CS 6041
Theory of Computation

Nondeterministic finite automata

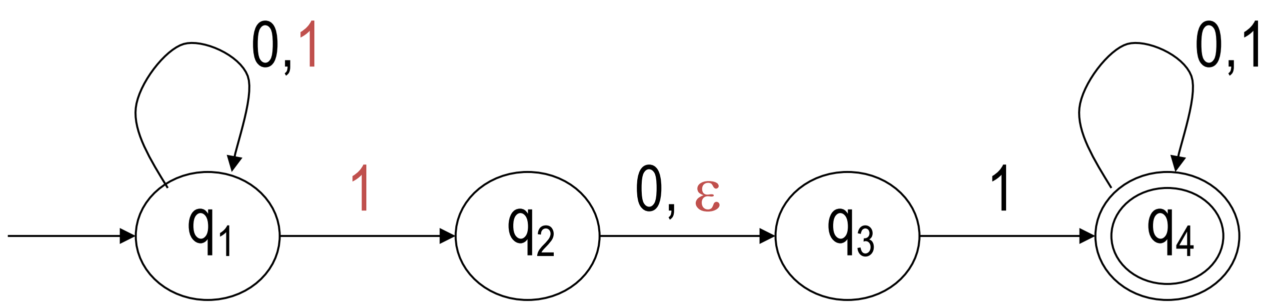
Kun Suo

Computer Science, Kennesaw State University

https://kevinsuo.github.io/

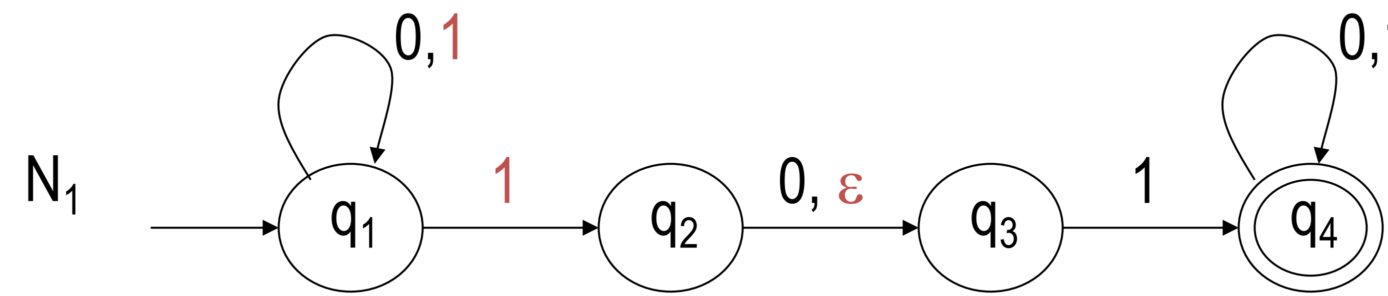
Nondeterminism

- ***Deterministic***:
  - Next state is unique
- ***Nondeterministic***:
  - Next state is not unique
  - ε move

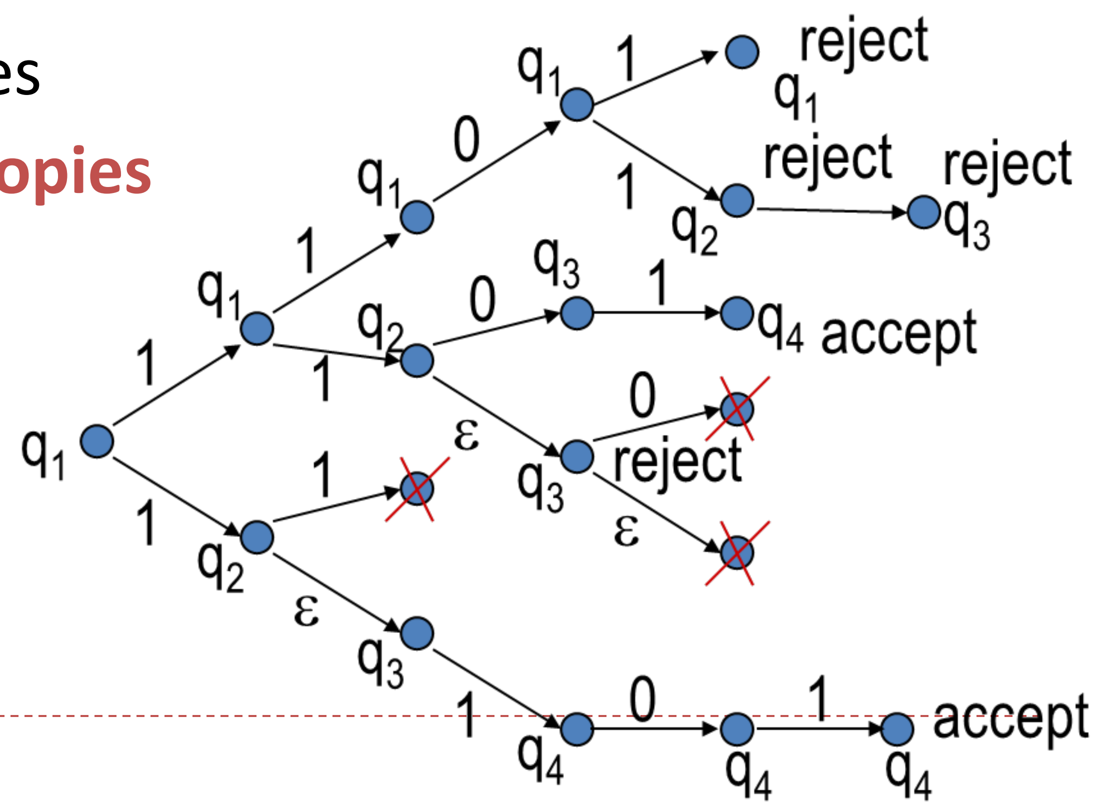


N_1

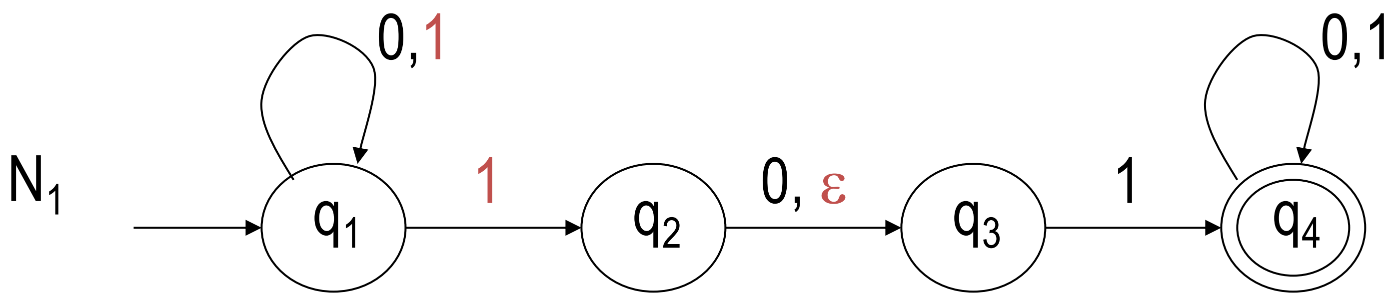
Nondeterminism



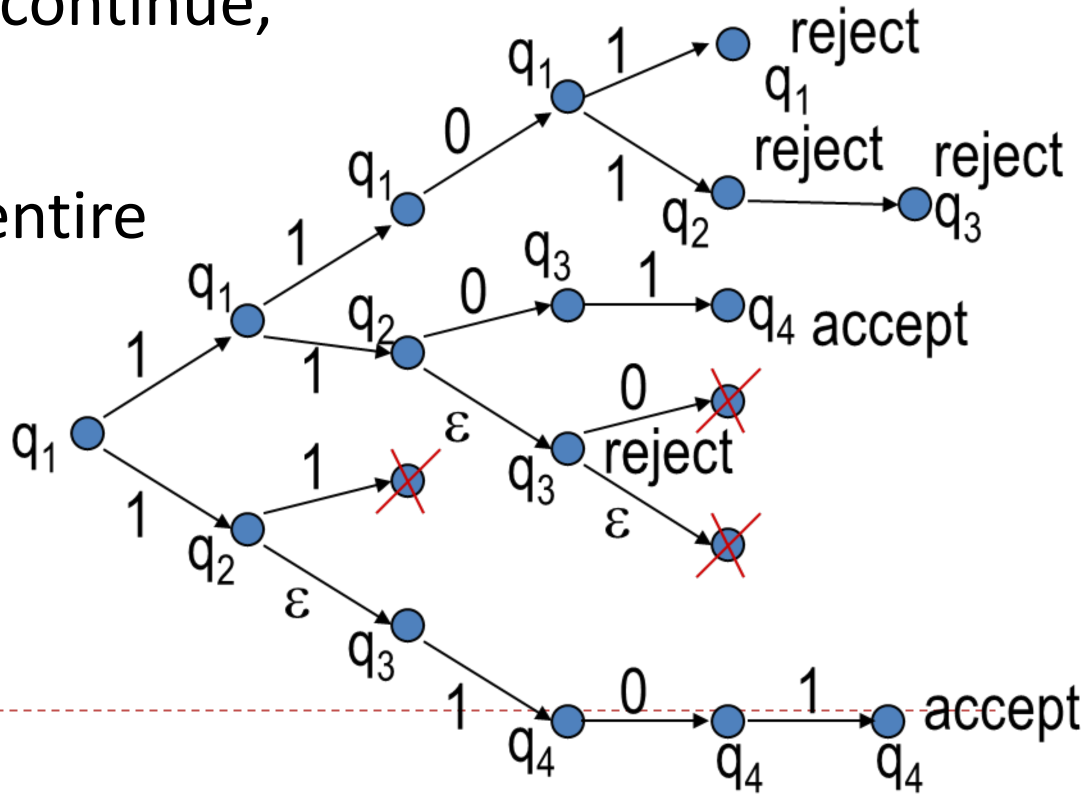
- ε and different choices generates **different copies**



