San José State University
Department of Computer Science

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

FIFOAs

CS 154
Formal Languages and Computability
Fall 2019

Template for Introducing a New Class of Automata

Repeated

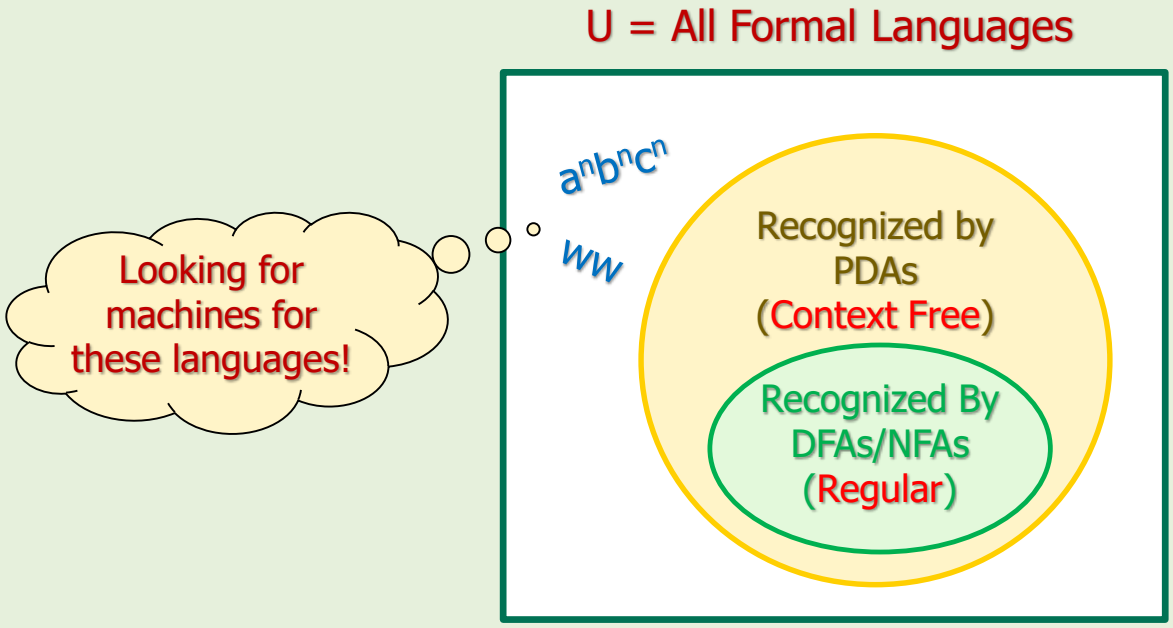
- To construct a new class of automata, we'll follow the following steps:

1. Why do we need this new class? (Justification)
2. Name of this new class
3. Building blocks of this new class
4. How do they work?
   - 4.1. What is the starting configuration?
   - 4.2. What would happen during a timeframe?
   - 4.3. When would the machine halt?
   - 4.4. How would a string be Accepted/Rejected?
5. The automata in action
6. Formal definition
7. Their power: this class versus previous class
8. What would be the next possible class?

- We'll just mention some of these steps because the rest are similar to PDAs.

1. Why do We Need a New Class?

- So far, we've learned that PDAs are more powerful than NFAs.
- But there are still some languages like ww or $a^nb^nc^n$ that cannot be recognized by PDAs.
- So, we are looking for a more powerful class of automata that can recognize all, or at least some of those languages.



2. Name of this New Class

- The memory of this new class is structured as "queue".
- Queues work in a "first in – first out" manner.
- So, generally we call this new class:

