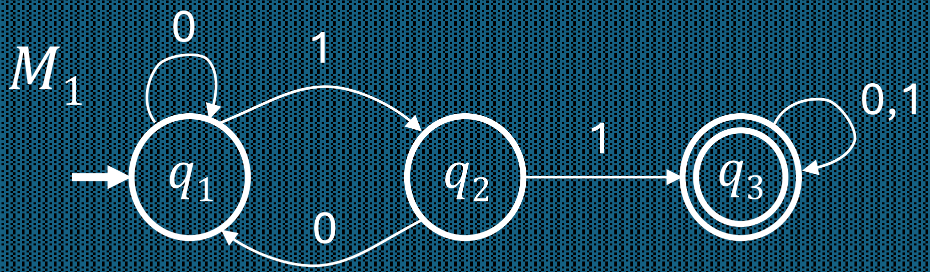
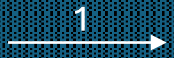
Intro to the Theory of Computation

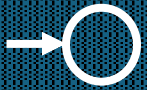
Instructor: Krishna Gaire

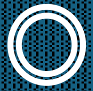
Let's begin: Finite Automata



States: $q_1\ q_2\ q_3$

Transitions: 

Start state: 

Accept states: 

Input: finite string
Output: Accept or Reject

Computation process: Begin at start state, read input symbols, follow corresponding transitions, Accept if end with accept state, Reject if not.

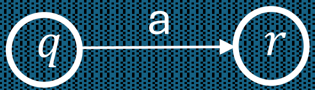
Examples: 01101 → Accept
00101 → Reject

M_1 accepts exactly those strings in A where
$A = \{w |\ w$ contains substring $11\}$.

Say that A is the language of M_1 and that M_1 recognizes A and that $A = L(M_1)$.

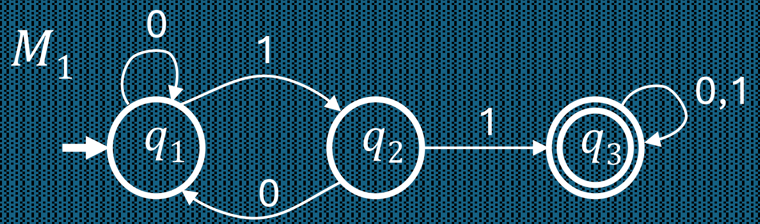
Finite Automata – Formal Definition

Defn: A <u>finite automaton</u> M is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$

- Q finite set of states
- Σ finite set of alphabet symbols
- δ transition function $\delta: Q \times \Sigma \rightarrow Q$ / $\delta(q, a) = r$ means 

- q_0 start state
- F set of accept states

Example:



$M_1 = (Q, \Sigma, \delta, q_1, F)$

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

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

$F = \{q_3\}$

$\delta =$

	0	1
q_1	q_1	q_2
q_2	q_1	q_3
q_3	q_3	q_3

Q. How many tuples does Finite Automata has?
(a) 4
(b) 5
(c) 6

Finite Automata – Computation

Strings and languages

- A string is a finite sequence of symbols in Σ
- A language is a set of strings (finite or infinite)
- The empty string ε is the string of length 0
- The empty language ø is the set with no strings

Defn: M accepts string $w = w_1 w_2 \dots w_n$ each $w_i \epsilon \Sigma$
if there is a sequence of states $r_0, r_1, r_2,, \dots, r_n \epsilon Q$
where:

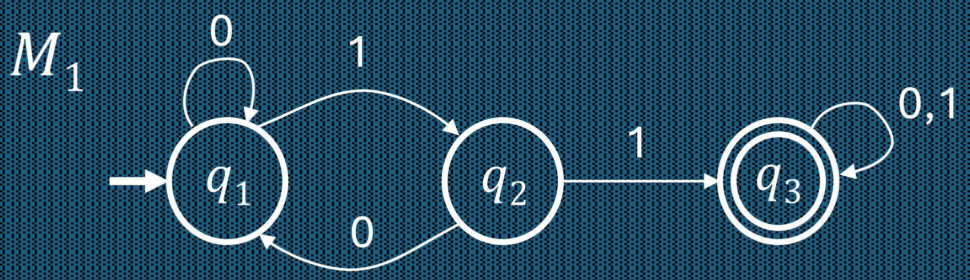
- $r_0 = q_0$
- $r_i = \delta(r_{i-1}, w_i)$ for $1 \leq i \leq n$
- $r_n \epsilon F$