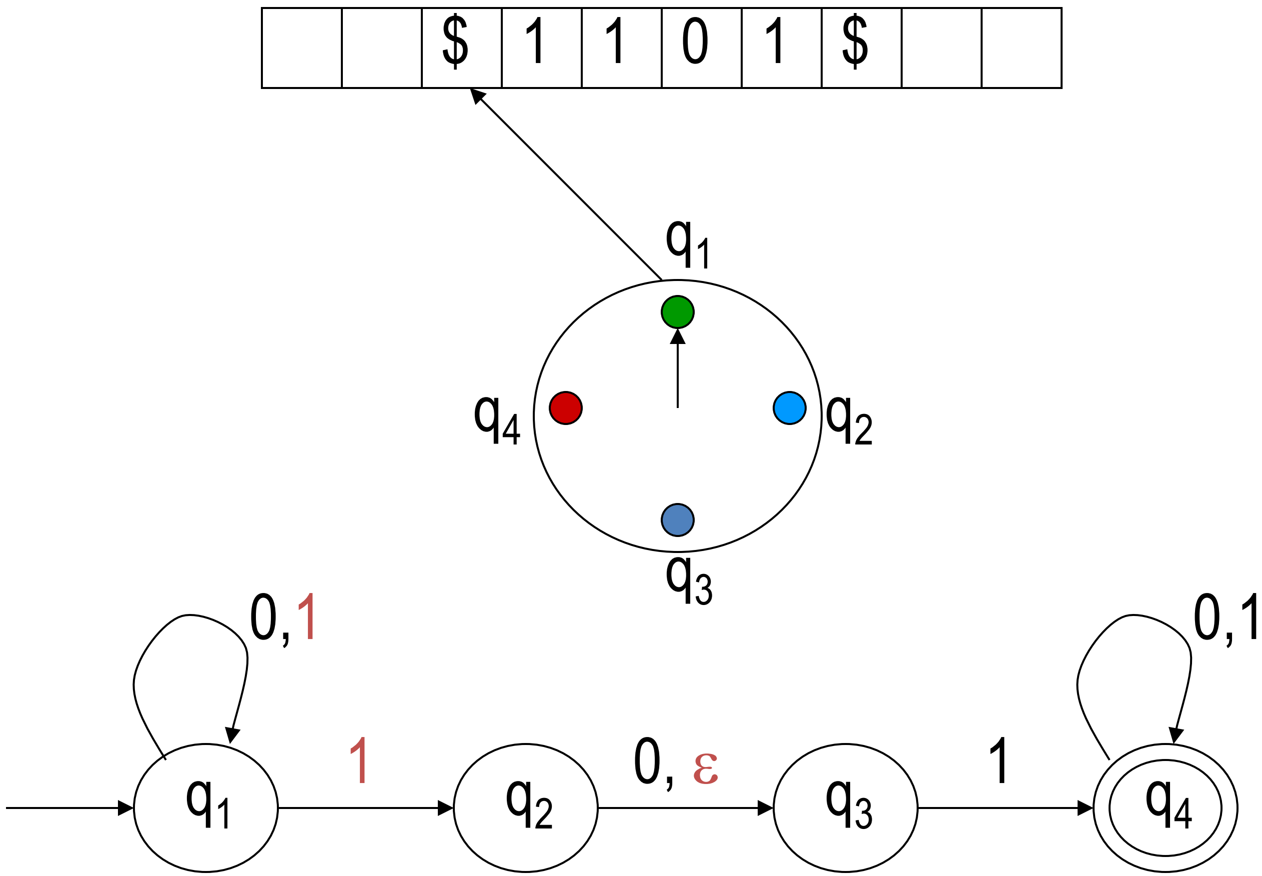
Nondeterminism



- When the move cannot continue, the copy will **disappear**
- If one copy accept, the entire computation **accept**

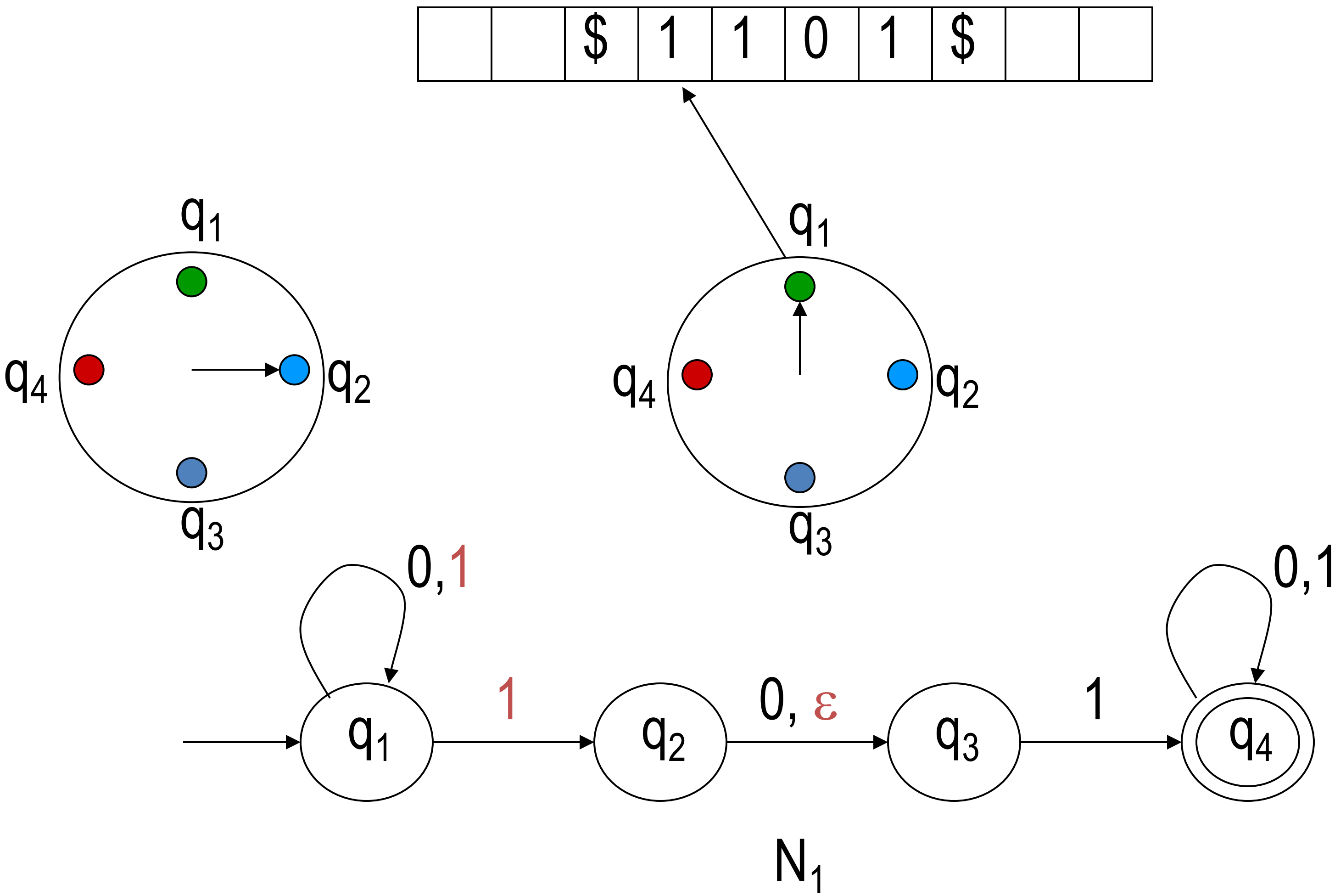


N_1 on input 1101 (0)

