CS311 Formal Language and Automata

Tingting Chen

Computer Science

Cal Poly Pomona

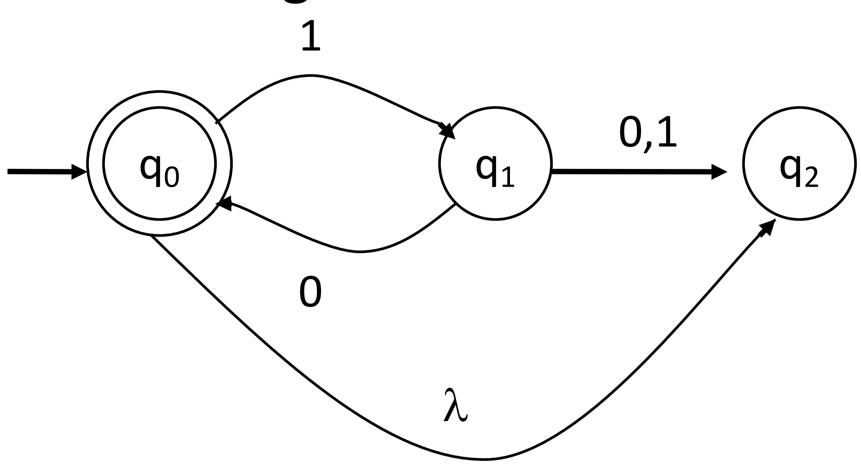
Nondeterministic Finite Automata

An NDFA can be non-deterministic by:

(1) having more than one edge with the same label originate from one vertex: see state q_1, which has two arcs labeled 0 emanating from it

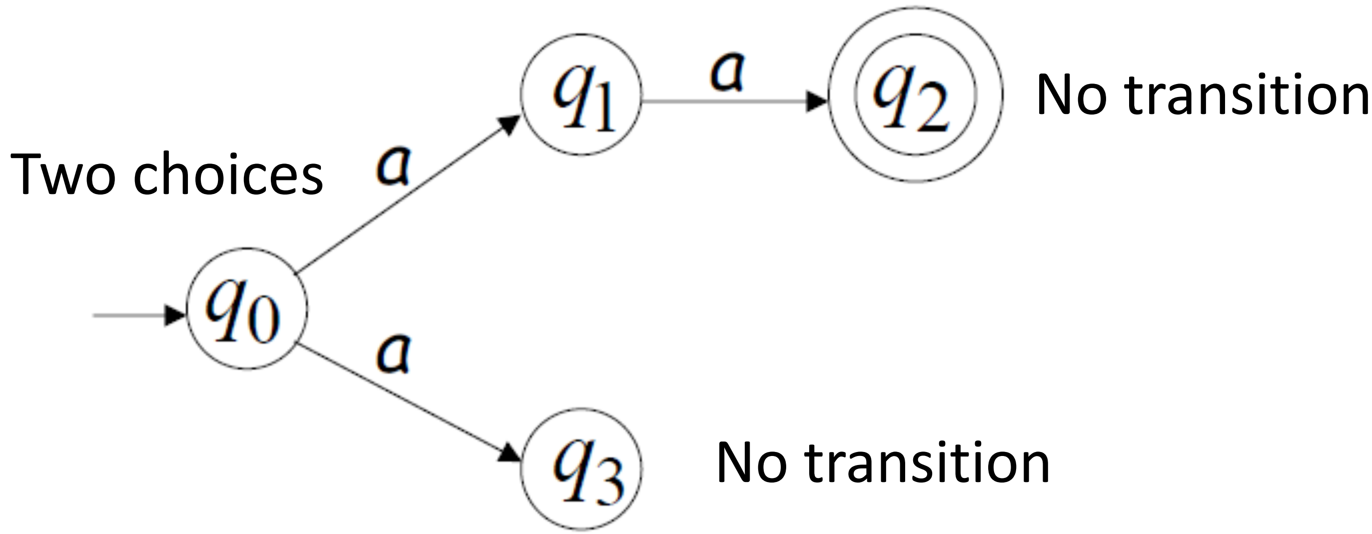
(2) having states without an edge originating from it for some symbol: see state q_2, which has no edges labeled 0 or 1. (This may be interpreted as a transition to the empty set.)

