# Data Structures & Algorithms

Finite State Machines-FSMs

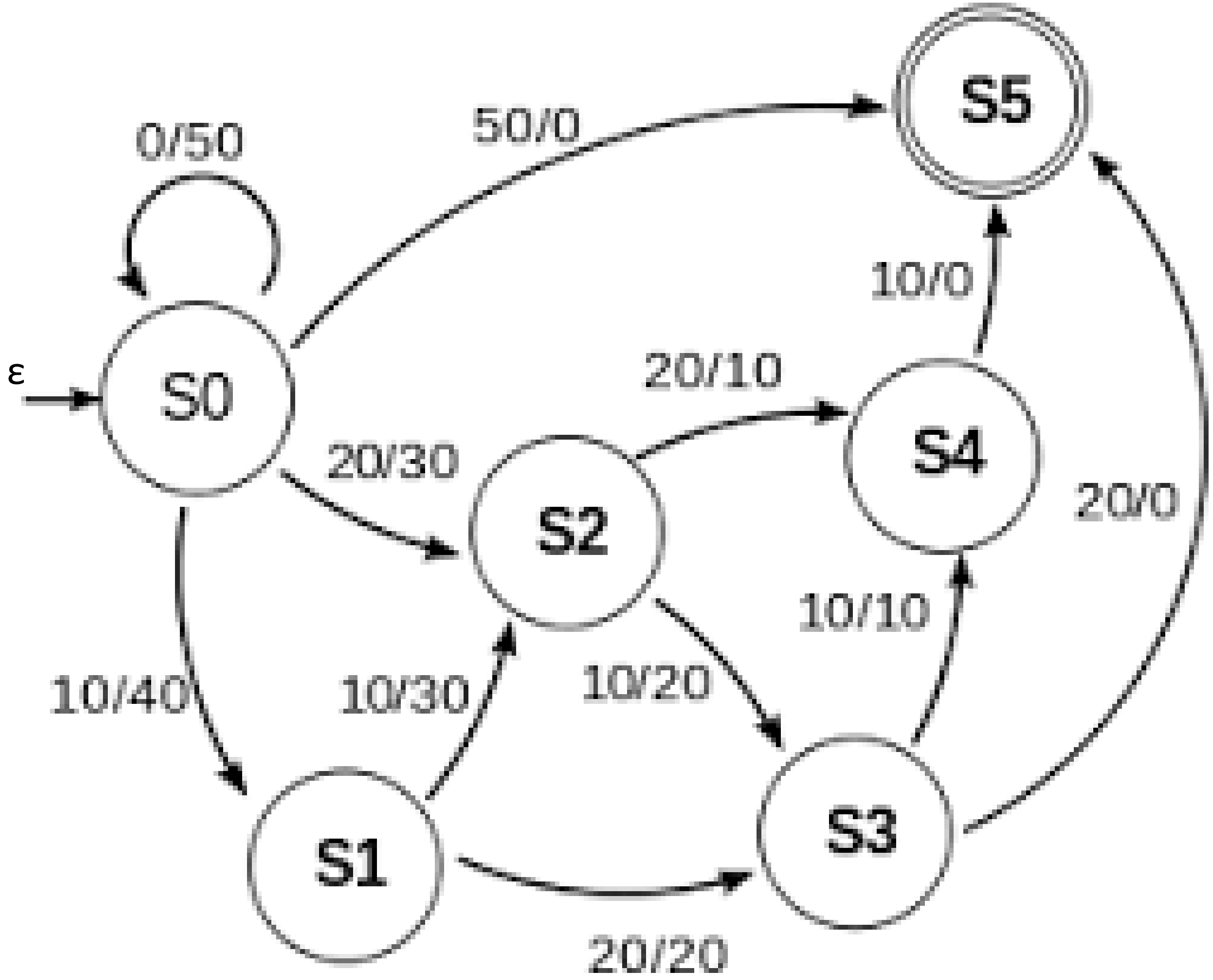
# Abstract model for coin operated telephone system