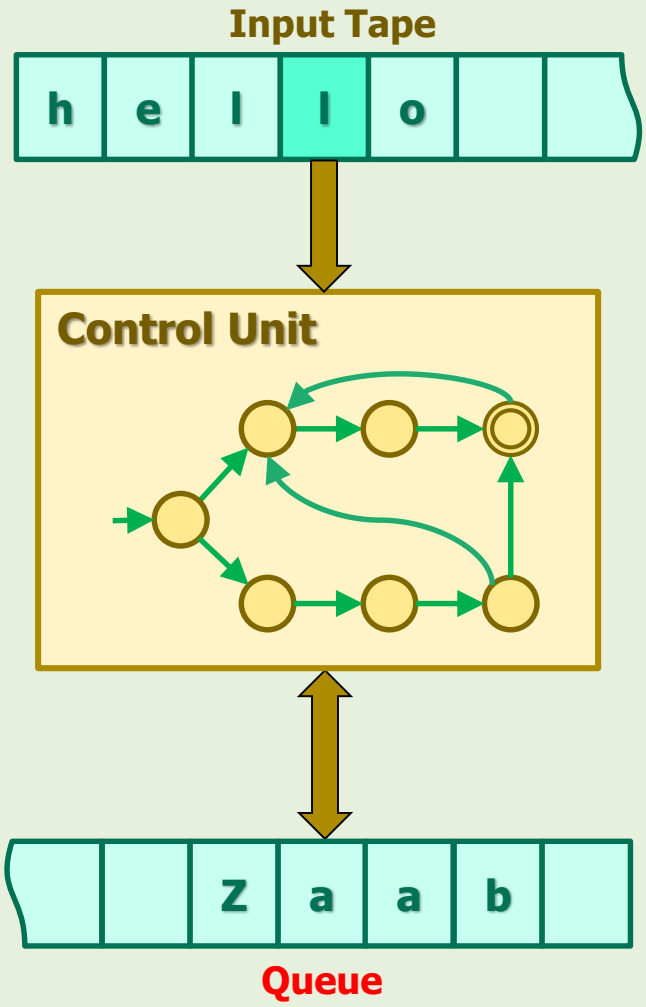
First in – First out Automata (FIFOA)

- This class is deterministic.
- And its number of states is finite.

3. FIFOAs Building Blocks

3. FIFOAs Building Blocks

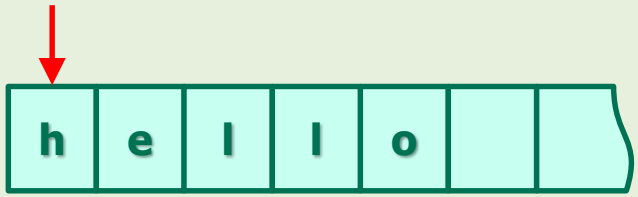
- FIFOAs have 3 main blocks:
  1. Input Tape
  2. Queue
  3. Control Unit



- Let's see each block in detail.