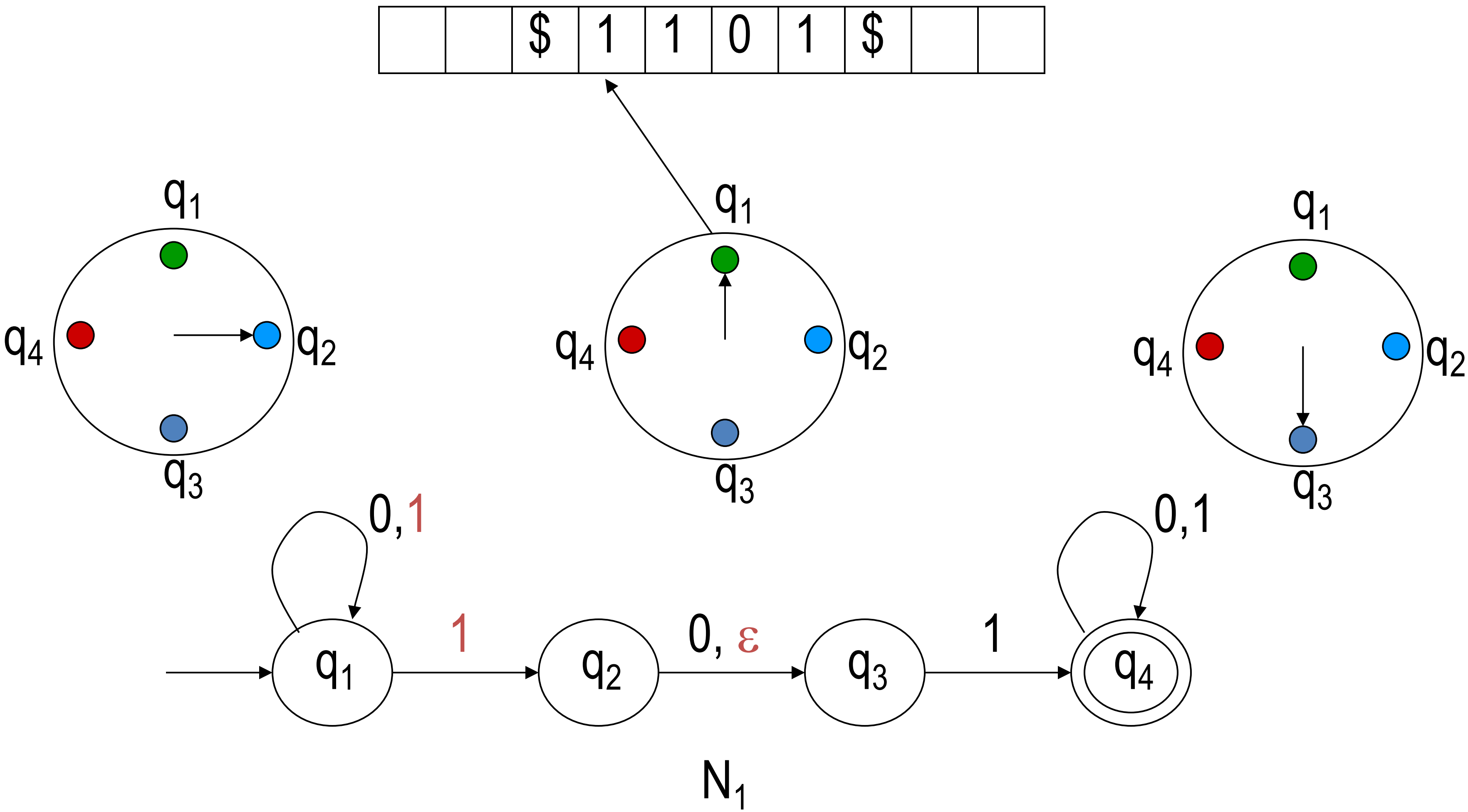


N_1

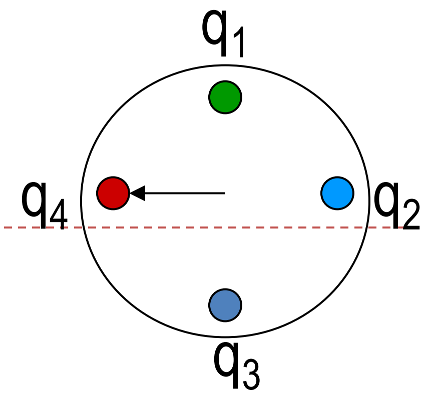
N_1 on input 1101 (1)

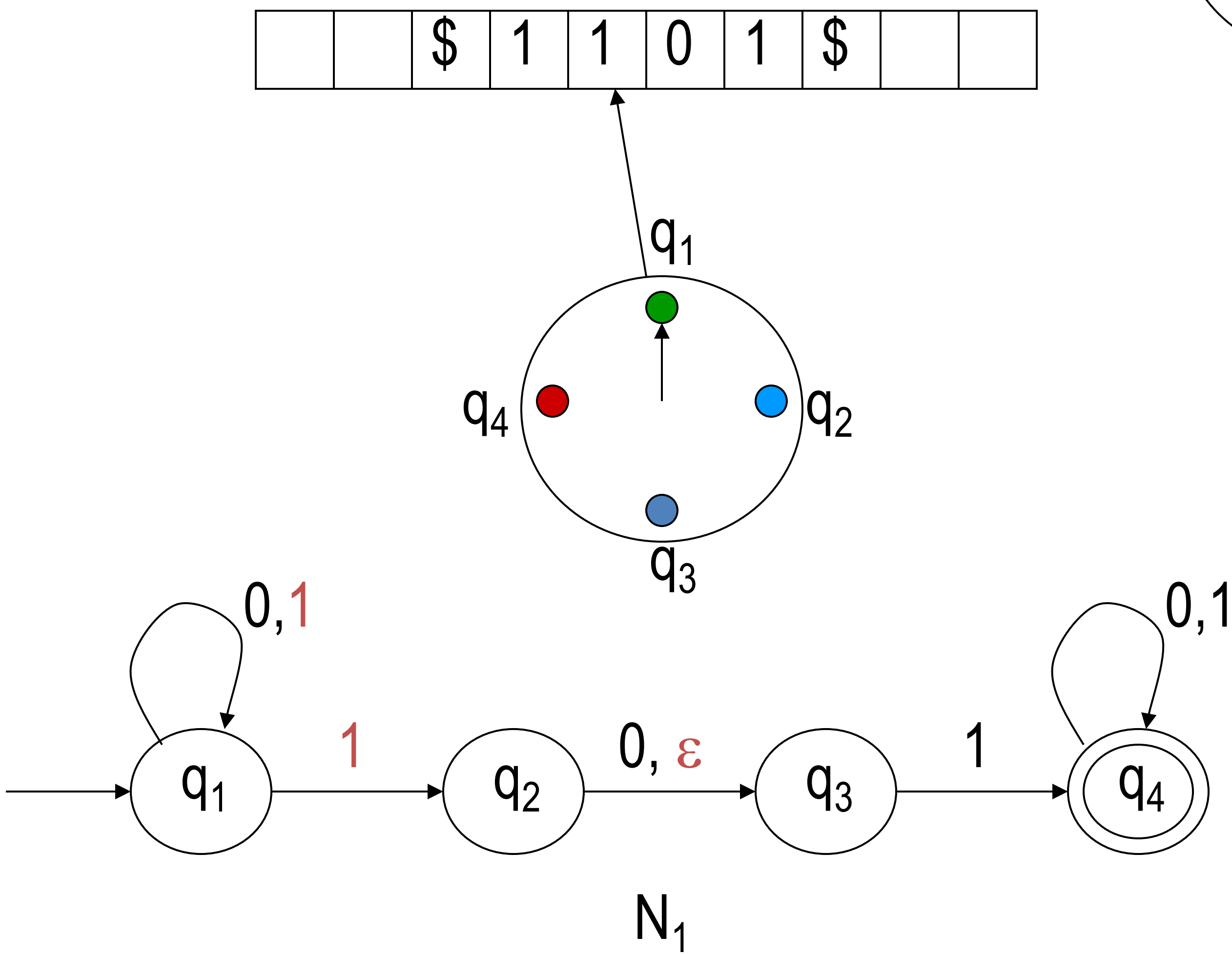


N_1 on input 1101 (1)

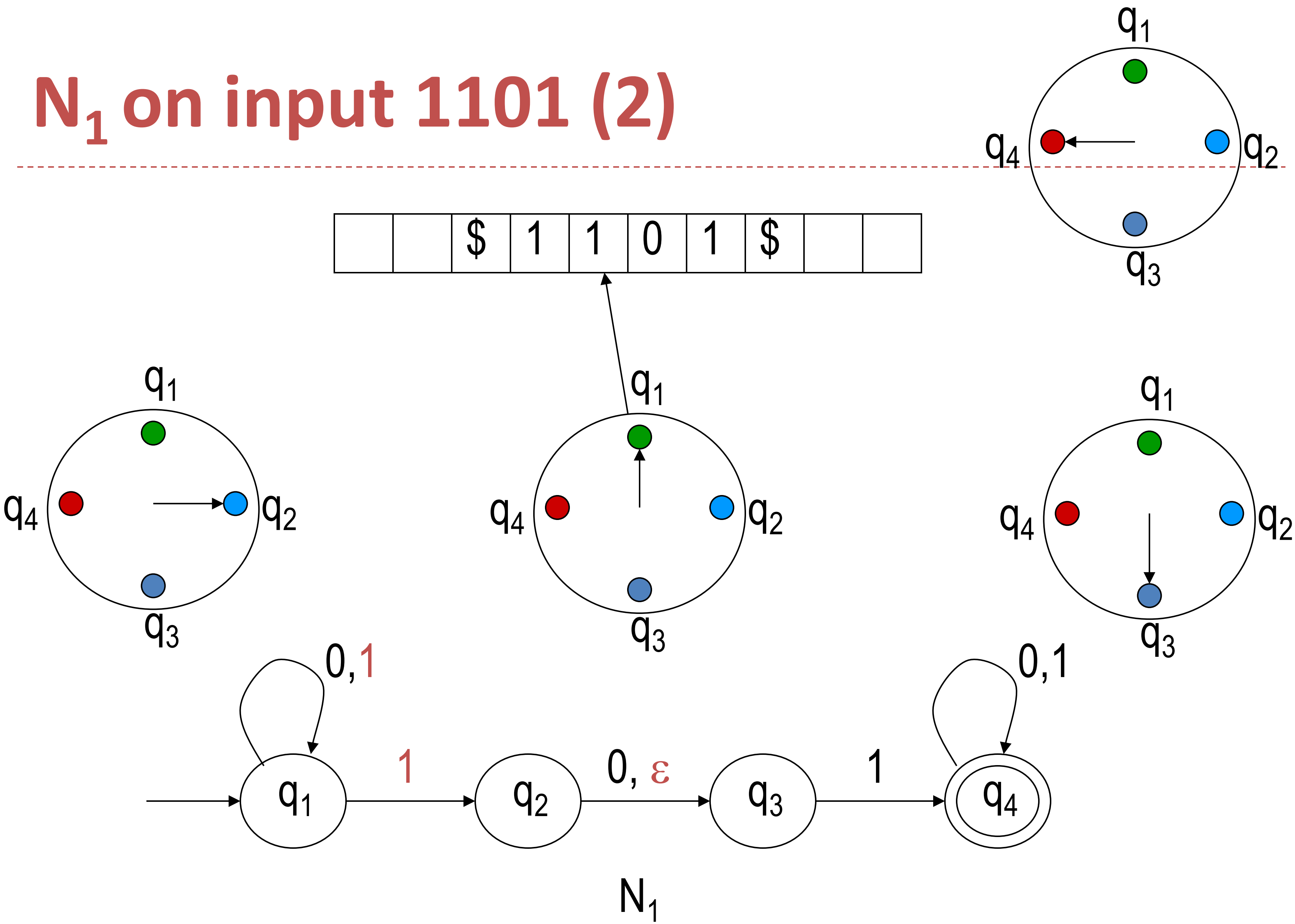


N_1 on input 1101 (2)

