Non–Deterministic Finite Automata (NFA)

Based on slides of Aaron Deever

Deterministic Finite Automata

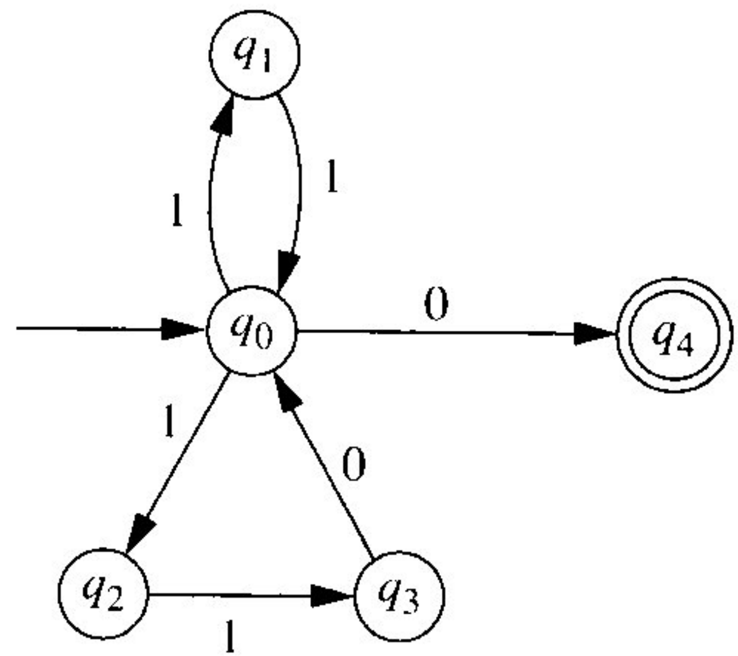
- Automata we've been dealing with have been deterministic
  - For every state and every alphabet symbol there is exactly one move that the machine can make.
  - $\delta : Q \times \Sigma \rightarrow Q$

Non-Deterministic Finite Automata

- Non-determinism
  - When machine is in a given state and reads a symbol:
    - The machine may have a choice of where to move to next
    - There may be states where, after reading a given symbol, the machine has nowhere to go
    - Applying the transition function will give, not necessarily 1 state, but 0 or more states
    - A DFA is just a special case of an NFA

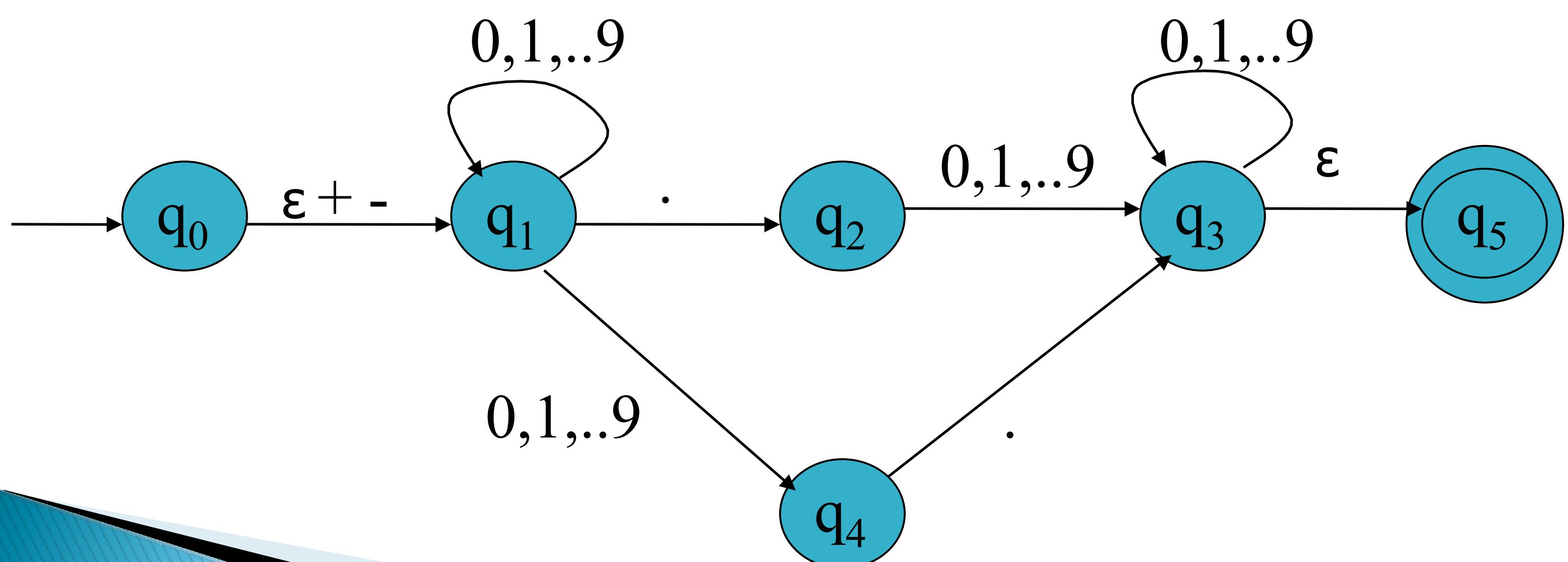
Non–Deterministic Finite Automata

- Example
  - q_0 has two choices where to go if a 1 symbol is received
  - q_4 has no exiting arrows
  - q_1, q_2 and q_3 have only some exiting arrows defined



Non-Deterministic Finite Automata

- Furthermore, transitions can be defined where the machine doesn't need to read in a symbol.
  - Such transitions are called ϵ-transitions



How Does an NFA Compute?

