COURSE: THEORY OF AUTOMATA COMPUTATION

TOPICS TO BE COVERED

- Non deterministic finite automata
- Language accepted by a NFA
- String accepted by Non Deterministic finite automata

NONDETERMINISTIC FINITE STATE AUTOMATA (NFA)

- An NFA is a five-tuple:

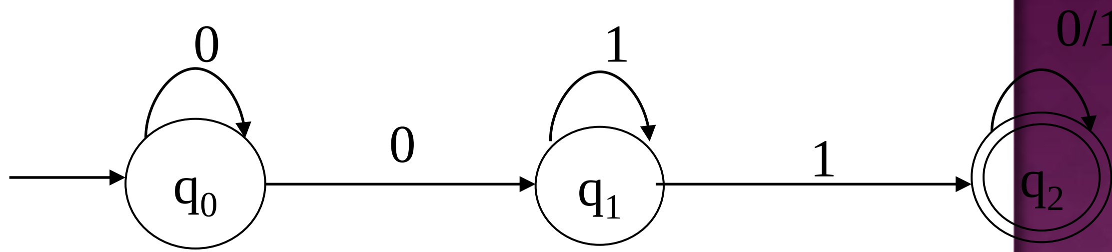
$M = (Q, \Sigma, \delta, q_0, F)$

Q	A finite set of states
Σ	A finite input alphabet
q_0	The initial/starting state, q_0 is in Q
F	A set of final/accepting states, which is a subset of Q
δ	A transition function, which is a total function from Q x Σ to 2^Q

$\delta: (Q \times \Sigma) \rightarrow \mathbf{2^Q}$	-2^Q is the power set of Q, the set of all subsets of Q
$\delta(q,s)$	-The set of all states p such that there is a transition labeled s from q to p

$\delta(q,s)$ is a function from Q x S to 2^Q (but not to Q)

- Example #1: some 0's followed by some 1's



$Q = \{q_0, q_1, q_2\}$

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

Start state is q_0

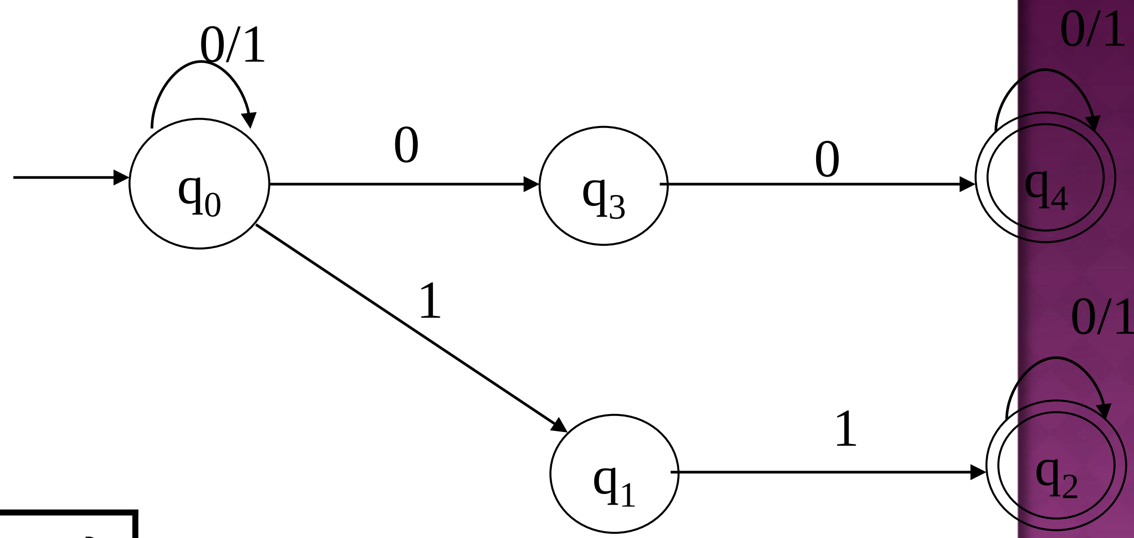
$F = \{q_2\}$

δ:

	0	1
q_0	$\{q_0, q_1\}$	{}
q_1	{}	$\{q_1, q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$

- Example #2: pair of 0's or pair of 1's

$Q = \{q_0, q_1, q_2, q_3, q_4\}$
$\Sigma = \{0, 1\}$
Start state is q_0
$F = \{q_2, q_4\}$