(3) having lambda-transitions: see state q_0, which has an arc indicating that a λ-move from q_0 to q_2 is possible



Nondeterministic Finite Automata (NFA)

Alphabet = {a}



Does it accept aa?
Does it accept a?

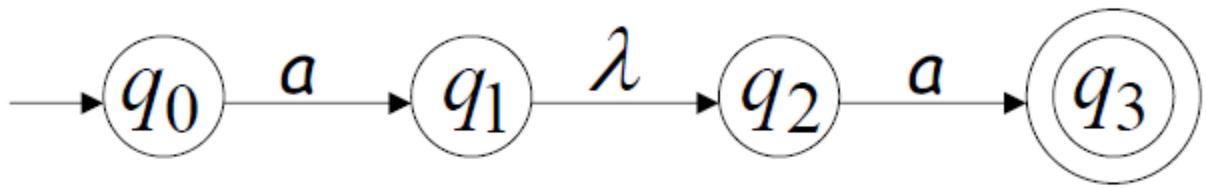
Nondeterministic Finite Automata (NFA)

- An NFA **accepts a string** when there is a computation of the NFA that accepts the string.
- There is a computation means that all the input is consumed and the automaton is in an accepting state.

- An NFA **rejects a string** when there is no computation of the NFA that accepts the string.
- For each computation:
- All the input is consumed and the automaton is in a non final state , OR
- The input cannot be consumed.

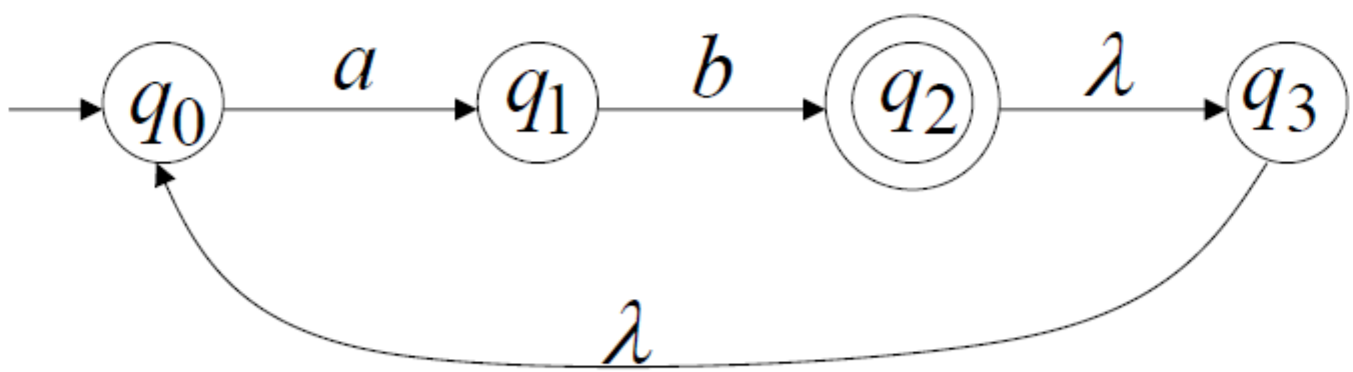
Lambda Transitions

- Example



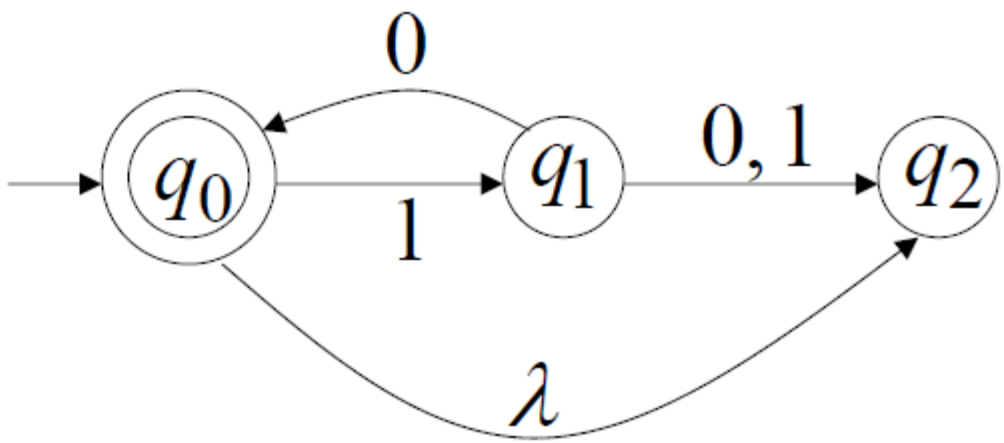
Does it accept aa?
Does it accept aaa?
The language that it accepts: {aa}.