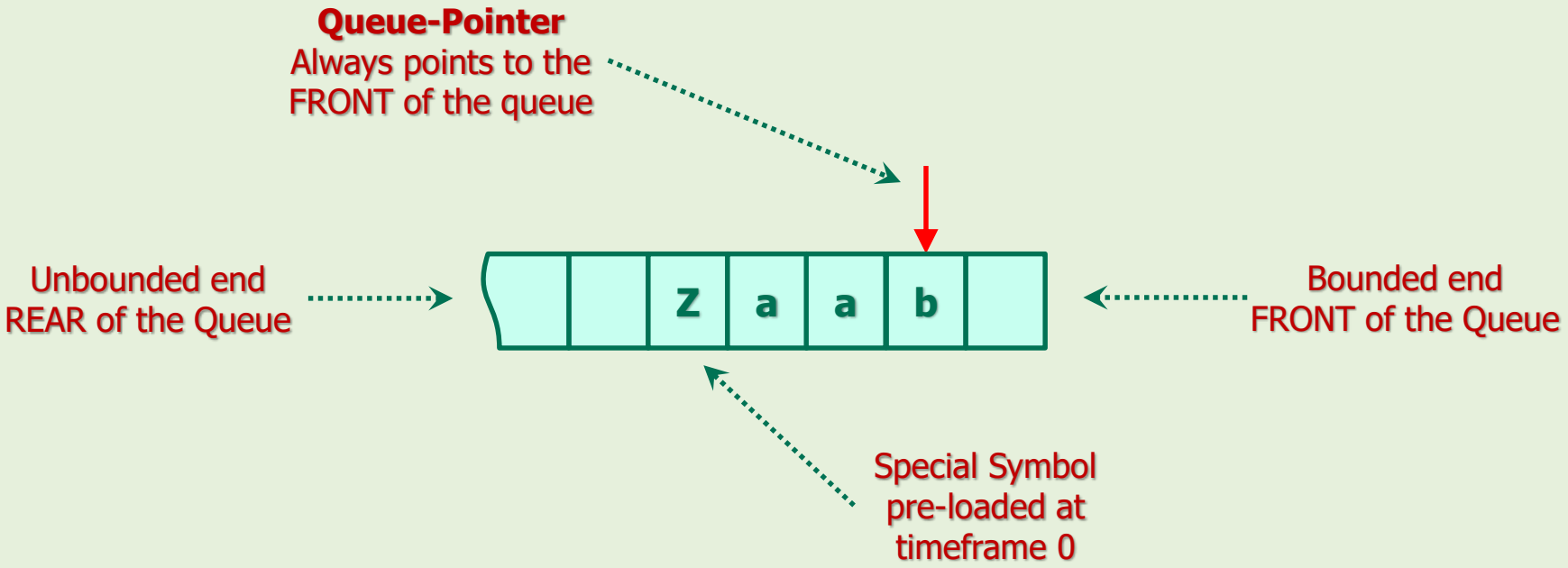
3.1. Input Tape

- The input tape of FIFOAs is exactly the same as DFAs'.



- For the detail, please refer to DFAs' input tape.

3.2. Queue: Structure

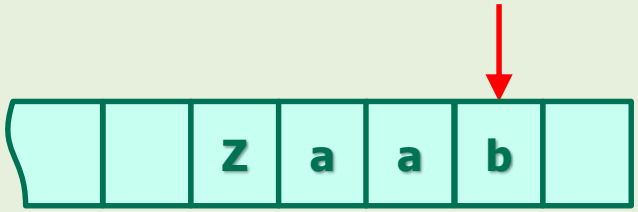


- The special symbol 'Z' is written at the REAR of the queue by the machine at timeframe 0.
  - Detail will be covered shortly.
- When queue pointer is pointing to 'Z', it means that the queue is empty.

3.2. Queue: How It Works

Operations on Queue

- Queue works based on first in – first out (FIFO) manner.

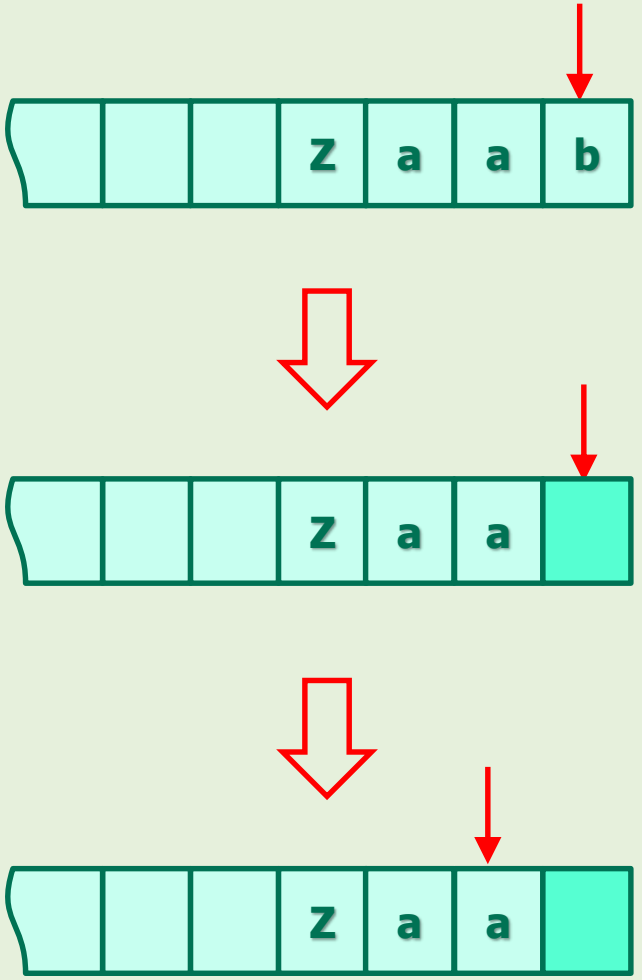


- The basic operations on queues are "dequeue" and "enqueue".
  - These operations are similar to what you've learned in data structure course for high-level queues.

