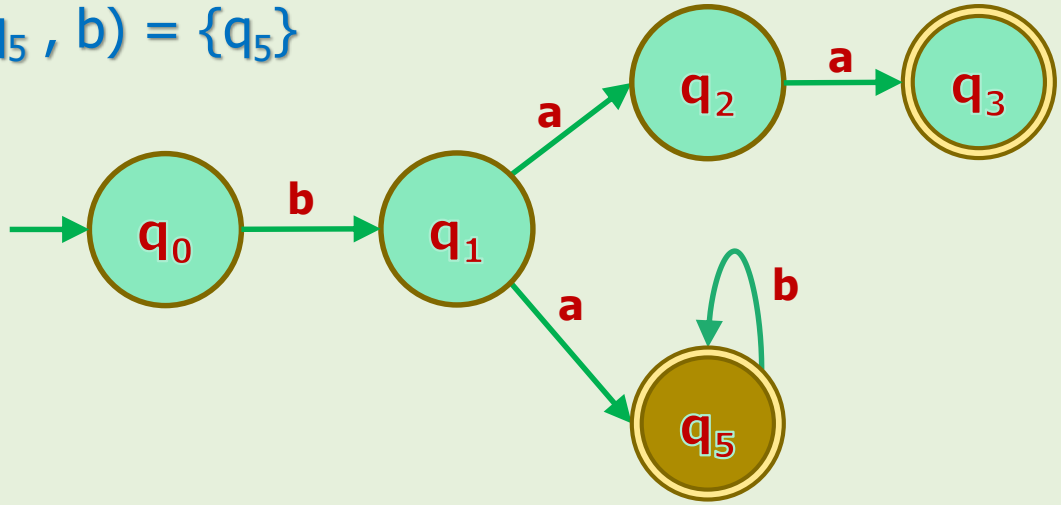
- Process #2 starts from where it was initiated.

NFAs in Action

Example 5: Process #2

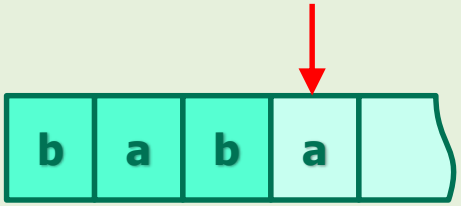
- $\Sigma = \{a , b\}$
- w = baba
- $\delta (q_5 , b) = \{q_5\}$