Example 2: L(M)=?



Lambda Transitions

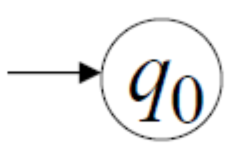
- Example 3: L(M) = ?



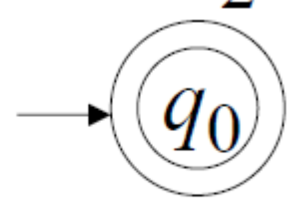
Redundant State

- The λ symbol never appears on the input tape.
- Consider two FAs, $L(M_1)$=?, $L(M_2)$=?

M_1

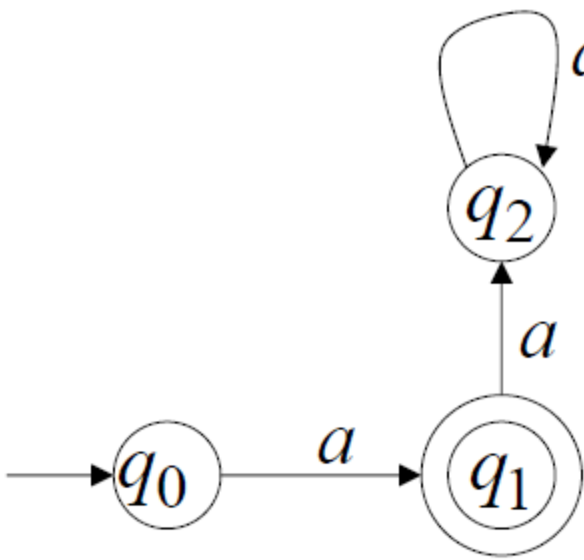


M_2

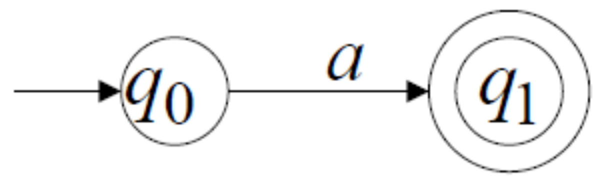


Why NFA?

- More natural when modeling a lot of real problems, such as in game theory.
- We can express languages easier than DFA.



$L(M_1) = \{a\}$



$L(M_2)=\{a\}$

Formal Definition of NFA

A non-deterministic finite accepter is defined by the quintuple: $M = (Q, \Sigma, \delta, q_0, F)$

Q is a finite, nonempty set of states

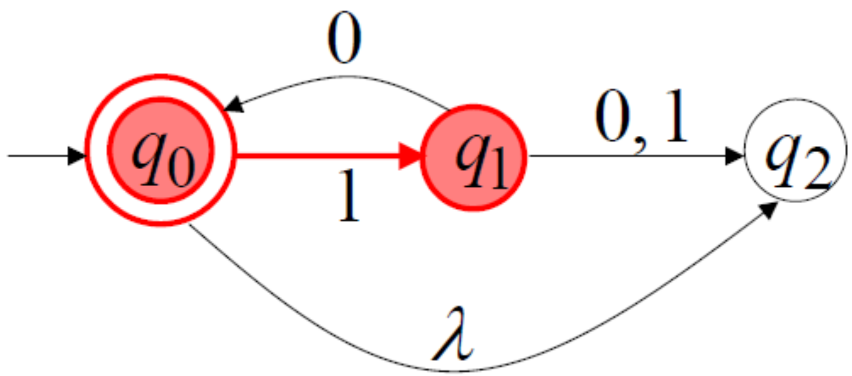
Σ is finite set of input symbols called alphabet

$\delta: Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$ is the transition function