- Running an NFA on an input string
  - We follow all possible paths out of a state in parallel
    - E.g. if there are four arrows out of state q when a symbol a is received, we branch to follow all 4 paths in parallel from that point forward.
    - If there are no paths out of a state for a given symbol, that branch is abandoned
    - If there is an ϵ symbol exiting a state, we immediately branch: one branch stays in the current state and one immediately proceeds along the ϵ arrow *without* reading an additional symbol
  - A string is accepted by the NFA if *any* of the branches is in an accept state at the end of the input

Why NFAs?

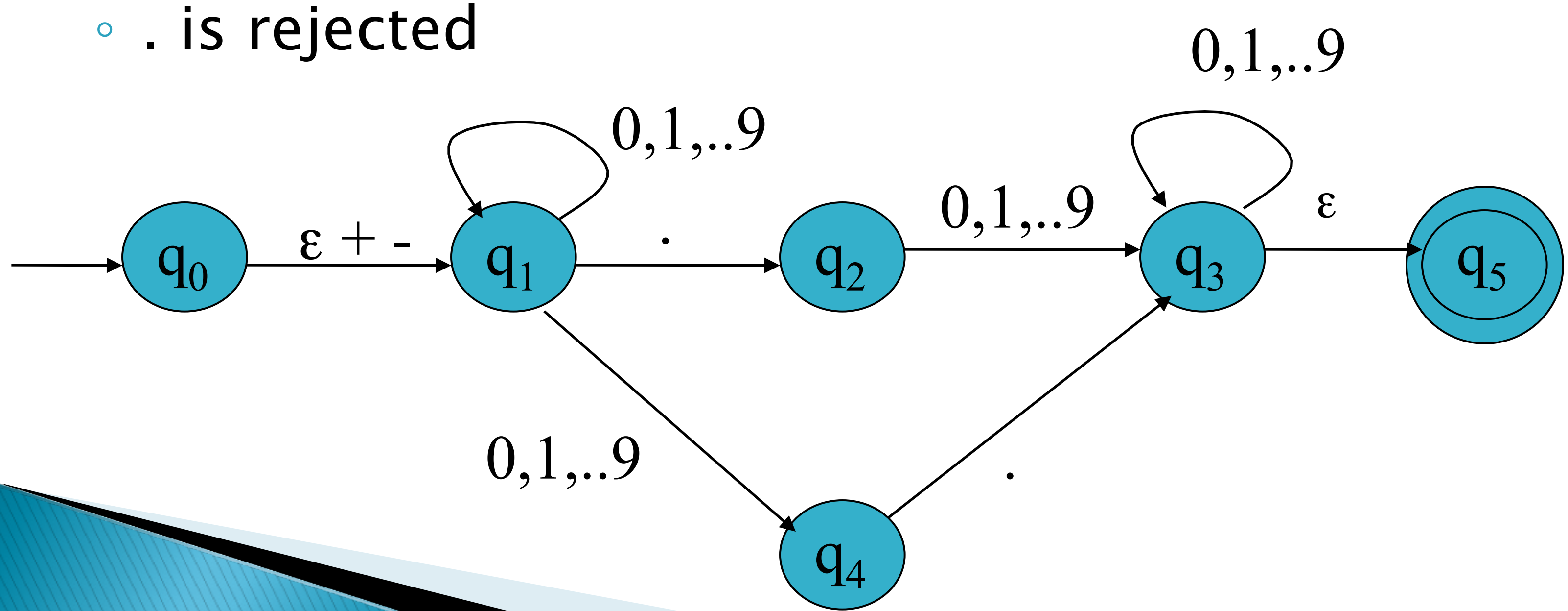
- We'll show later that actually any NFA can be converted into an equivalent DFA
  - So it's not actually representing any new languages
- But…
  - Sometimes NFAs are easier to construct than corresponding DFAs
    - Not constrained to one transition per state per symbol
  - An NFA may be smaller than its corresponding DFA
  - An NFA's functioning may be easier to understand
  - Helpful in proofs

String Acceptance in an NFA

- How does such a machine accept?
  - A string will be accepted if there is <u>at least one</u> sequence of state transitions on an input that leaves the machine in an accepting state.

Non-Deterministic Finite Automata

- Example:
  - −3.45 is accepted
  - .5678 is accepted
  - +99. is accepted
  - 37 is rejected
  - . is rejected



Thanks for your attention!