- Nevertheless, let's have a quick review of these basic operations.

3.2. Queue: Operations on Queues

Dequeue

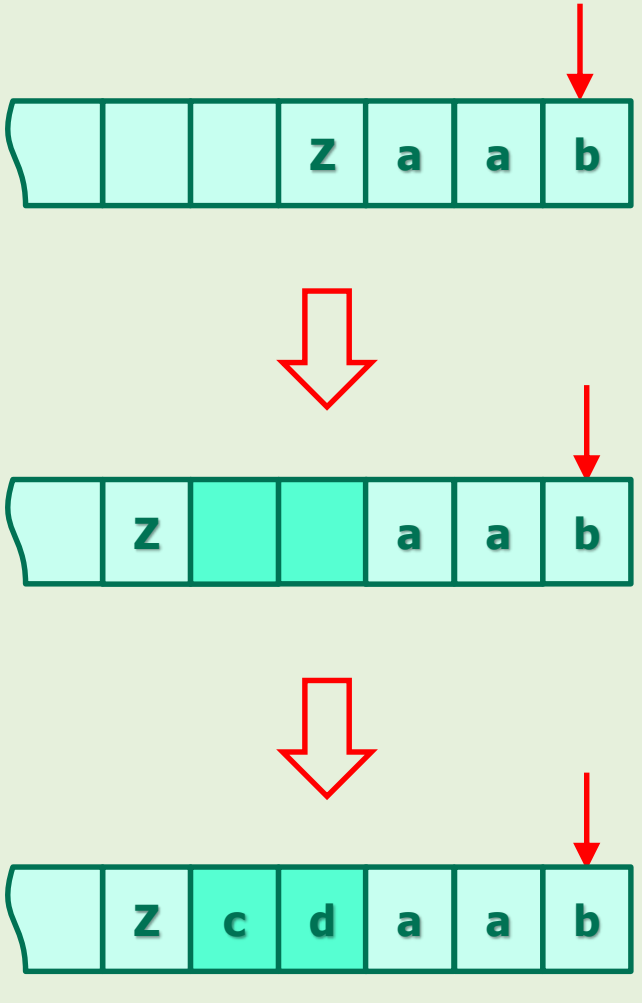


1. Remove the symbol at which the queue-pointer is pointing

2. Move the queue-pointer one cell left
   - All of these phases happen during one timeframe.

3.2. Queue: Operations on Queues

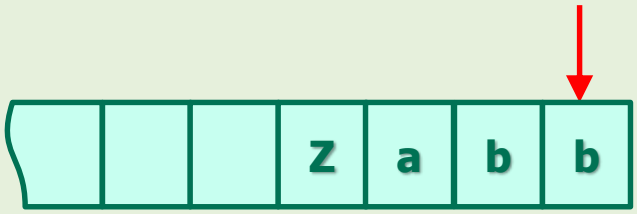
Enqueue



1. Move the 'Z' one (or more) cell(s) left, depends on |w|

2. Put the string w in the queue the right symbol goes first (e.g. if w = cd, put d first, then 'c')
   - All of these phases happen during one timeframe.