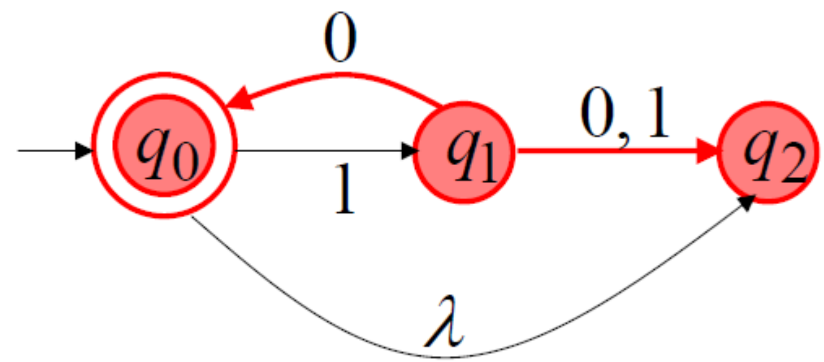
$q_0 \in Q$ is the initial state

$F \subseteq Q$ is a set of final or “accepting” states

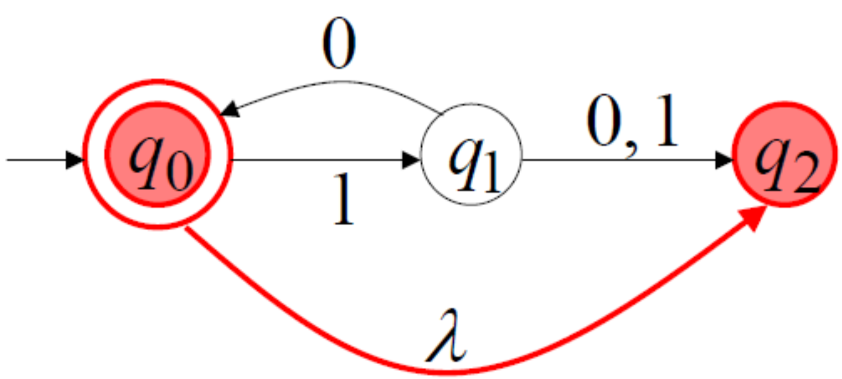
Transition Function Examples



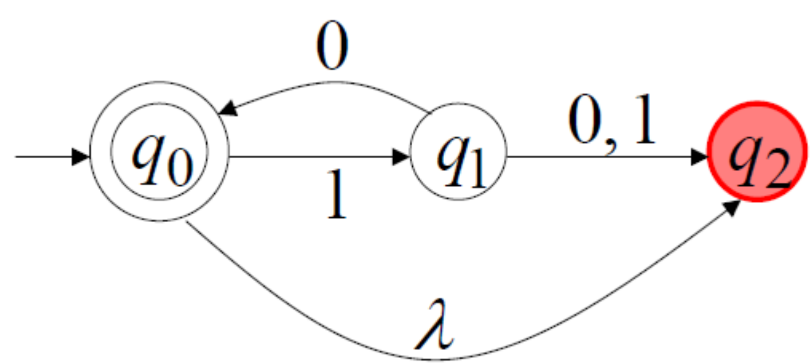
$\delta(q_0, 1) = \{q_1\}$



$\delta(q_1, 0) = \{q_0, q_2\}$

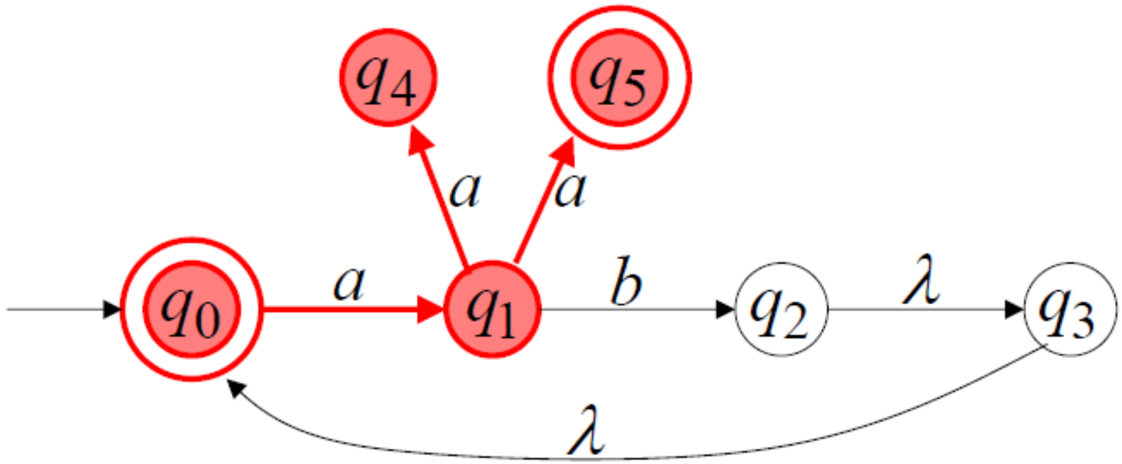


$\delta(q_0, \lambda) = \{q_0, q_2\}$

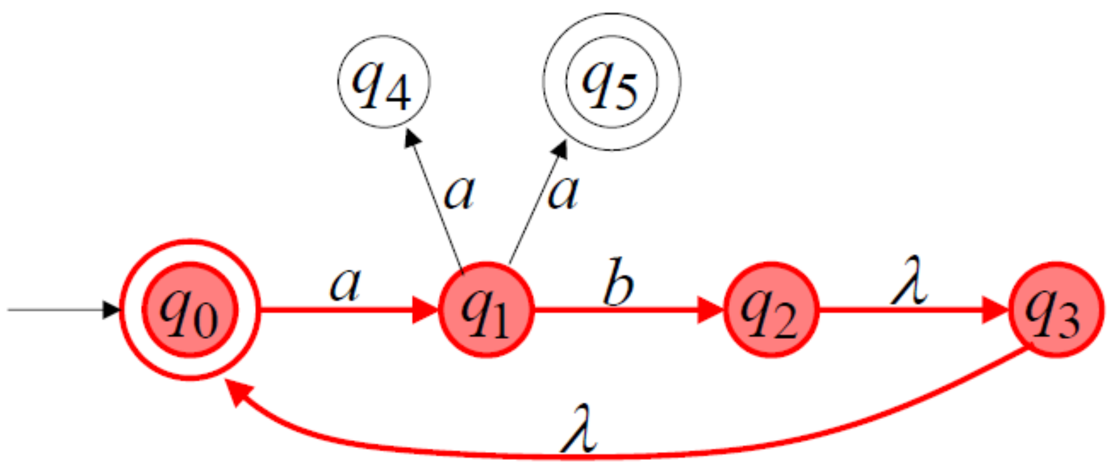


$\delta(q_2, 1) = \phi$

Extended Transition Function



