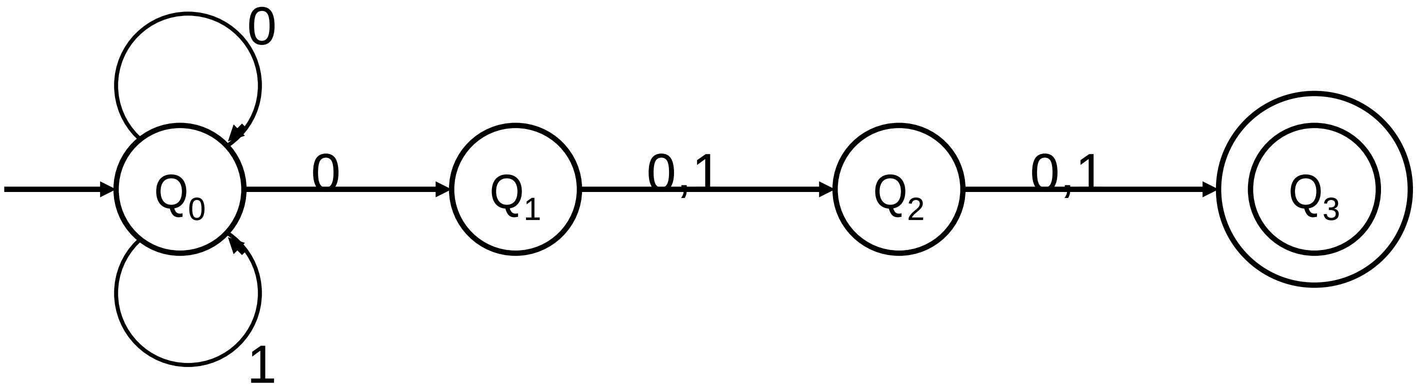
Nondeterministic Finite Automata (NFA)

- When an NFA receives an input symbol a, it can make a transition to a number (including 0) of states (each state can have multiple edges labeled with the same symbol).

- An NFA accepts a string w iff there exists a path labeled w from the initial state to one of the final states.

Example N1

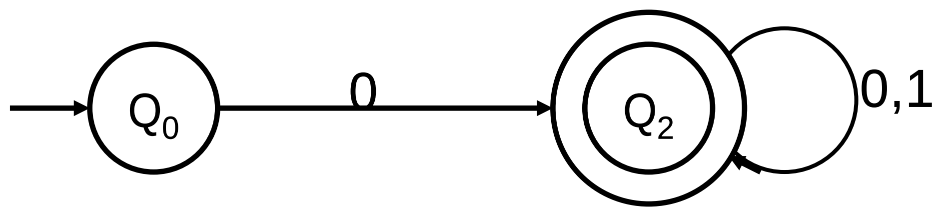
- The language of the following NFA consists of all strings over `{0,1}` whose 3rd symbol from the right is `0`.



- Note Q_0 has multiple transitions on 0

Example N2

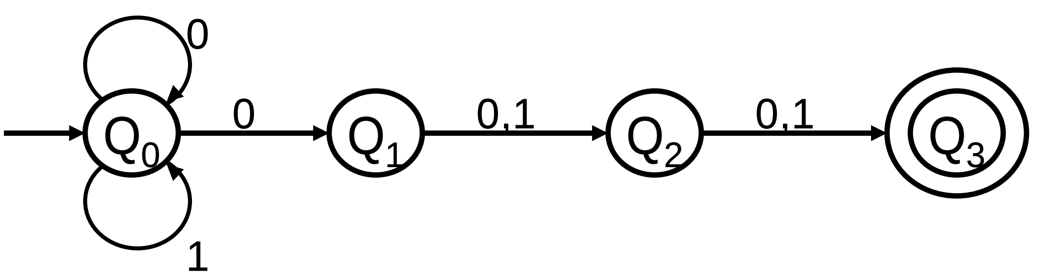
- The NFA N_2 accepts strings beginning with `0`.



- Note Q_0 has no transition on 1

NFA Processing

- Suppose N_1 receives the input string `0011`. There are three possible execution sequences:
- $q_0 \longrightarrow q_0 \longrightarrow q_0 \longrightarrow q_0 \longrightarrow q_0$
- $q_0 \longrightarrow q_0 \longrightarrow q_1 \longrightarrow q_2 \longrightarrow q_3$
- $q_0 \longrightarrow q_1 \longrightarrow q_2 \longrightarrow q_3$



- Only the second finishes in an accept state. The third even gets stuck (cannot even read the fourth symbol).