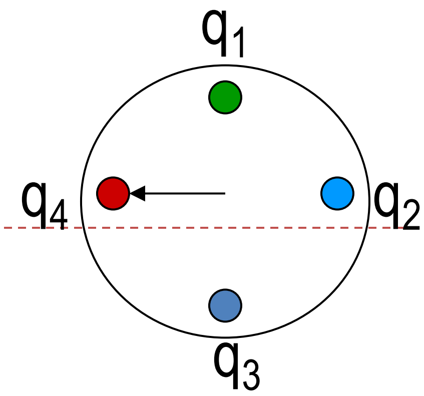


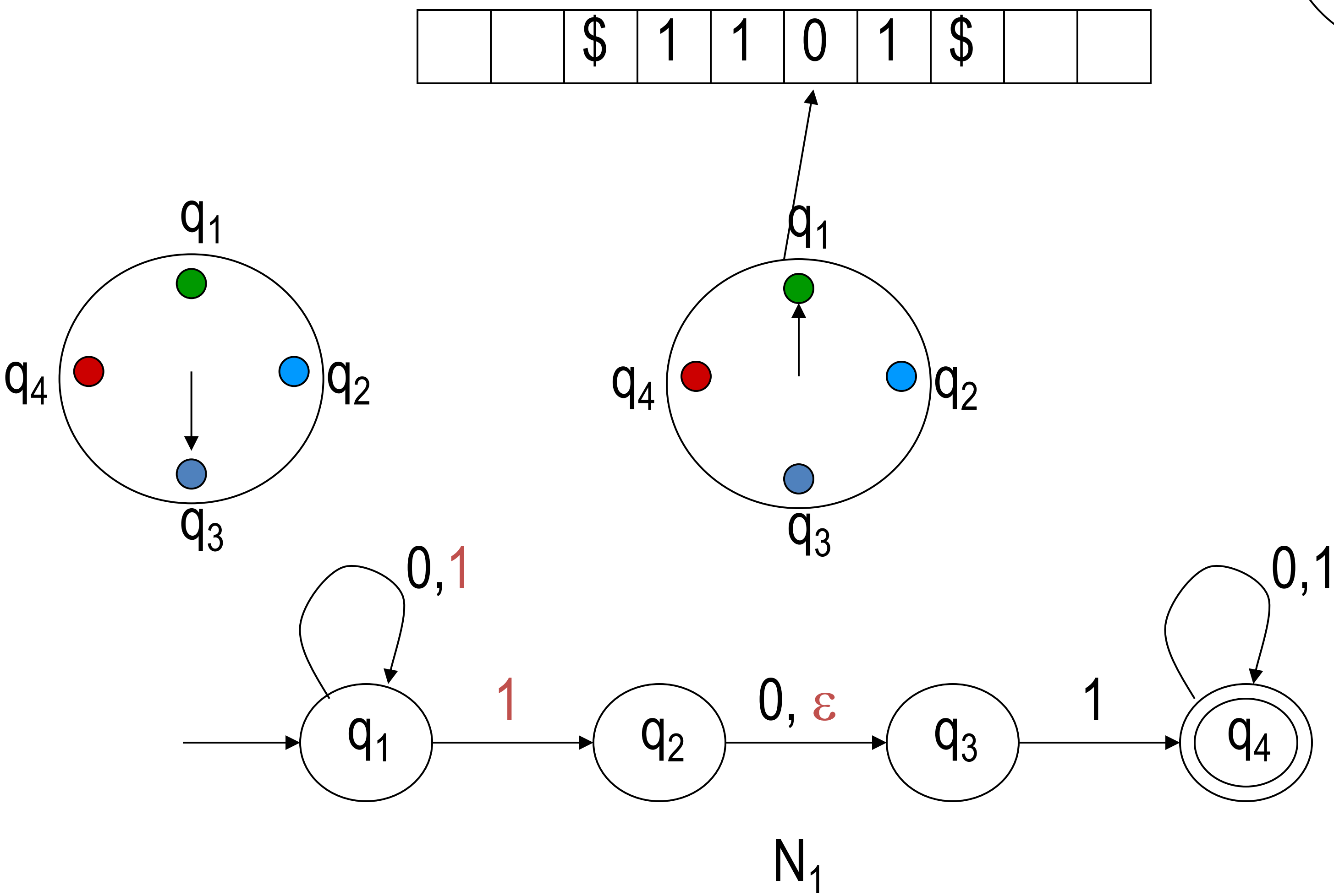


N_1 on input 1101 (2)

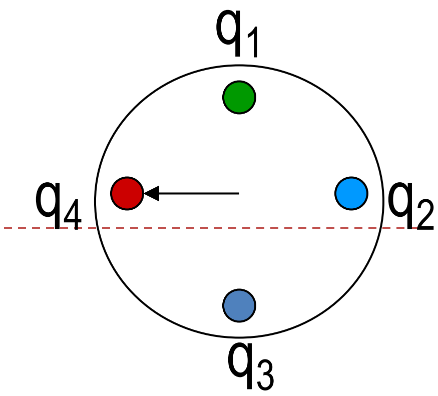


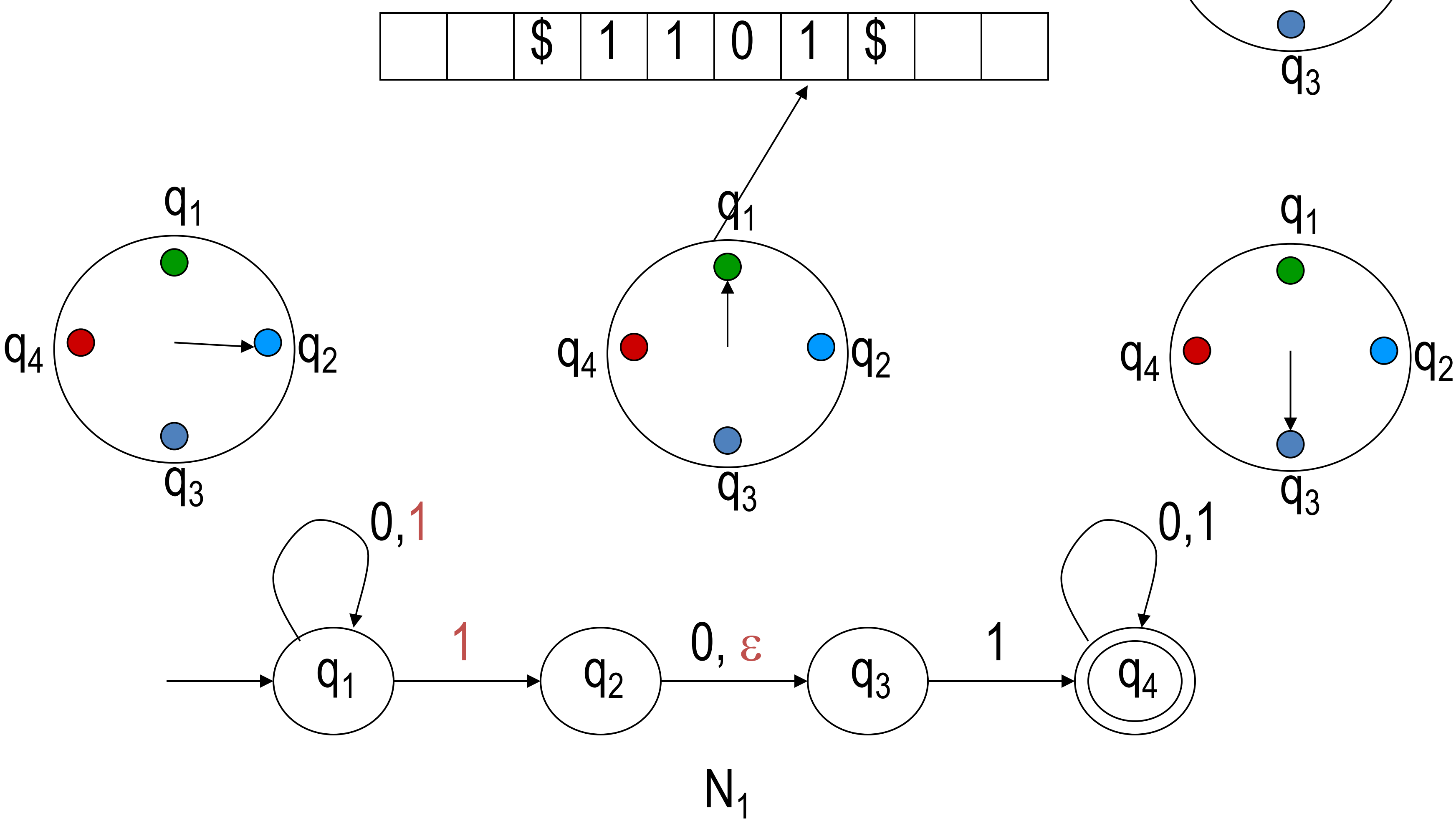
N_1 on input 1101 (3)



