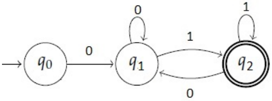
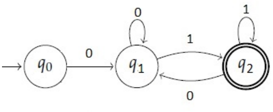
CSC236H

Introduction to the Theory of Computation

Let $L = \{0s1 : s \in \{0,1\}^*\}$.
Give a DFA which **only accepts** strings in L.



Let $L = \{0s1 : s \in \{0,1\}^*\}$.
Give a DFA which **only accepts** strings in L.



- How can we formally prove that the DFA only accepts strings in L?

A **Deterministic Finite Automaton (DFA)** $\mathcal{D}$ is a quintuple $\mathcal{D} = \langle Q, \Sigma, \delta, q_0, F \rangle$ where:

- Q is the **set of states** in $\mathcal{D}$;
- Σ is the **alphabet** of symbols used by $\mathcal{D}$;
- $\delta : Q \times \Sigma \to Q$ is the **transition function**;
- $q_0 \in Q$ is the **initial state** of $\mathcal{D}$;
- $F \subseteq Q$ is the set of **accepting states** of $\mathcal{D}$.

DFA's – Extended Transition Function

Let Σ^* be the smallest set such that:

- $\epsilon \in \Sigma^*$.
- If $x \in \Sigma^*$ and $a \in \Sigma$ then $xa \in \Sigma^*$.

Let $\delta : Q \times \Sigma \to Q$ be the transition function of a DFA $\mathcal{D}$. The **extended transition function** of the DFA is the function $\delta^* : Q \times \Sigma^* \to Q$ defined by structural induction on x:

- $\delta^*(q, \epsilon) = q$.
- For some $x \in \Sigma^*$ and $a \in \Sigma$,

$$\delta^*(q, xa) = \delta(\delta^*(q, x), a).$$

If $\delta^*(q, w) = q'$ we say that w takes the automaton $\mathcal{D}$ from q to q'.

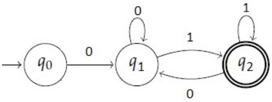
- A string $w \in \Sigma^*$ is **accepted** by $\mathcal{D}$, if and only if w takes the automaton from the initial state to an accepting state.

$$\delta^*(q_0, w) \in F.$$

- The language **accepted** (or **recognised**) by a DFA $\mathcal{D}$, denoted $\mathcal{L}(\mathcal{D})$, is the set of all strings accepted by $\mathcal{D}$.

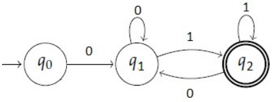
$$\mathcal{L}(\mathcal{D}) = \{w \in \Sigma^* : \delta^*(q_0, w) \in F\}$$

Let $L = \{0s1 : s \in \{0,1\}^*\}$.
Give a DFA which **only accepts** strings in L.



$$\delta^*(q_0, w) = q_2 \text{ iff } w \in L.$$

Let $L = \{0s1 : s \in \{0,1\}^*\}$.
Give a DFA which **only accepts** strings in L.

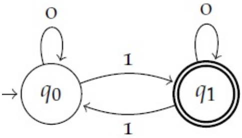


$$\delta^*(q_0, w) = q_2 \qquad \text{if } w \text{ starts by 0 and end by 1.}$$
$$\delta^*(q_0, w) = q_1 \qquad \text{if } w \text{ starts by 0 and end by 0.}$$
$$\delta^*(q_0, w) = q_0 \qquad \text{if } w \text{ is empty.}$$

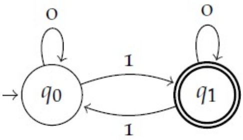
DFA's – State Invariants

- **Invariant for a state q:** a predicate P over domain Σ^* such that for every string $w \in \Sigma^*$, $\delta^*(q_0, w) = q$ if and only if $P(w)$ is true.
- The state invariants for a DFA should be **mutually exclusive**.
  No string should satisfy two different state invariants.
- The state invariants for a DFA should be **exhaustive**.
  Every string in Σ^*, including ϵ, should satisfy one of the state invariants.

Let $L = \{w \in \{0,1\}^* : w$ has an odd number of 1's$\}$.



Let $L = \{w \in \{0,1\}^* : w$ has an odd number of 1's$\}$.



$$P(w) : \begin{cases} \delta^*(q_0, w) = q_1 & \text{if } w \text{ has an odd number of 1's.} \\ \delta^*(q_0, w) = q_0 & \text{if } w \text{ has an even number of 1's.} \end{cases}$$

The set of **regular languages** over an alphabet Σ is defined recursively as follows:

- $\{\}$ is a regular language.
- $\{\epsilon\}$ is a regular language.
- For any symbol $a \in \Sigma$, $\{a\}$ is a regular language.
- If L, M are regular languages, then so are $L \cup M, L \circ M$, and $L^{\circledast}$.

Let $\mathcal{L}(R)$ denote the language that a regular expression R represents:

- $\varnothing$ is a regex, with $\mathcal{L}(\varnothing) = \{\}$ (matches no string)
- ϵ is a regex, with $\mathcal{L}(\epsilon) = \{\epsilon\}$
- For all symbols $a \in \Sigma$, a is a regex with $\mathcal{L}(a) = \{a\}$
- Let S, T be regexes. Then $S + T$, ST, and S^* are regexes, with
  - $\mathcal{L}(S + T) = \mathcal{L}(S) \cup \mathcal{L}(T)$
  - $\mathcal{L}(ST) = \mathcal{L}(S) \circ \mathcal{L}(T)$
  - $\mathcal{L}(S^*) = (\mathcal{L}(S))^{\circledast}$

Regular Expressions (regex) – Example

Let $\Sigma = \{0, 1\}$. Consider the regex $01 + 1(0+1)^*$.

Regular Expressions (regex) – Example

Give a regular expression for the following regular language:

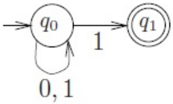
$$L = \{w \in \{0, 1\}^* : w \text{ has length at most } 2\}.$$

Regular Expressions (regex) – Example

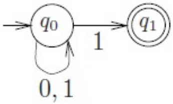
Give a regular expression for the following regular language:

$$L = \{w \in \{0,1\}^* : w \text{ has 11 as a substring}\}.$$

Nondeterministic Finite State Automaton (NFA's)



Nondeterministic Finite State Automaton (NFA's)



A **Nondeterministic Finite Automaton (NFA)** $\mathcal{N}$ is a quintuple $\mathcal{N} = \langle Q, \Sigma, \delta, q_0, F \rangle$ where:

- Q is the **set of states** in $\mathcal{N}$;
- Σ is the **alphabet** of symbols used by $\mathcal{N}$;
- $\delta : Q \times (\Sigma \cup \{\epsilon\}) \to \mathcal{P}(Q)$ is the **transition function**;
- $q_0 \in Q$ is the **initial state** of $\mathcal{N}$;
- $F \subseteq Q$ is the set of **accepting states** of $\mathcal{N}$.

- A string $w \in \Sigma^*$ is **accepted** by $\mathcal{N}$, if and only if at least one of the possible states in which the automaton could be after processing input w is an accepting state.

- The language **accepted** (or **recognised**) by a NFA $\mathcal{N}$, denoted $\mathcal{L}(\mathcal{N})$, is the set of all strings accepted by $\mathcal{N}$.