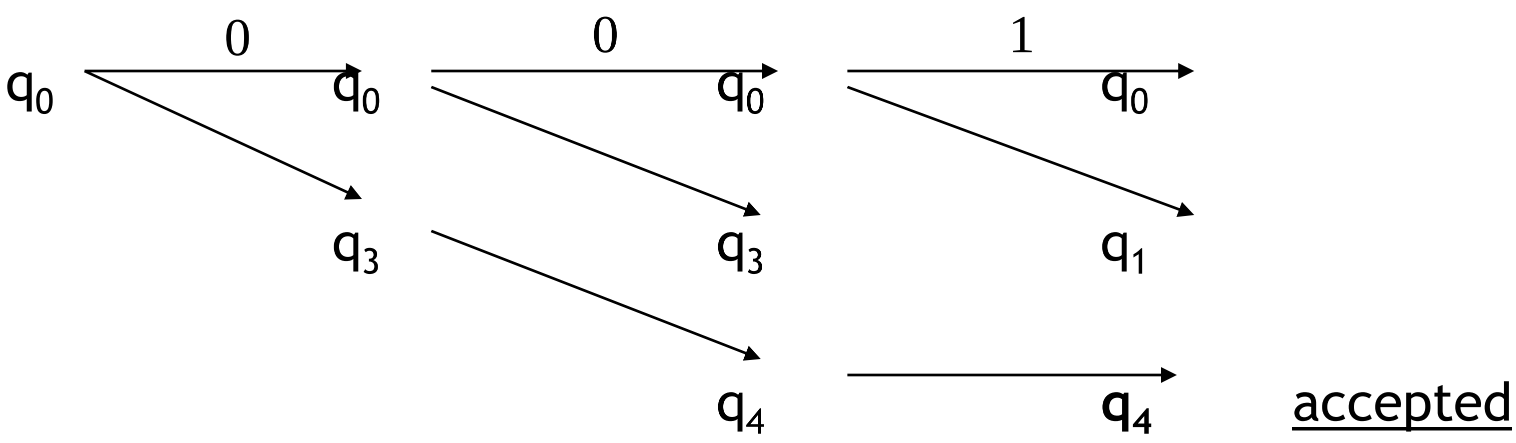
δ:

	0	1
q_0	$\{q_0, q_3\}$	$\{q_0, q_1\}$
q_1	$\{\}$	$\{q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$
q_3	$\{q_4\}$	$\{\}$
q_4	$\{q_4\}$	$\{q_4\}$

- Notes:
  - δ(q,s) may not be defined for some q and s (why?).
  - Informally, a string is said to be accepted *if there exists* a path to some state in F.
  - The language accepted by an NFA is the set of all accepted strings.

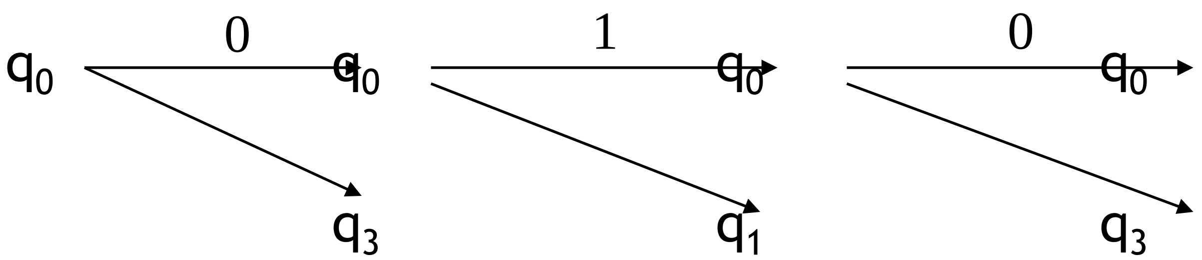
- Question: How does an NFA find the correct/accepting path for a given string?
  - NFAs are a non-intuitive computing model.
  - We are *primarily* interested in NFAs as language defining devices, i.e., do NFAs accept languages that DFAs do not?
  - Other questions are secondary, including practical questions such as whether or not there is an algorithm for finding an accepting path through an NFA for a given string,

- Determining if a given NFA (example #2) accepts a given string (001) can be done algorithmically:



- Each level will have at most n states

- Another example (010):



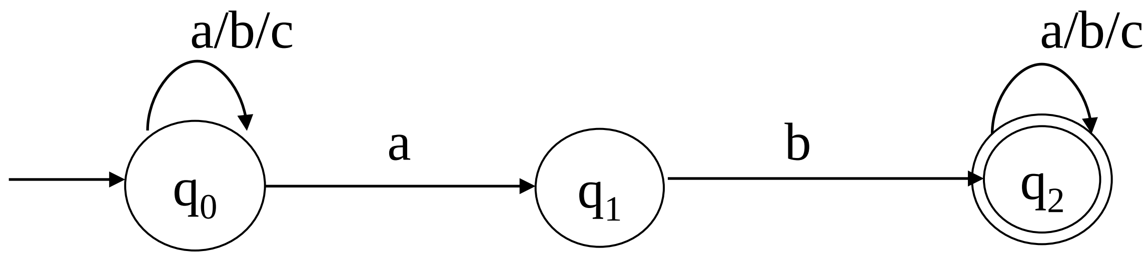
<u>not accepted</u>

- All paths have been explored, and none lead to an accepting state.

- Question: Why non-determinism is useful?
  - Non-determinism = Backtracking
  - Non-determinism hides backtracking
  - Programming languages, e.g., Prolog, hides backtracking => Easy to program at a higher level: what we want to do, rather than how to do it
  - Useful in complexity study
  - Is NDA more "powerful" than DFA, i.e., accepts type of languages that any DFA cannot?

- Let Σ = {a, b, c}. Give an NFA M that accepts:

  $L = \{x \mid x \text{ is in } \Sigma^* \text{ and } x \text{ contains ab}\}$



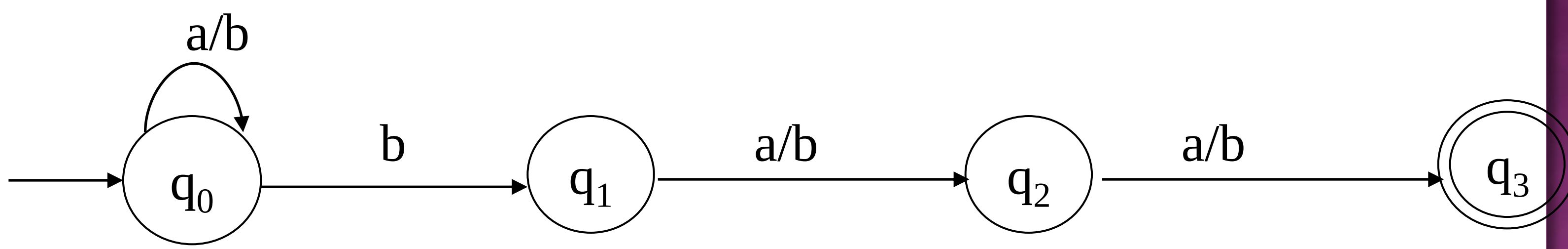
  Is L a subset of L(M)?

  Is L(M) a subset of L?

- Is an NFA necessary? Could a DFA accept L? Try and give an equivalent DFA as an exercise.
- Designing NFAs is not trivial: easy to create bug

- Let $\Sigma = \{a, b\}$. Give an NFA M that accepts:

  $L = \{x \mid x \text{ is in } \Sigma^* \text{ and the third to the last symbol in x is b}\}$