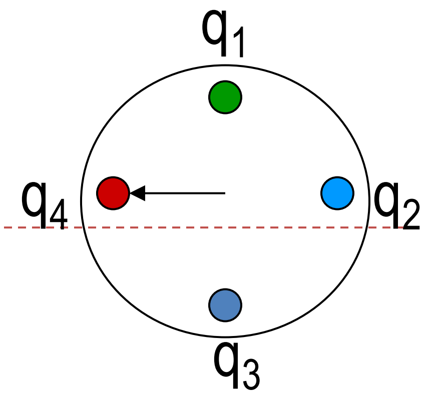


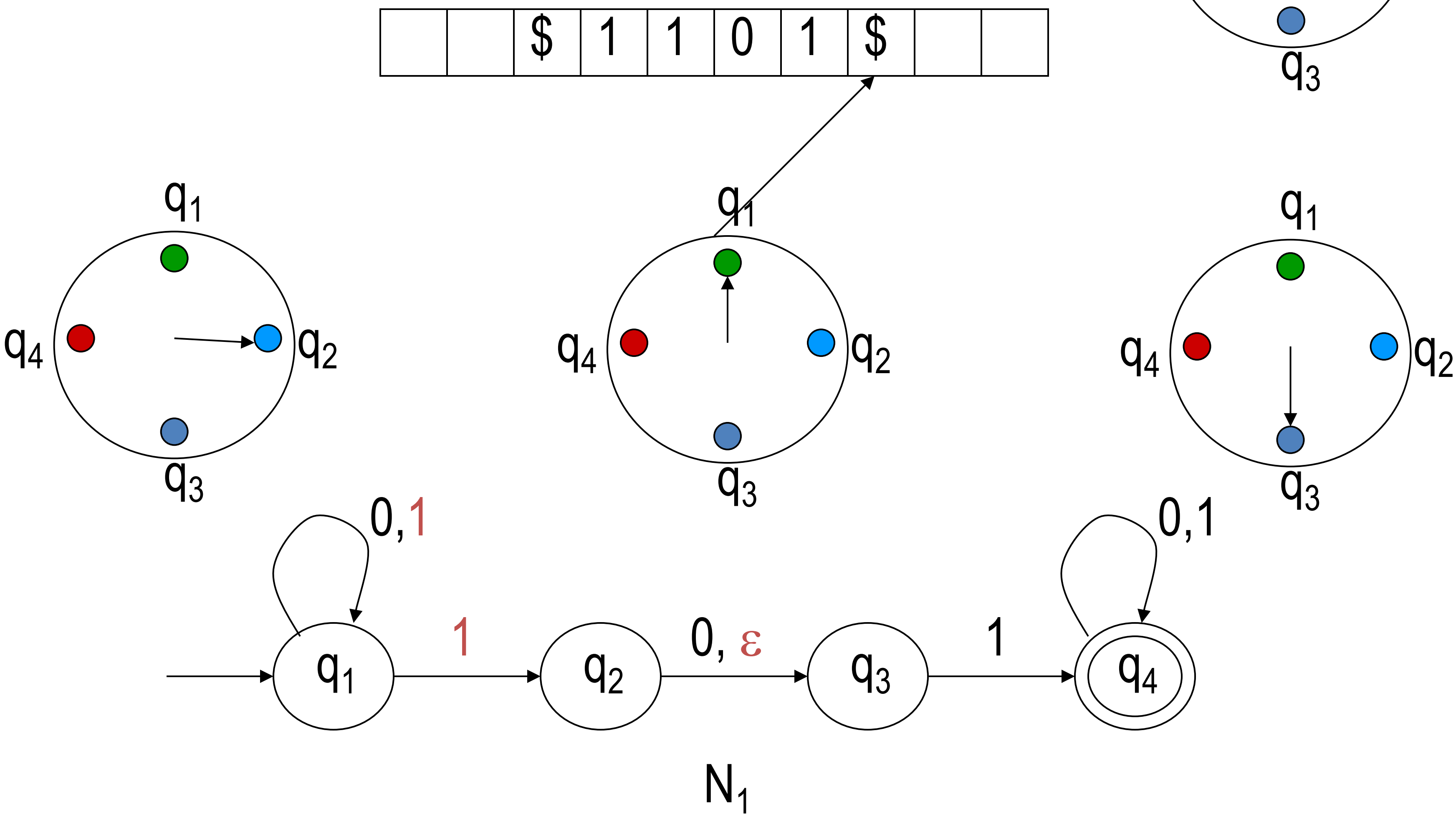
N_1 on input 1101 (4)



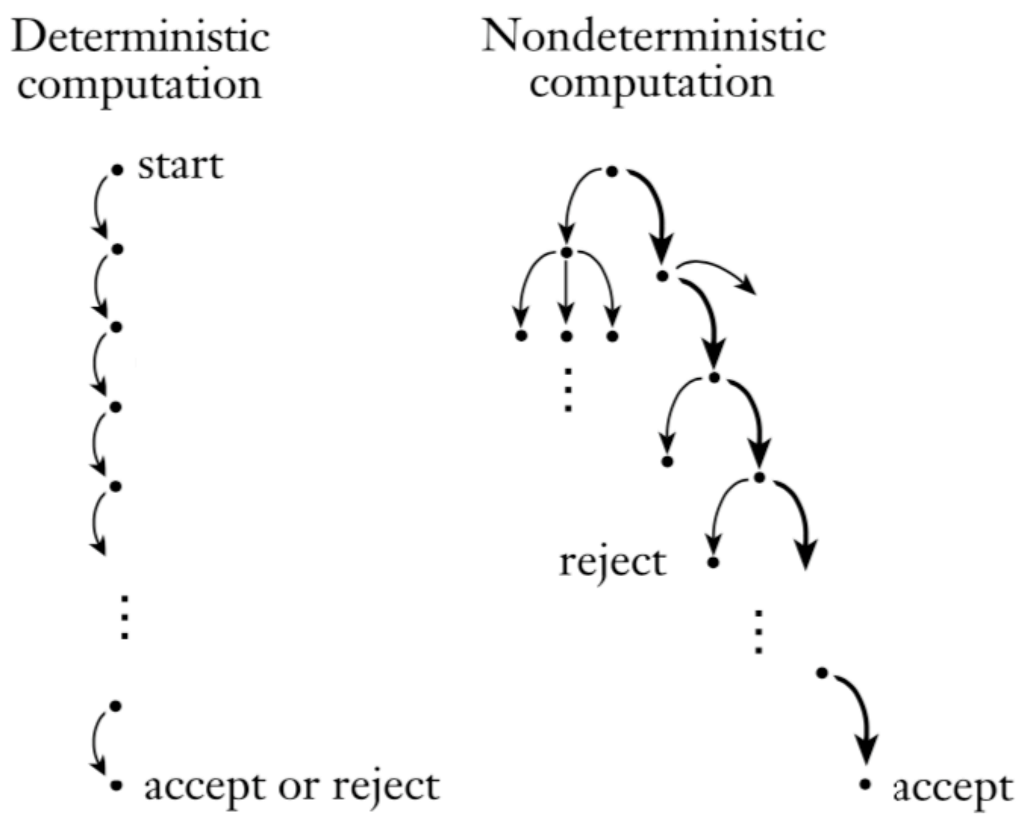


N_1 on input 1101 (5)

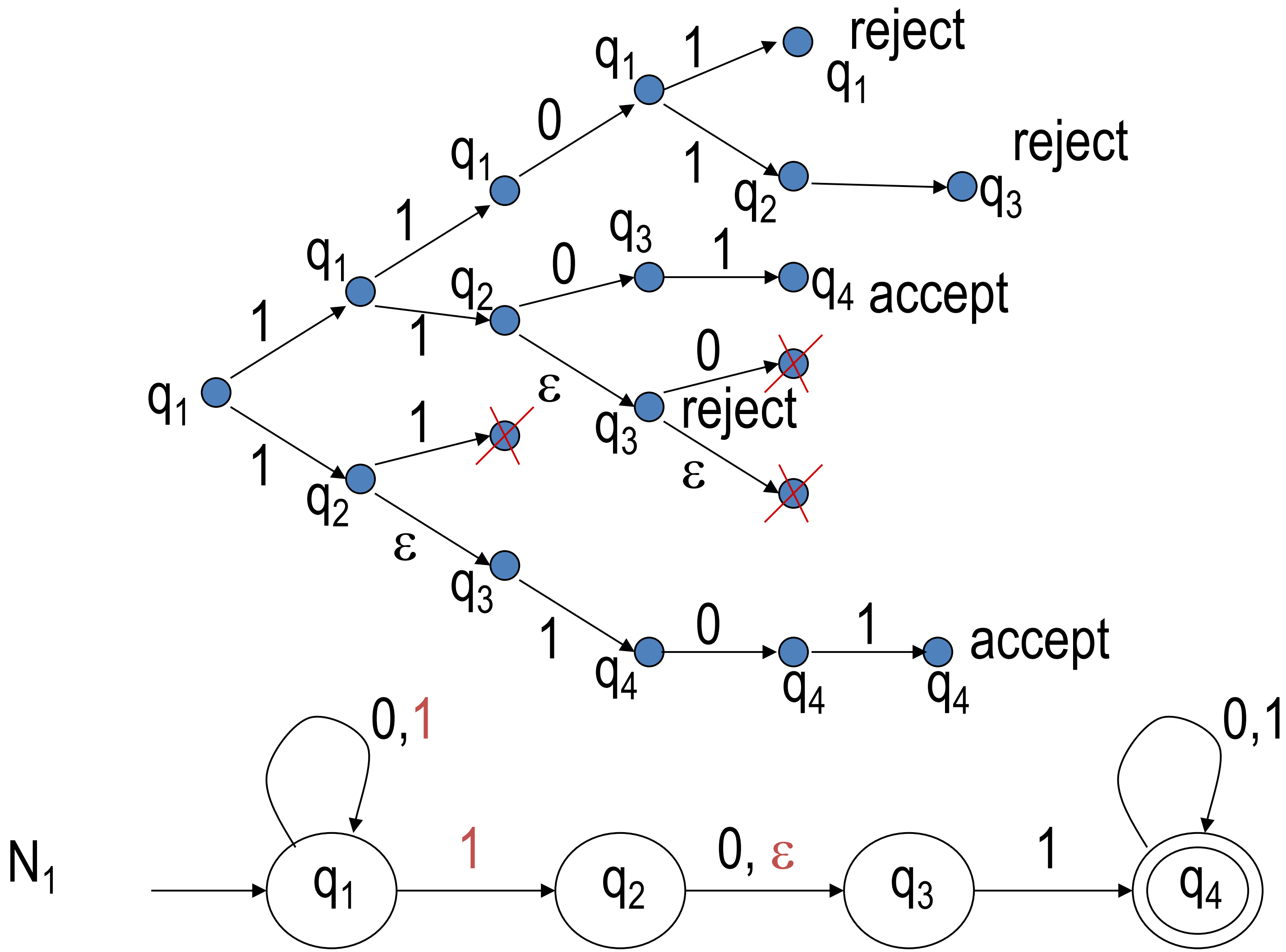




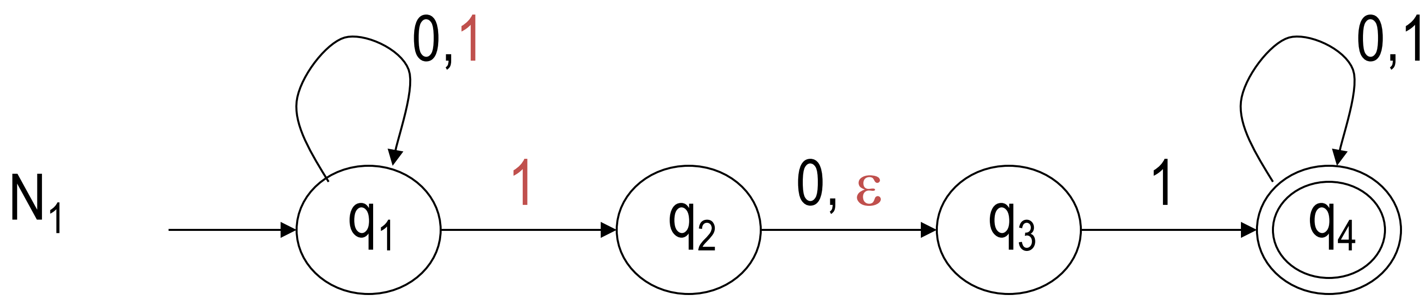
Deterministic finite automaton (DFA) vs Nondeterministic finite automaton (NFA)