Implementation

- Implementation of NFAs has to be deterministic, using some form of backtracking to go through all possible executions .
- Any thoughts on how this might be accomplished?

Formal Definition

- An NFA is a quintuple $A=(Q,\Sigma,s,F,\delta)$, where the first four components are as in a DFA, and the transition function takes values in $P(Q)$ instead of Q. Thus
  - $\delta: Q \times \Sigma \longrightarrow P(Q)$
- The extension $\underline{\delta} : Q \times \Sigma^* \longrightarrow P(Q)$ is defined by
  - $\underline{\delta}(q,\varepsilon)=\{q\}$
  - $\underline{\delta}(q,ua)$ is the union of the sets $\delta(p,a)$, where p varies over all states in $\underline{\delta}(q,u)$
  - $\cup_{p\in \underline{\delta}(q,u)} \delta(p,a)$,

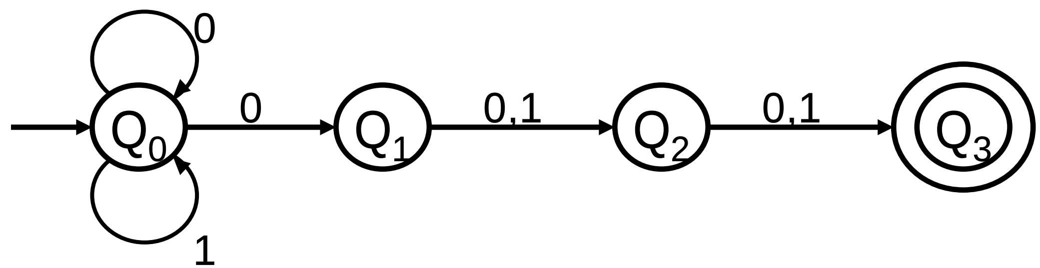
NFA Acceptance

- An NFA accepts a string `w` iff $\underline{\delta}$`(s,w)` contains a final state. The language of an NFA `N` is the set `L(N)` of accepted strings:

- $\mathbf{L(N) = \{w \mid \underline{\delta}(s,w) \cap F \neq \varnothing\}}$

compute $\underline{\delta}(q_0, 000)$

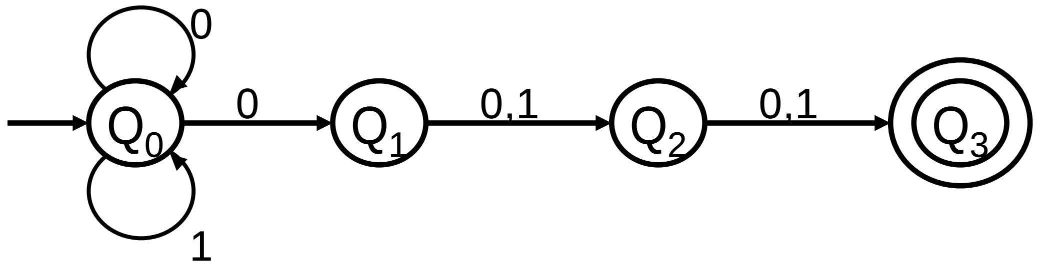
- $\underline{\delta}(q, ua) = \cup_{p \in \underline{\delta}(q,u)} \delta(p, a)$



- $\underline{\delta}(q_0, 000) = \cup_{x \in \underline{\delta}(q0,00)} \delta(x, 0)$
- $\underline{\delta}(q_0, 00) = \cup_{y \in \underline{\delta}(q0,0)} \delta(y, 0)$
- $\underline{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$
- $\underline{\delta}(q_0, 00) = \cup_{y \in \{q0,q1\}} \delta(y, 0)$
- $\underline{\delta}(q_0, 00) = \{q_0, q_1\} \cup \{q_2\} = \{q_0, q_1, q_2\}$
- $\underline{\delta}(q_0, 000) = \cup_{x \in \{q0,q1,q2\}} \delta(x, 0)$
- $\underline{\delta}(q_0, 000) = \{q_0, q_1\} \cup \{q_2\} \cup \{q_3\}$
- $\underline{\delta}(q_0, 000) = \{q_0, q_1, q_2, q_3\}$

Intuition

- At any point in the walk over a string, such as “000” the machine can be in a set of states.
- To take the next step, on a character ‘c’, we create a new set of states. Those reachable from the old set on a single ‘c’



	0	1
{Q0}	{Q0,Q1}	{Q0}
{Q0,Q1}	{Q0,Q1,Q2}	{Q0,Q2}
{Q0,Q2}	{Q0,Q1,Q3}	{Q0,Q3}
{Q0,Q1,Q3}	?	?
{Q0,Q3}	?	?