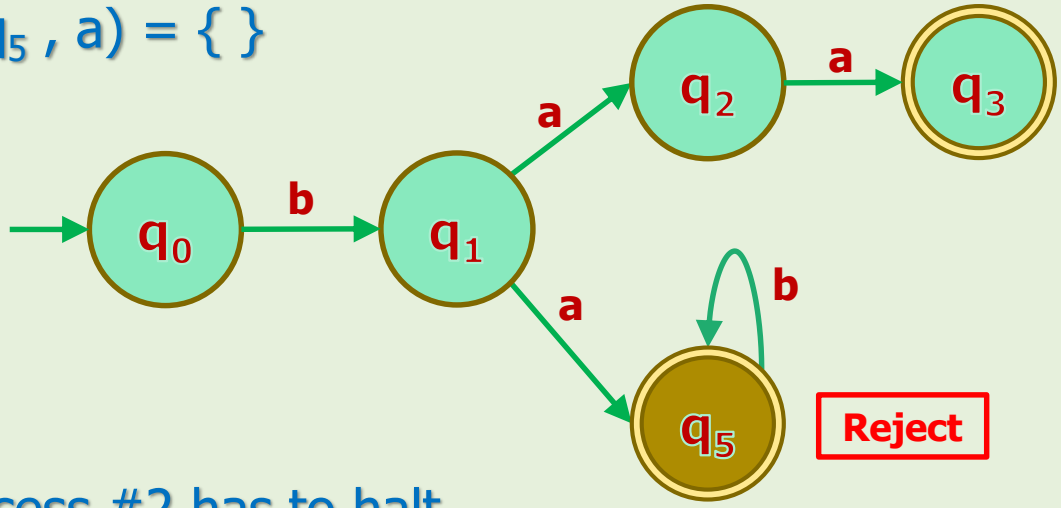


NFAs in Action

Example 6: Process #2



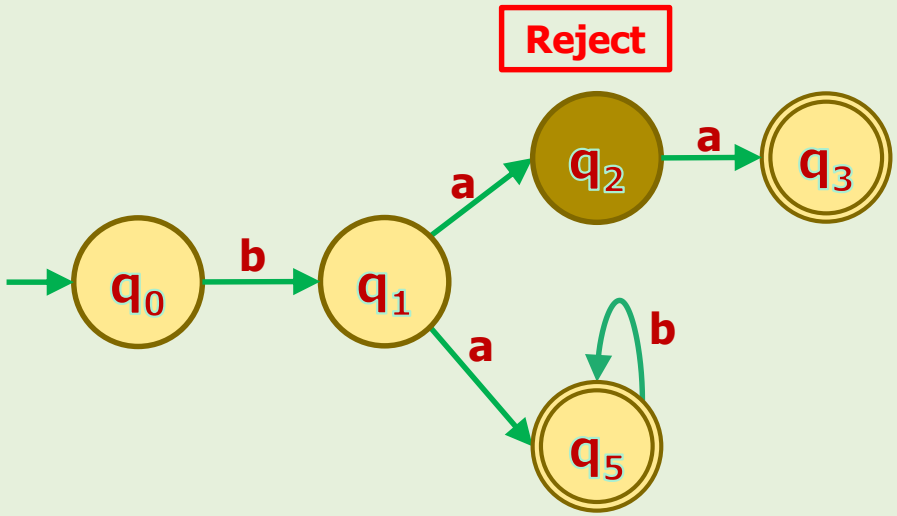
- $\Sigma = \{a , b\}$
- w = baba
- $\delta (q_5 , a) = \{ \}$

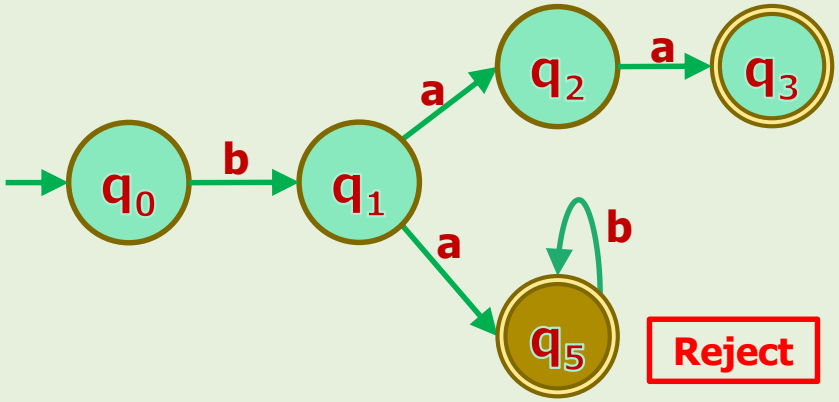


- Process #2 has to halt
  because for input 'a', it has NO transition.
- Also, all input symbols are NOT consumed. (One reason is enough.)
- So, process #2 rejects w.

NFAs in Action

Example 6: Overall Result





Process #1 REJECTED w = baba

Process #2 REJECTED w = baba

- Overall, the string was REJECTED because both processes rejected it.

References

1. Linz, Peter, "An Introduction to Formal Languages and Automata, 5th ed.," Jones & Bartlett Learning, LLC, Canada, 2012
2. Kenneth H. Rosen, "Discrete Mathematics and Its Applications, 7th ed.," McGraw Hill, New York, United States, 2012
3. Michael Sipser, "Introduction to the Theory of Computation, 3rd ed.," CENGAGE Learning, United States, 2013
   ISBN-13: 978-1133187790