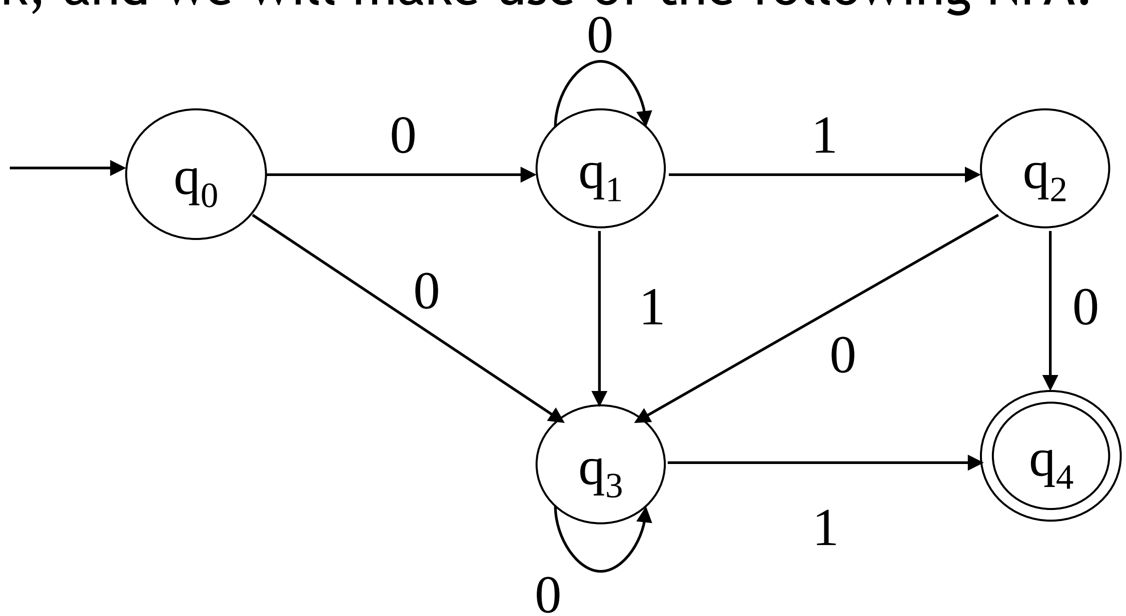
  Is L a subset of L(M)?

  Is L(M) a subset of L?

- Give an equivalent DFA as an exercise.

EXTENSION OF Δ TO STRINGS AND SETS OF STATES

- What we currently have: $\delta : (Q \times \Sigma) \rightarrow 2^Q$
- What we want (why?): $\delta : (2^Q \times \Sigma^*) \rightarrow 2^Q$
- We will do this in two steps, which will be slightly different from the book, and we will make use of the following NFA.



Extension of δ to Strings and Sets of States

- Step #1:

  Given δ: $(Q \times \Sigma) \rightarrow 2^Q$ define $\delta^\#$: $(2^Q \times \Sigma) \rightarrow 2^Q$ as follows:

  1) $\delta^\#(R, a) = \bigcup_{q \in R} \delta(q, a)$ for all subsets R of Q, and symbols a in Σ

- Note that:

  $\delta^\#(\{p\},a) = \bigcup_{q \in \{p\}} \delta(q, a)$ by definition of $\delta^\#$, rule #1 above

  $= \delta(p, a)$

- Hence, we can use δ for $\delta^\#$

  $\delta(\{q_0, q_2\}, 0)$ These now make sense, but previously

  $\delta(\{q_0, q_1, q_2\}, 0)$ they did not.

- Example:

$$\delta(\{q_0, q_2\}, 0) = \delta(q_0, 0) \cup \delta(q_2, 0)$$
$$= \{q_1, q_3\} \cup \{q_3, q_4\}$$
$$= \{q_1, q_3, q_4\}$$

$$\delta(\{q_0, q_1, q_2\}, 1) = \delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1)$$
$$= \{\} \cup \{q_2, q_3\} \cup \{\}$$
$$= \{q_2, q_3\}$$

- Step #2:

Given δ: $(2^Q \times \Sigma) \rightarrow 2^Q$ define $\delta^\wedge$: $(2^Q \times \Sigma^*) \rightarrow 2^Q$ as follows:

$\delta^\wedge(R,w)$ – The set of states M could be in after processing string w, having starting from any state in R.

Formally:

2) $\delta^\wedge(R, \varepsilon) = R$	for any subset R of Q
3) $\delta^\wedge(R,wa) = \delta(\delta^\wedge(R,w), a)$	for any w in Σ^*, a in Σ, and subset R of Q

- Note that:

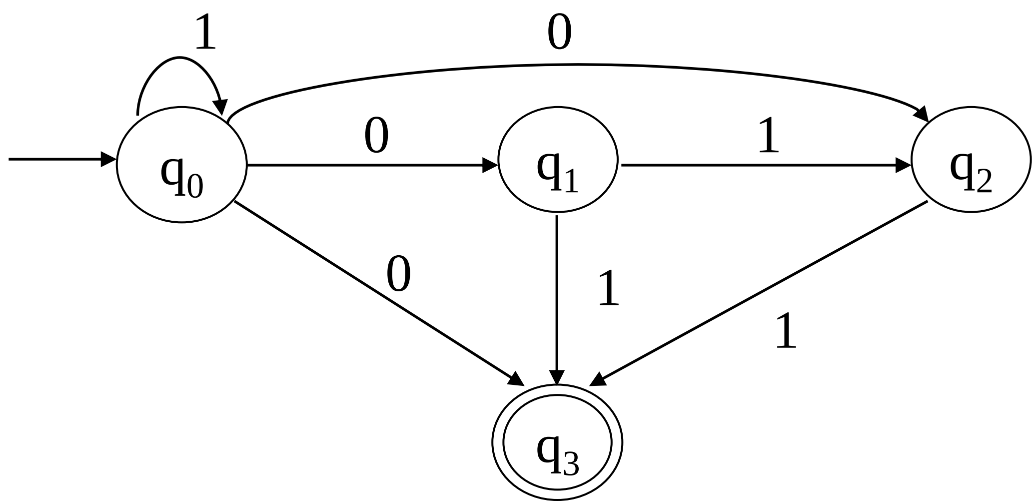
$\delta^\wedge(R, a)$	$= \delta(\delta^\wedge(R, \varepsilon), a)$	by definition of $\delta^\wedge$, rule #3 above
	$= \delta(R, a)$	by definition of $\delta^\wedge$, rule #2 above

- Hence, we can use δ for $\delta^\wedge$

$\delta(\{q_0, q_2\}, 0110)$
$\delta(\{q_0, q_1, q_2\}, 101101)$

These now make sense, but previously they did not.

- Example:



What is $\delta(\{q_0\}, 10)$?

Informally: The set of states the NFA could be in after processing 10, having started in state q_0, i.e., $\{q_1, q_2, q_3\}$.

Formally: $\delta(\{q_0\}, 10) = \delta(\delta(\{q_0\}, 1), 0)$

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

$= \{q_1, q_2, q_3\}$

Is 10 accepted? Yes!

- Example:

What is $\delta(\{q_0, q_1\}, 1)$?

$$\begin{aligned}\delta(\{q_0, q_1\}, 1) &= \delta(\{q_0\}, 1) \cup \delta(\{q_1\}, 1) \\ &= \{q_0\} \cup \{q_2, q_3\} \\ &= \{q_0, q_2, q_3\}\end{aligned}$$

What is $\delta(\{q_0, q_2\}, 10)$?

$$\begin{aligned}\delta(\{q_0, q_2\}, 10) &= \delta(\delta(\{q_0, q_2\}, 1), 0) \\ &= \delta(\delta(\{q_0\}, 1) \cup \delta(\{q_2\}, 1), 0) \\ &= \delta(\{q_0\} \cup \{q_3\}, 0) \\ &= \delta(\{q_0, q_3\}, 0) \\ &= \delta(\{q_0\}, 0) \cup \delta(\{q_3\}, 0) \\ &= \{q_1, q_2, q_3\} \cup \{\} \\ &= \{q_1, q_2, q_3\}\end{aligned}$$

- Example:

$$
\begin{aligned}
\delta(\{q_0\}, 101) &= \delta(\delta(\{q_0\}, 10), 1) \\
&= \delta(\delta(\delta(\{q_0\}, 1), 0), 1) \\
&= \delta(\delta(\{q_0\}, 0), 1) \\
&= \delta(\{q_1 , q_2, q_3\}, 1) \\
&= \delta(\{q_1\}, 1) \cup \delta(\{q_2\}, 1) \cup \delta(\{q_3\}, 1) \\
&= \{q_2, q_3\} \cup \{q_3\} \cup \{\} \\
&= \{q_2, q_3\}
\end{aligned}
$$

Is 101 accepted? Yes!

DEFINITIONS FOR NFAS

- Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA and let w be in Σ^*. Then w is *accepted* by M iff $\delta(\{q_0\}, w)$ contains at least one state in F.

- Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA. Then the *language accepted* by M is the set:

  $L(M) = \{w \mid w \text{ is in } \Sigma^* \text{ and } \delta(\{q_0\},w) \text{ contains at least one state in } F\}$

- Another equivalent definition:

  $L(M) = \{w \mid w \text{ is in } \Sigma^* \text{ and } w \text{ is accepted by } M\}$