Computation branch of N_1 on input 1101



NFA diagram - -> Description



- $L(N_1)$ = { w | w contains substring 101 or 11 }

NFA description - -> Diagram

- $L(N_2)$={w | The third from end letter of w is 1}

  Σ={0,1}

  e.g., 0101, 0010111

NFA description - -> Diagram

- $L(N_2)$={w | The third from end letter of w is 1}

  Σ={0,1}

- Non-determinism: to test the third from end letter

~~q4: last letter of w~~
~~q3: the first from end letter of w~~
~~q2: the second from end letter of w~~
~~q1: the third from end letter of w~~

q4: the third from end letter of w is 1
q3: the second from end letter of w is 1
q2: the first from end letter of w is 1
q1: all strings

NFA description - -> Diagram

- $L(N_2)$={w | The third from end letter of w is 1}

  Σ={0,1}

- Non-determinism: to test the third from end letter

q4: the third from end letter of w is 1
q3: the second from end letter of w is 1
q2: the first from end letter of w is 1
q1: all strings

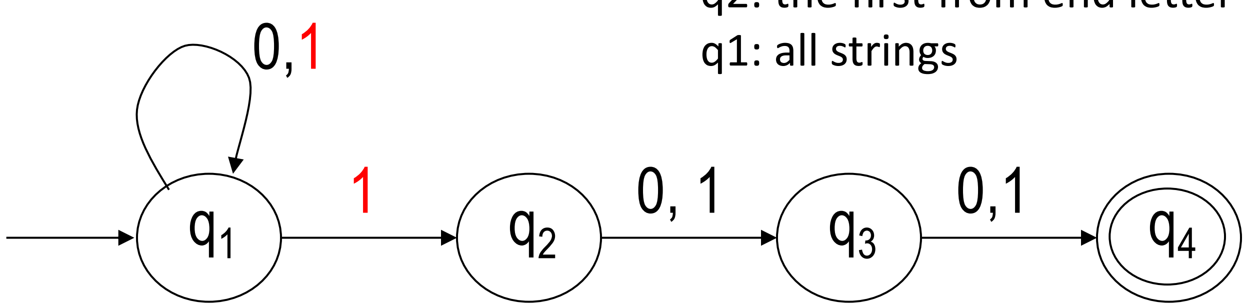
NFA description - -> Diagram

- $L(N_2)$={w | The third from end letter of w is 1}

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

- Non-determinism: to test the third from end letter

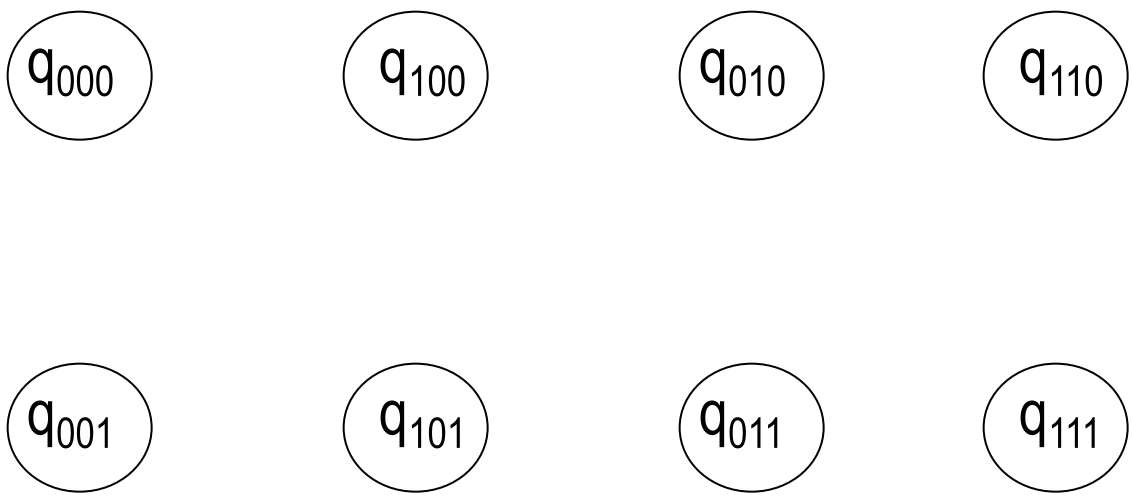
q4: the third from end letter of w is 1
q3: the second from end letter of w is 1
q2: the first from end letter of w is 1
q1: all strings



Language description - -> DFA diagram

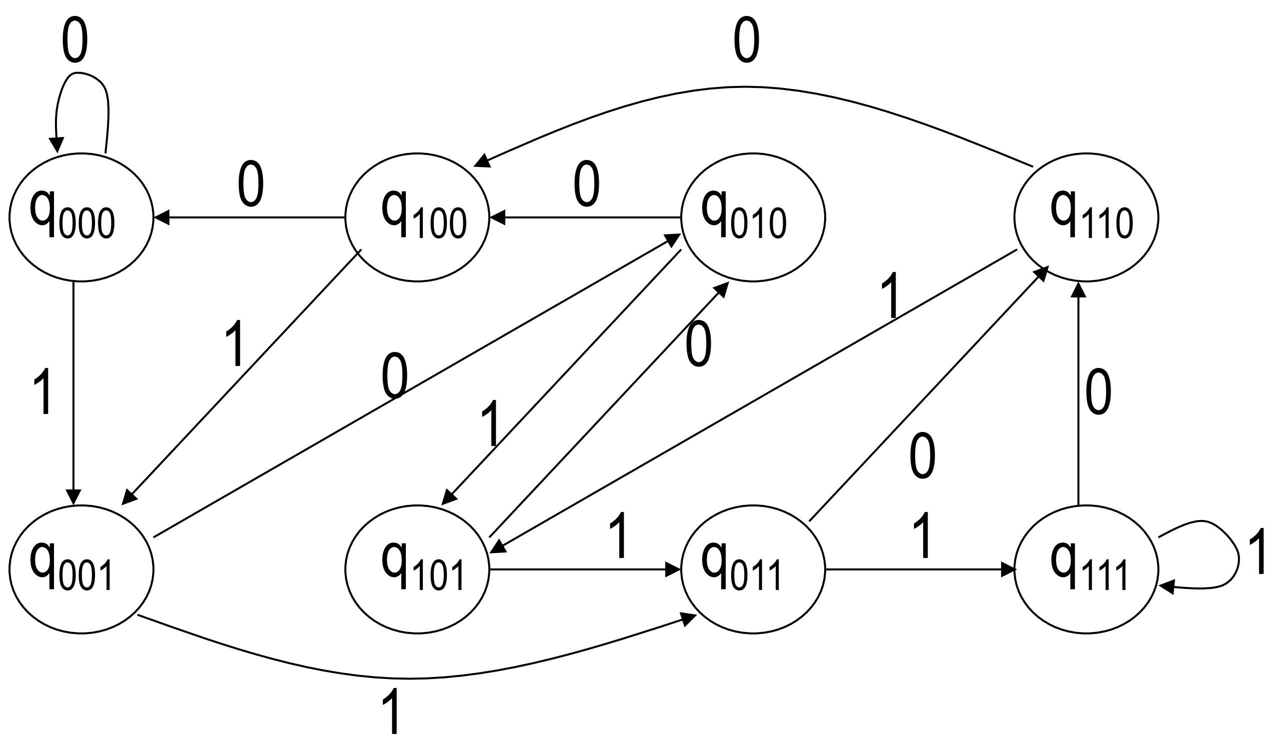
- $L(N_2)=\{w \mid \text{The third from end letter of } w \text{ is } 1\}$, $\Sigma=\{0,1\}$