$\delta*(q_0, aa) = \{q_4, q_5\}$



$\delta*(q_0, ab) = \{q_2, q_3, q_0\}$

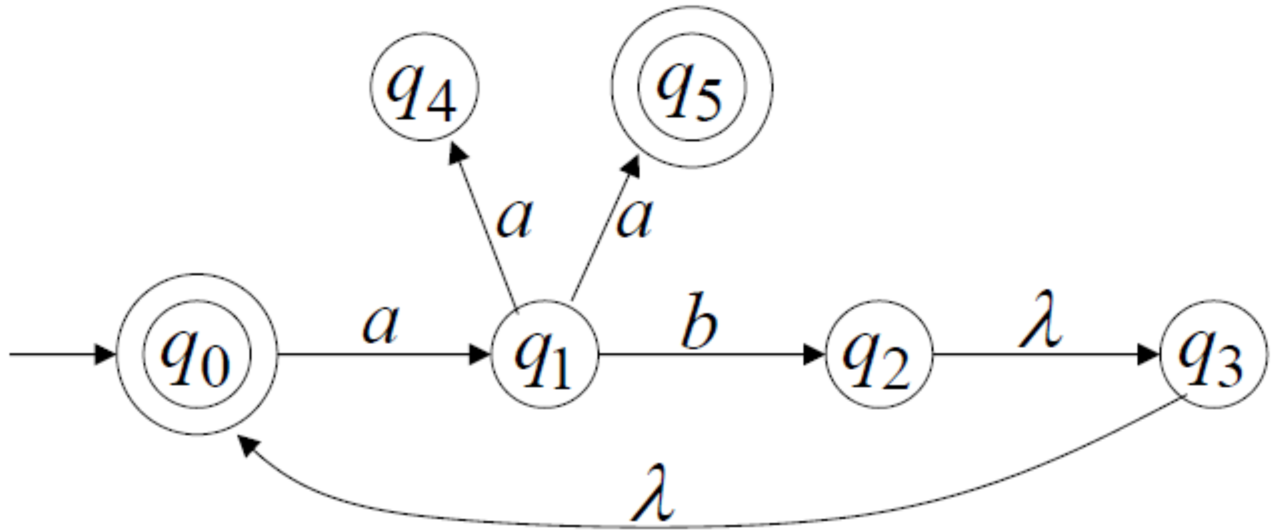
Formally

- The extended transition function for an NFA is defined so that $\delta^*(q_i, w)$ contains q_j, iff there is a walk in the transition graph from q_i to q_j labeled w.

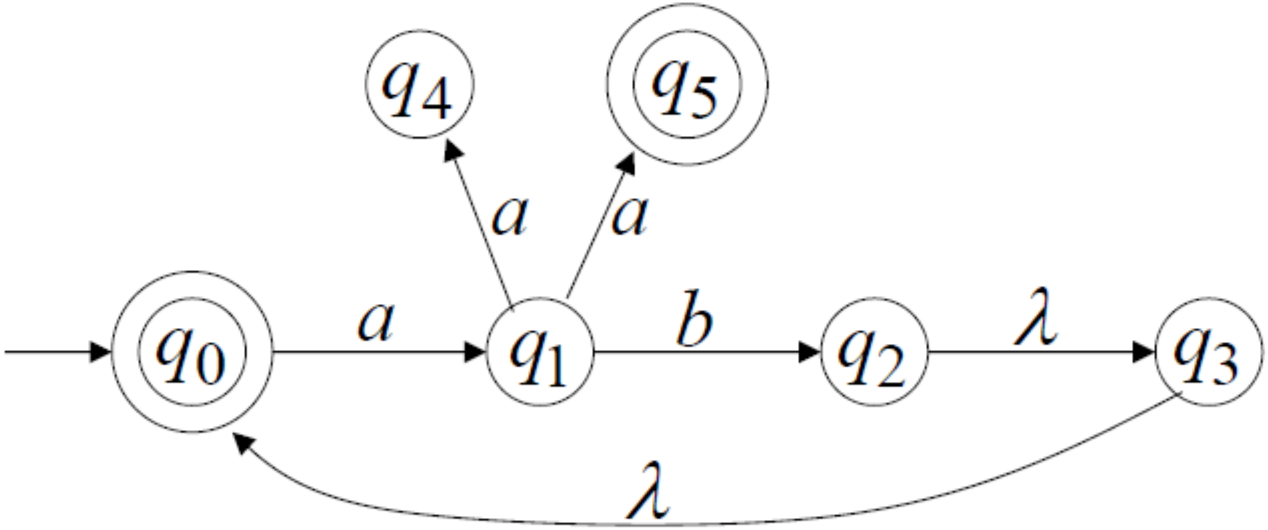
- The language L accepted by an NFA $M = (Q, \Sigma, \delta, q_0, F)$ is defined as
$$L(M) = \{w \in \Sigma^* : \delta^*(q_0, w) \cap F \neq \emptyset\}$$

- That is, the language consists of all strings such that for each of them there is a walk from the start state to a final state in the transition graph.

Exercise: L(M)=?



Exercise: L(M)=?



λ

aa

ab? $\delta^*(q_0, ab) = \{q_2, q_3, q_0\}$

abab

abaa

aba? $L(M)=\{ab\}^* \cup \{(ab)^n aa : n \geq 0\}$

Formally, again

- The language L accepted by an NFA M is

$$L(M) = \{w_1, w_2, w_3 \ldots\}$$

where $\delta^* (q_0, w_m) = \{q_i, q_j, \ldots, q_k, \ldots\}$ and

there is some $q_k \in F$ (final/accepting states).