*Si* State

Transition
Input/output

**ε** void

Set of symbols
{**ε**, 10, 20, 50}



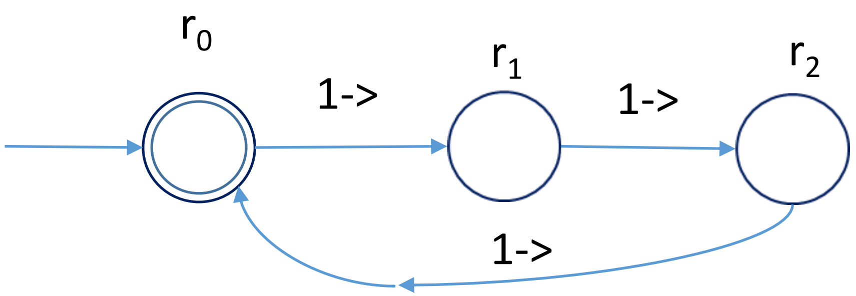
# Finite State Machines-FSMs

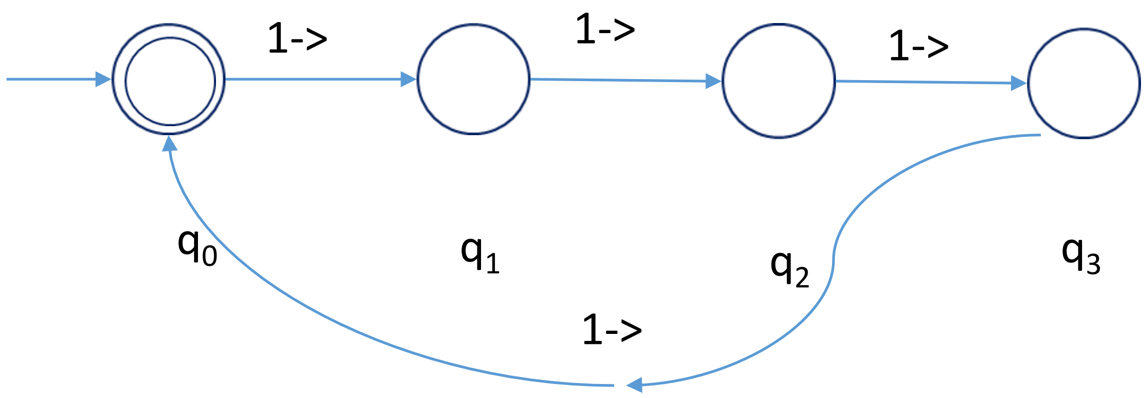
- FSM is a 4-tuple ***(Q, $q_0$, Next, Out)***
- ***Q*** is a finite state of possible states
- ***$q_0$*** is the initial state
- ***Next*** is a function mapping the <state, input symbols> to states
- ***Out*** is a function mapping the <state, input symbols> to output symbols

There are Finite State Recognizers (FSRs) that get a limited number of input sentences and we call the input set as *alphabet*