- determinism: we need to record the last three letters



Language description - -> DFA diagram

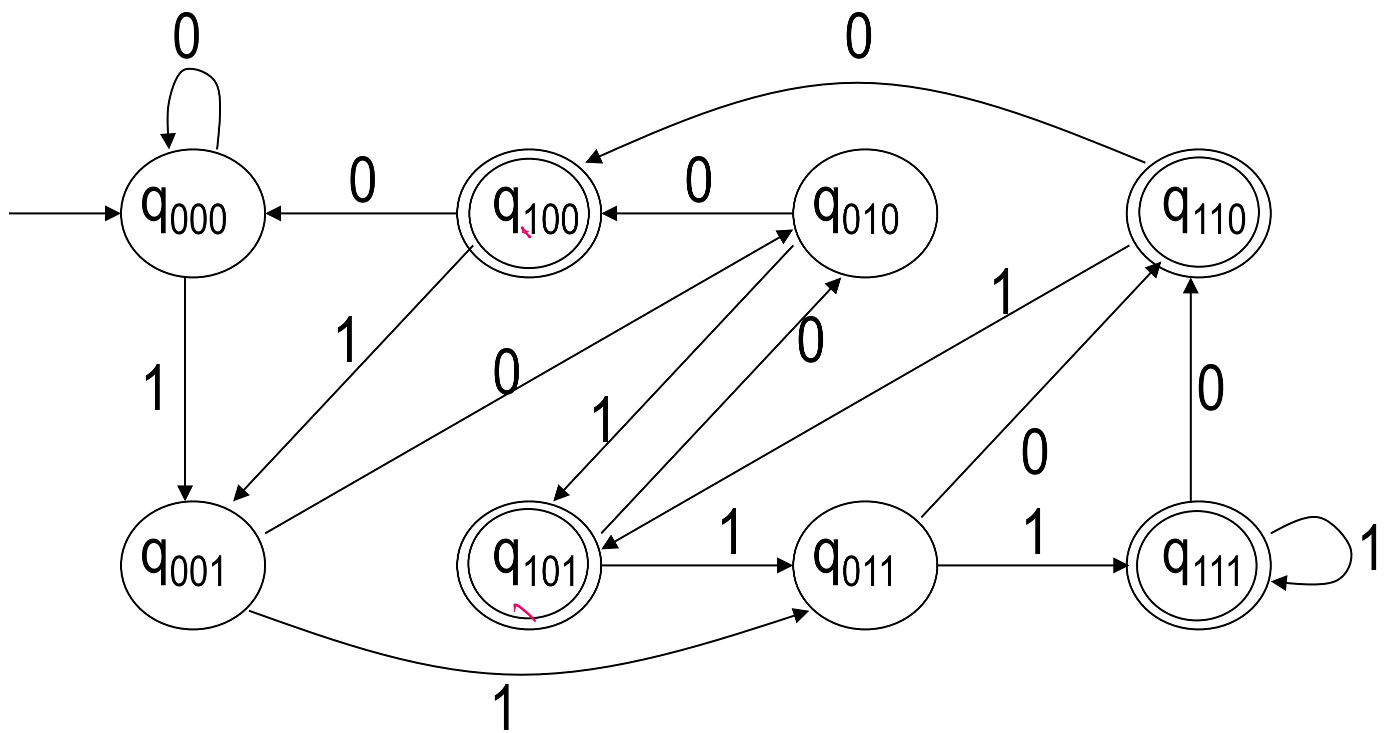
- $L(N_2)=\{w \mid \text{The third from end letter of w is 1}\}$, $\Sigma=\{0,1\}$
- determinism: we need to record the last three letters



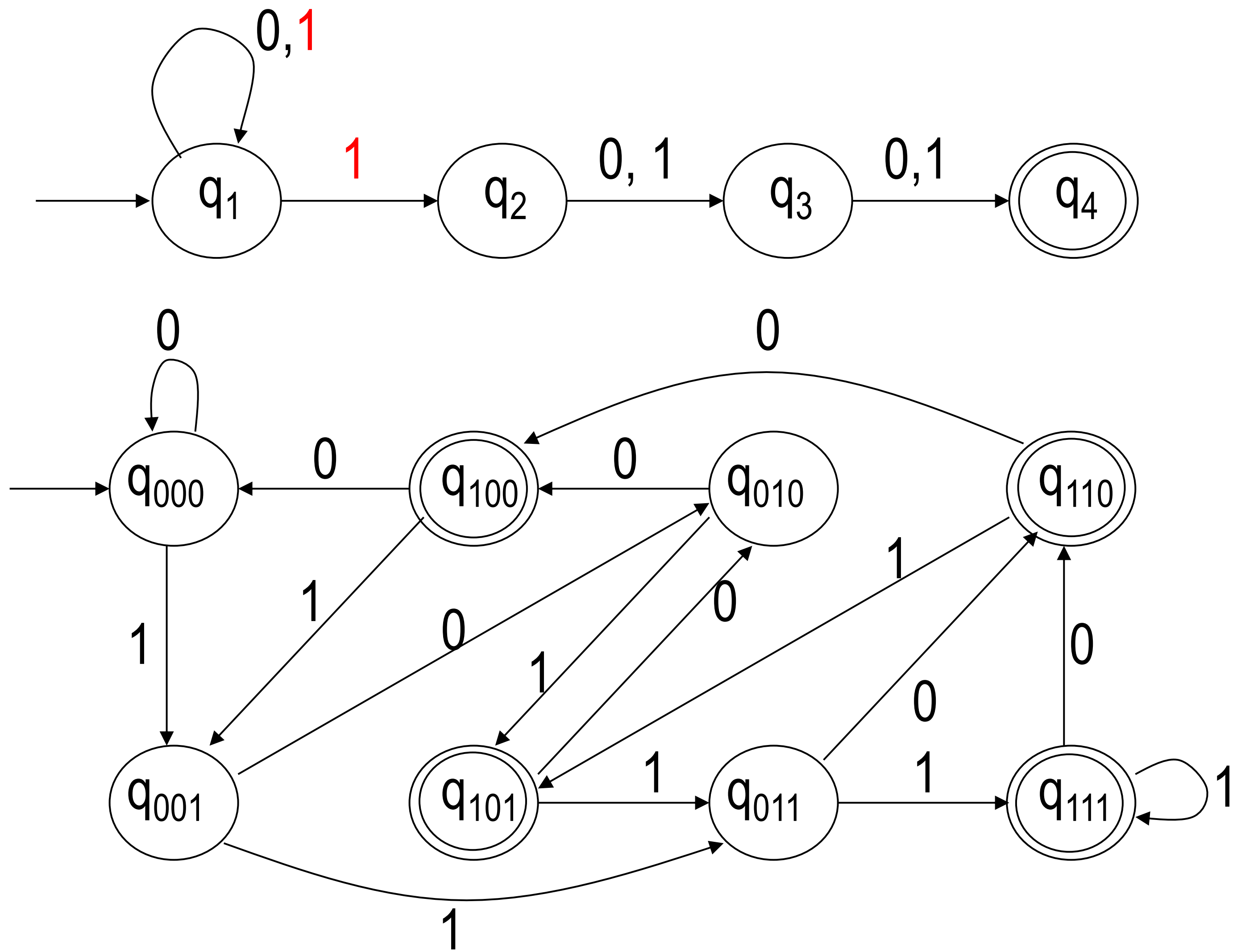
Language description - -> DFA diagram

- $L(N_2)$={w | The third from end letter of w is 1} , Σ={0,1}

- determinism: we need to record the last three letters



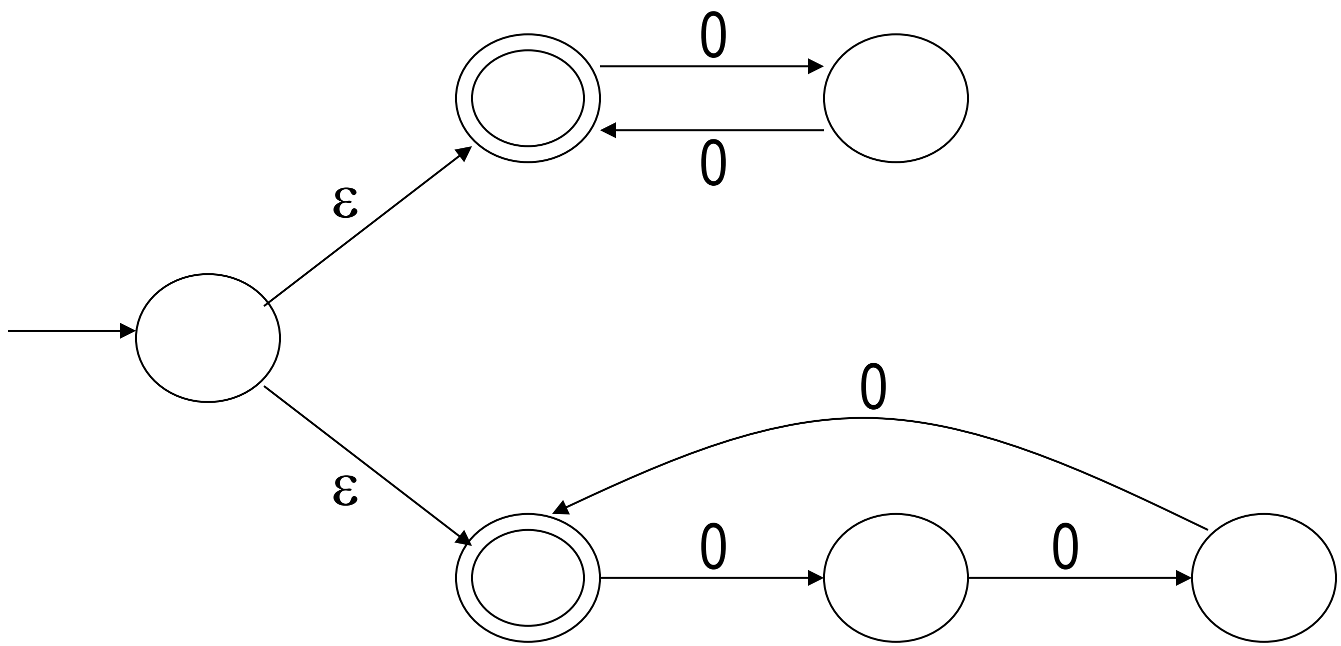
Comparison



0,1
1
0, 1
0,1
q1
q2
q3
q4
0
0
0
0
q000
q100
q010
q110
1
1
0
1
0
1
0
0
q001
q101
q011
q111
1
1
1
1