3.2. Queue: Note



1. The "queue alphabet" and the "input tape alphabet" can be totally different.

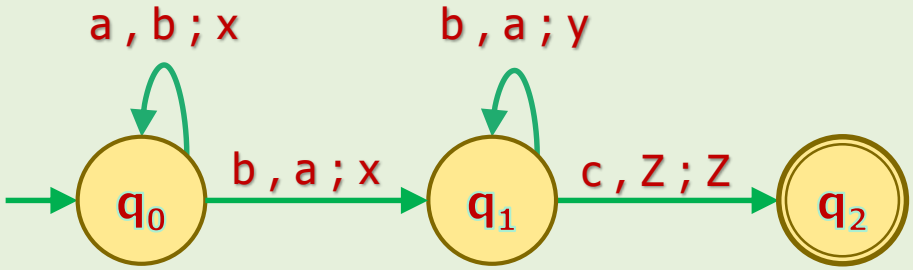
2. If your algorithm requires, you can enqueue 'Z' as many times as you like, and you can dequeue even the last one!

   Therefore, it's designer's responsibility to take care of the contents of the queue.

3.3. Control Unit: Structure

- The control unit of FIFOAs look pretty much like PDAs'.
  - They are represented by "transition graphs".

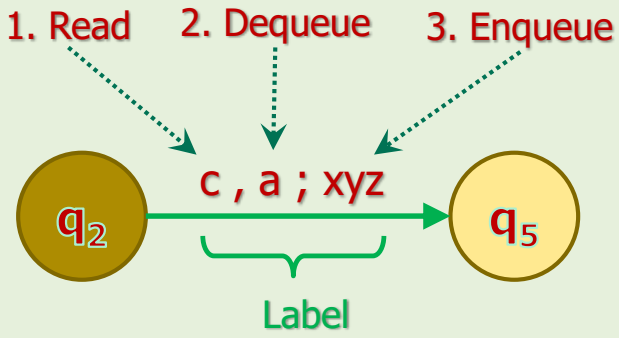
- This is an example of a FIFOA's transition graph.



- The only difference is how the edges are labeled.

- Let's analyze a transition in detail.

3.3. Control Unit: Labels



- The label has 3 parts delimited by comma and semicolon:
  1. The input symbol (e.g. 'c') that should be read from the tape
  2. The symbol at the FRONT of the queue (e.g. 'a') that should be dequeued
  3. The string (e.g. 'xyz') that should be enqueued into the REAR of the queue

4. How FIFOAs Work

4. How FIFOAs Work

Repeated

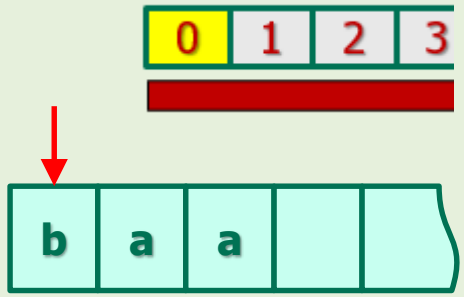
- To understand how FIFOAs work, we should clearly respond to the following questions:

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

4.1. FIFOAs Starting Configuration

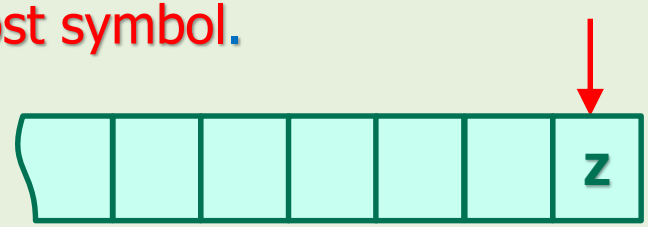
Clock

- The clock is set at timeframe 0.



Input Tape

- The input string has already been written on the tape.
- The read-head is pointing to the left-most symbol.

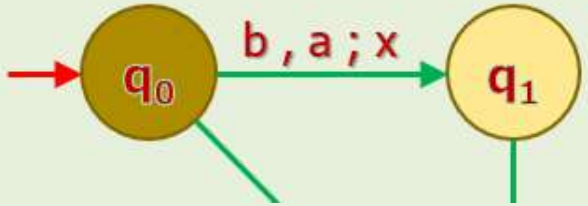


Queue

- The queue is initialized by the symbol 'Z'.
- The queue-pointer is pointing to the 'Z'.

Control Unit

- The control unit is set to initial state.