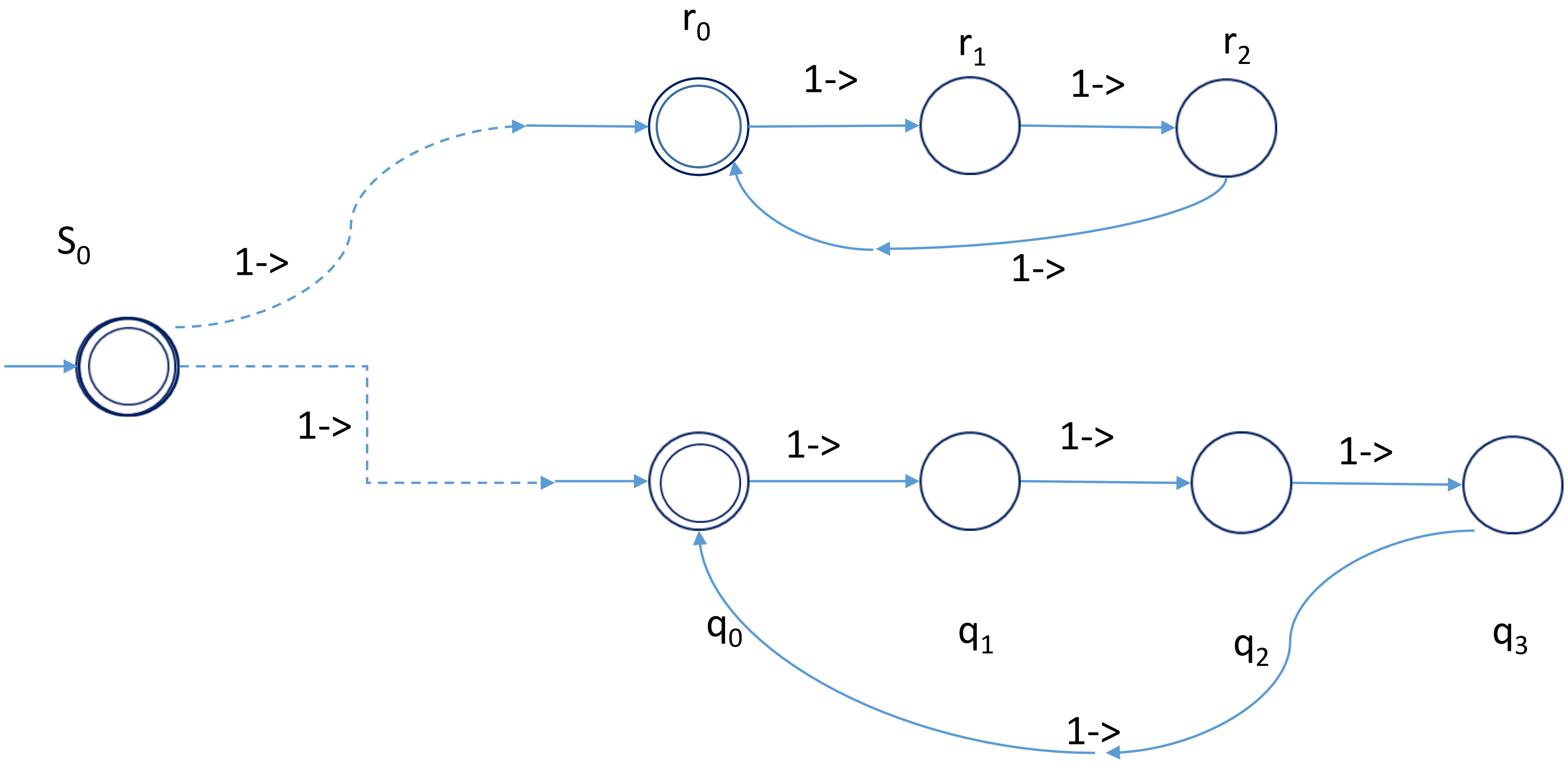
There are also the Finite State Generators (FSGs) that produce certain output strings (a string/sentence = a set of output symbols, produced one-symbol/stage)

# Non Deterministic FSM



# Non Deterministic FSM



# Non Deterministic FSM

- NDFSM is a 4-tuple ***(Q, $q_0$, Next, F)***
- ***Q*** is a finite state of states
- ***$q_0$*** is the initial state
- ***F*** is a set of final states ***F*** $\subseteq$ ***Q***
- ***Next*** is a function defined on certain pairs (q, a) of states and input symbols (a can be **ε**) and yields sets of possible next states (subsets of Q). If Next is defined for (q, **ε**) then it is undefined for (q, b), b another input symbol.

# $\forall$ NDFSM $\exists$ DFSM

Proof by construction: Let NDFSM ***(Q, $q_0$, Next, F)***. Construct ***(Q', $q_o$', Next', F')*** so that:

- $Q'=2^Q$
- ***$q_0'=\{q_0\}$***
- ***Next'***$(\{q_1, \ldots, q_r\}, a) = Next(q_1, a) \cup \ldots \cup Next(q_r, a)$ . The ***Next'***$(\{q_1, \ldots, q_r\}, a)$ provides a transition to the state that represents the set $\{q_1, \ldots, q_r\}$.
- ***F'***$=\{q' \subseteq Q' \mid q' \cap F \neq \{\}\}$. Final state is any state that contains a final state of the original NDFSM