Example: NFA description - -> Diagram

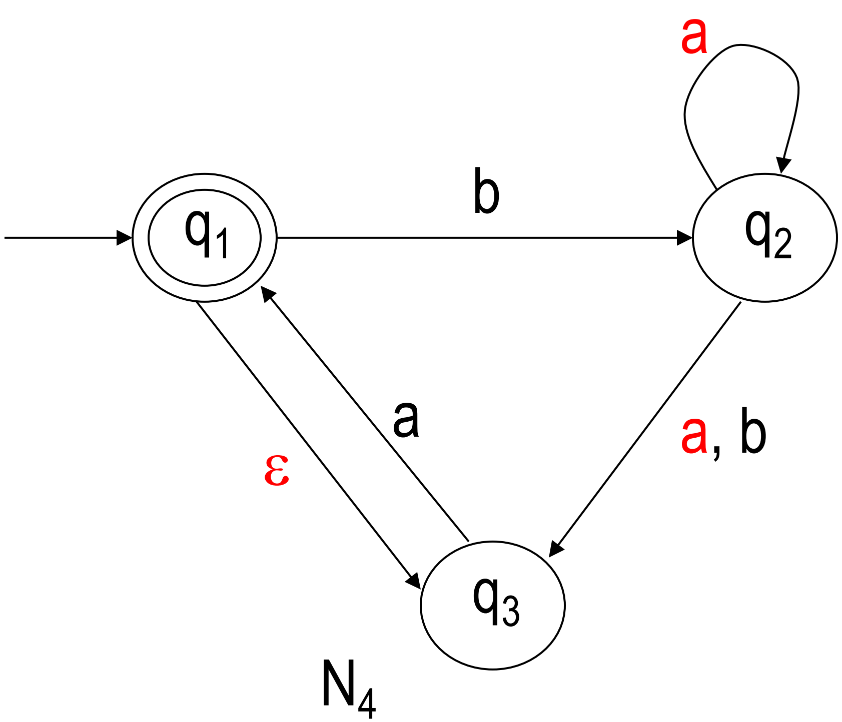
- $L(N_3) = \{ 0^k \mid$ where k is a multiple of 2 or 3 $\}$



N_3

NFA diagram: accept or reject?

- N_4: What strings does it accept/reject?



ε, a, b, bb, baa, baba, babba?

N_4 accepts ε, a, baba, baa, rejects b, bb, babba.

Definition of nondeterministic finite automaton

$N = (Q,\Sigma,\delta,q_0,F)$, where

- Q: finite set of states
- Σ: finite alphabet as input;   ($\Sigma_\varepsilon=\Sigma\cup\{\varepsilon\}$)
- δ: $Q\times\Sigma_\varepsilon\rightarrow P(Q)$, transition function
- $q_0\in Q$: start state
- $F\subseteq Q$: accept state set

DFA vs. NFA definition comparison

A ***finite automaton*** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

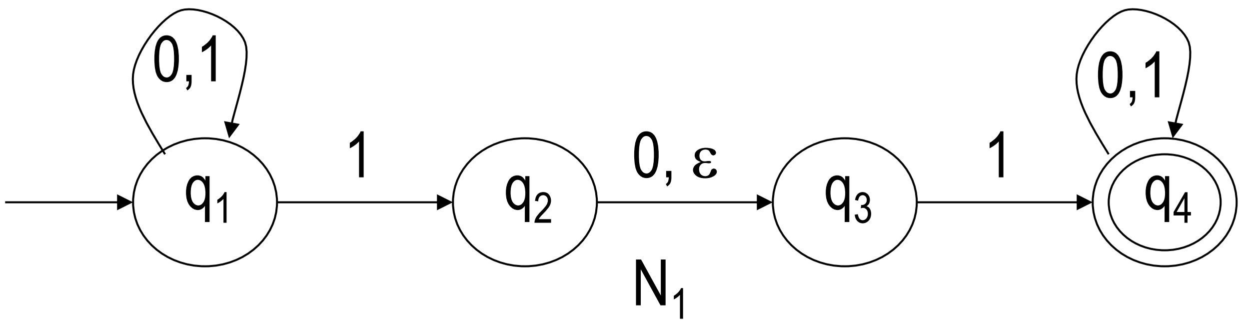
1. Q is a finite set called the ***states***,
2. Σ is a finite set called the ***alphabet***,
3. $\delta \colon Q \times \Sigma \longrightarrow Q$ is the ***transition function***,[1]
4. $q_0 \in Q$ is the ***start state***, and
5. $F \subseteq Q$ is the ***set of accept states***.[2]

A ***nondeterministic finite automaton*** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set of states,
2. Σ is a finite alphabet,
3. $\delta \colon Q \times \underline{\Sigma_\varepsilon} \longrightarrow \underline{\mathcal{P}(Q)}$ is the transition function,
4. $q_0 \in Q$ is the start state, and
5. $F \subseteq Q$ is the set of accept states.

Input: ε
Transition function: to some states, destination is a set of states

Example: NFA diagram - -> definition



$N_1=(Q,\Sigma,\delta,q_1,F)$;

$Q = \{q_1,q_2,q_3,q_4\}$;

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

$q_0 = q_1$

$F = \{q_4\}$;

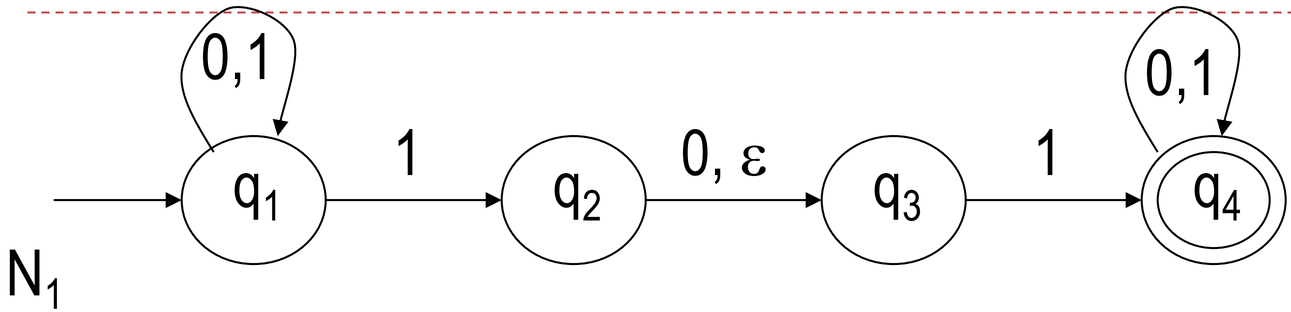
$\delta =$

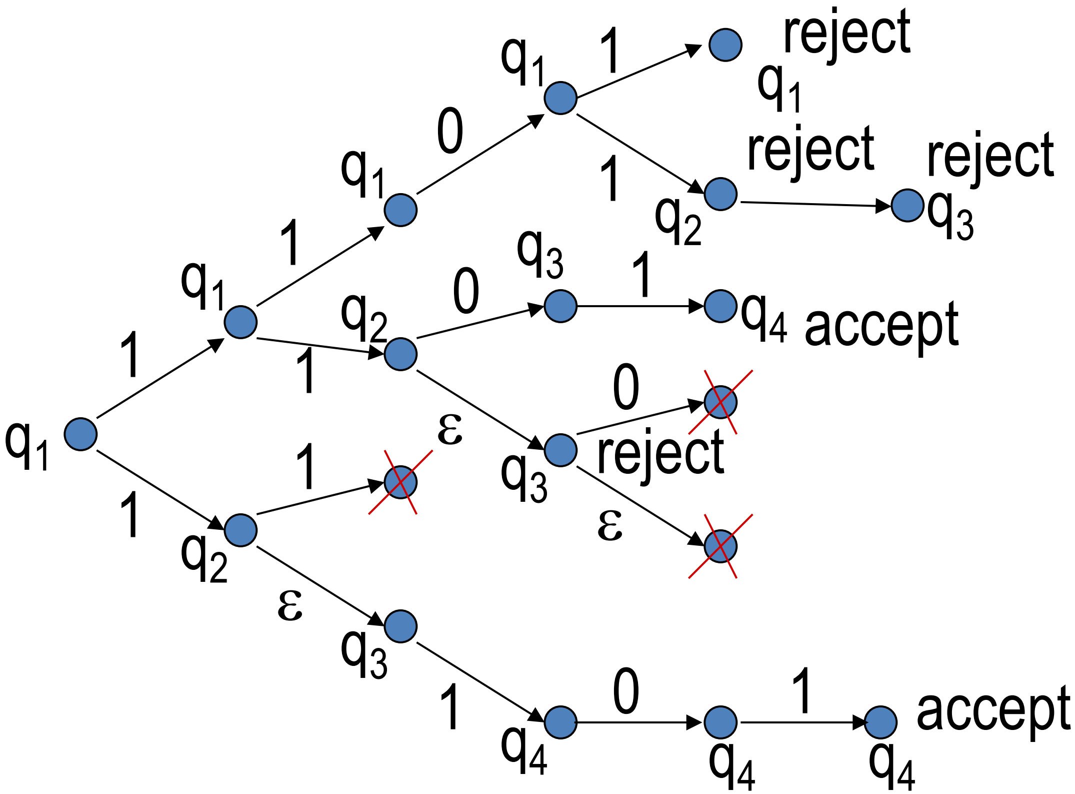
	0	1	ε
q_1	$\{q_1\}$	$\{q_1,q_2\}$	$\emptyset$
q_2	$\{q_3\}$	$\emptyset$	$\{q_3\}$
q_3	$\emptyset$	$\{q_4\}$	$\emptyset$
q_4	$\{q_4\}$	$\{q_4\}$	$\emptyset$

Definition of computation for NFAs

- NFA $N=(Q,\Sigma,\delta,q_0,F)$
  - Input $w=w_1w_2\ldots w_m$
- Computation: state sequence $r_0,r_1,\ldots,r_m$
  - $r_0=q_0$
  - $r_{i+1}\in\delta(r_i,w_{i+1})$ $(i=0,1,\ldots,m-1)$
- Accept:
  - $r_m\in F$
- M accepts w: there exists one accept
  - $L(M)=\{x \mid M \text{ accept } x\}$

Computation of N_1 on input 1101





computation1: q_1,q_1,q_1,q_1,q_1

computation2: q_1,q_1,q_1,q_1,q_2

computation3: q_1,q_1,q_1,q_1,q_2,q_3

computation4: q_1,q_1,q_2,q_3,q_4

computation5: q_1,q_1,q_2,q_3

computation6: q_1,q_2

computation7: q_1,q_2,q_3,q_4,q_4,q_4