Relaxing the Conditions and Operations by λ

- If we put λ in any part of the labels, it means "no condition" or "no action" in that part.

- So, by using 'λ', we can relax the conditions and the operations.

4.2. What Happens During a Timeframe

- The following tasks happen during a timeframe.

1. Zero or one symbol at which the cursor is pointing, is consumed.
2. Zero or one symbol is dequeued from the queue.
3. A string (could be empty) is enqueued into the queue.
4. The control unit makes its move based on the "logic of the transition".

4.3. When FIFOAs Halt

- FIFOAs halt when the next transition conditions are NOT satisfied.

- Transition Conditions = input symbol + front of the queue

Halt Logical Representation

FIFOAs halt. ≡ **h**

IFF

They have zero transition. ≡ **z**

} **z ↔ h**

4.4. How FIFOAs Accept Strings

Logical Representation of Accepting Strings

FIFOAs accept a string w. ≡ **a**

IFF

They halt. ≡ **h**

AND

All symbols of w are consumed. ≡ **c**

AND

They are in an accepting (final) state. ≡ **f**

$$(h \wedge c \wedge f) \leftrightarrow a$$

4.4. How FIFOAs Reject Strings

Logical Representation of Rejecting Strings

$$\sim(h \wedge c \wedge f) \leftrightarrow \sim a$$

$$(\sim h \vee \sim c \vee \sim f) \leftrightarrow \sim a$$

Translation

FIFOAs reject a string w. ≡ ~**a**

IFF

They do NOT halt. ≡ ~**h**

OR

At least one symbol of w is NOT consumed. ≡ ~**c**

OR

They are NOT in an accepting (final) state. ≡ ~**f**

4.4. Accepting/Rejecting Strings: Notes

- The final contents of the queue is NOT important in accepting or rejecting a string.
  - Because queue is in fact a workspace for drafting (like scratch paper).
  - It is a place to store the middle results of the computation.

5. FIFOAs in Action

Design Examples

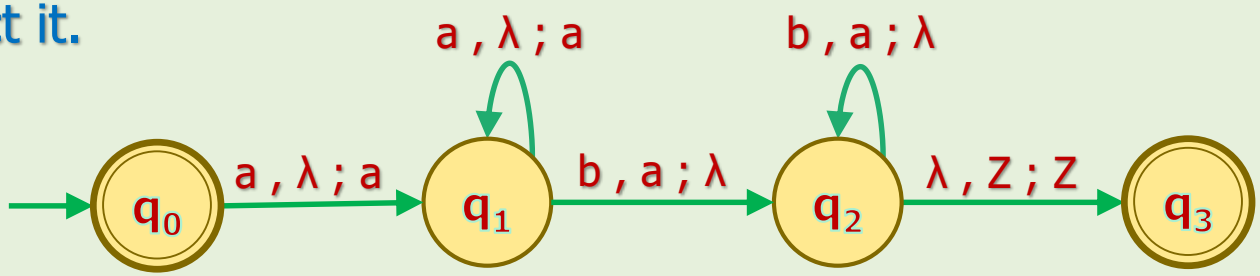
5. FIFOAs in Action

Example

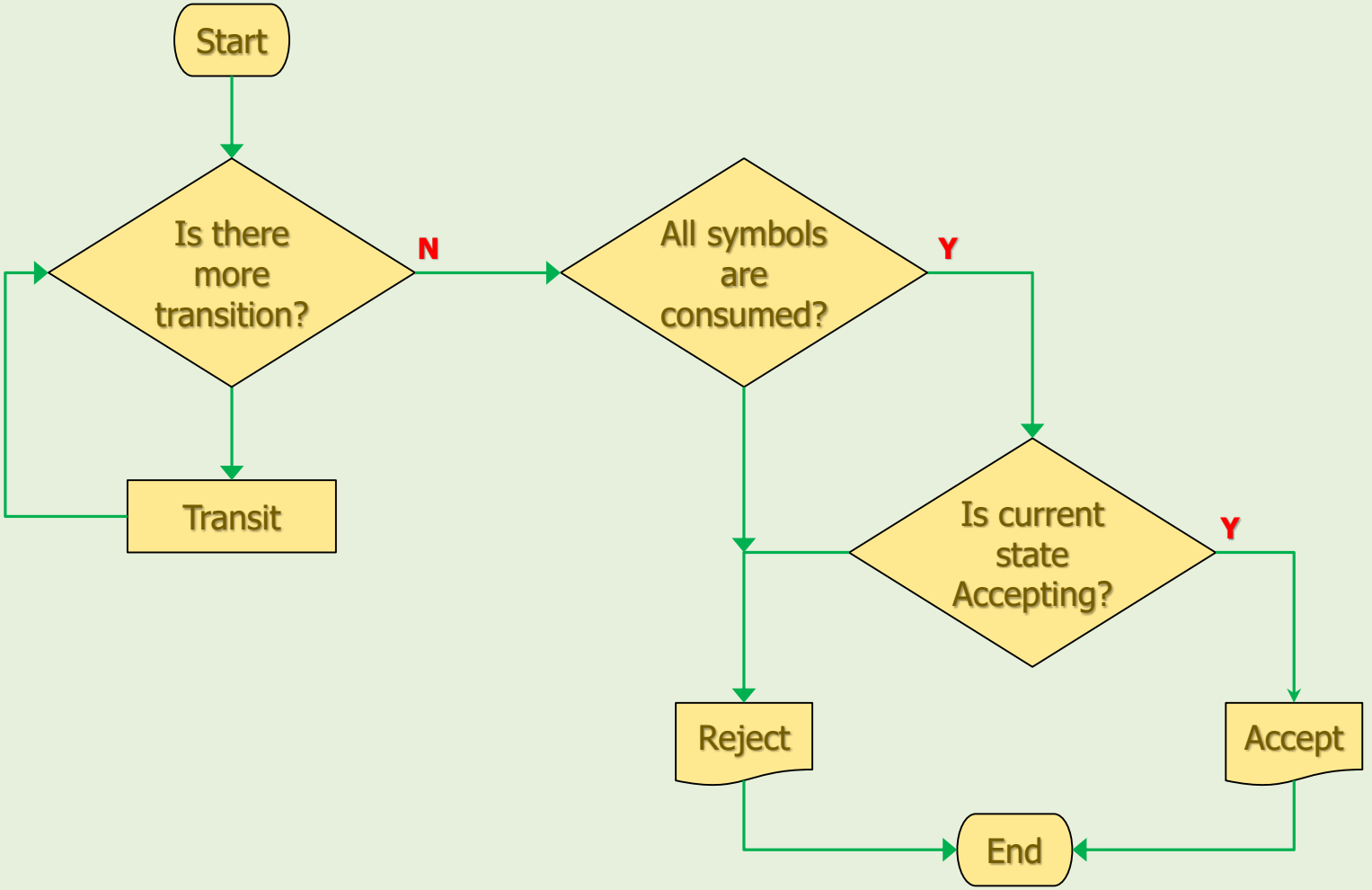
- Design a FIFOA to accept our famous language $L = \{a^n b^n : n \geq 0\}$ over $\Sigma = \{a, b\}$.

Solution

- **Strategy**: read a's and enqueue them in the queue.
- When the first b is sensed, start dequeuing a's to match them with b's.
- Continue dequeuing a's until you are out of b.
- If end of queue is reached, means the number of a's and b's are equal, so, accept the string,
  otherwise, reject it.



FIFOAs Operation Flowchart



6. Formal Definition of FIFOAs

- A FIFOA M is defined by the septuple (7-tuple):

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

- Where:
  - 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.
  - Γ is a finite and nonempty set of symbols called queue alphabet.
  - δ is called transition function and is defined as:

$$\delta: Q \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \rightarrow Q \times \Gamma^*$$

δ is total function.

  - $q_0 \in Q$ is the initial state of the transition graph.
  - $Z \in \Gamma$ is a special symbol called queue start symbol.
  - $F \subseteq Q$ is the set of accepting states of the transition graph.