Recognizing languages

- $L(M) = \{w | M \text{ accepts } w\}$
- $L(M)$ is the language of M
- M recognizes $L(M)$

Defn: A language is regular if some
finite automaton recognizes it.

Regular Languages – Examples



$L(M_1) = \{w \mid w \text{ contains substring } 11\} = A$

Therefore A is regular

More examples:

Let $B = \{w \mid w$ has an even number of 1s$\}$
B is regular (make automaton for practice).

Let $C = \{w \mid w$ has equal numbers of 0s and 1s$\}$
C is not regular (we will prove).

Goal: Understand the regular languages

Closure Properties on regular languages

Dfn: are defined as certain operations on regular language that are guaranteed to produce regular language.

Q. Closure properties of regular language ?

(a) Union

(b) Intersection

(c) Complementation

Regular Expressions

Regular operations. Let A, B be languages:

- **Union:** $A \cup B = \{w \mid w \in A \text{ or } w \in B\}$
- **Concatenation:** $A \circ B = \{xy \mid x \in A \text{ and } y \in B\} = AB$
- **Star:** $A^* = \{x_1 \ldots x_k \mid \text{each } x_i \in A \text{ for } k \geq 0\}$

  Note: $\varepsilon \in A^*$ always

Example. Let $A = \{\text{good, bad}\}$ and $B = \{\text{boy, girl}\}$.

- $A \cup B = \{\text{good, bad, boy, girl}\}$
- $A \circ B = AB = \{\text{goodboy, goodgirl, badboy, badgirl}\}$
- $A^* = \{\varepsilon$, good, bad, goodgood, goodbad, badgood, badbad, goodgoodgood, goodgoodbad, … $\}$

Regular expressions

- Built from Σ, members $\Sigma, \emptyset, \varepsilon$ [Atomic]
- By using $\cup, \circ, *$ [Composite]

Examples:

- $(0 \cup 1)^* = \Sigma^*$ gives all strings over Σ
- $\Sigma^* 1$ gives all strings that end with 1
- $\Sigma^* 11 \Sigma^*$ = all strings that contain 11 = $L(M_1)$

Goal: Show finite automata equivalent to regular expressions

Example 3 Let $\Sigma = \{a, b\}$ for each of the following languages over Σ, find a regular expression representing the following:

(a) All strings that contain exactly one 'a'.

(b) All strings begining with 'ab'.

(c) All strings that contains either the substring 'aaa' or 'bbb'.

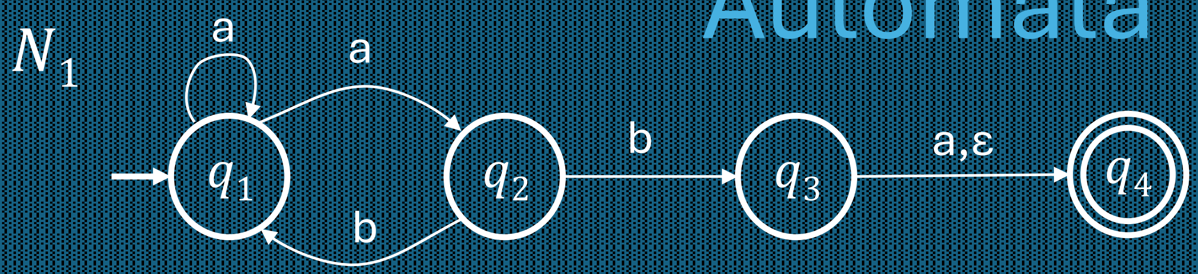
Solution

(a) $b^* a b^*$.

(b) $ab(a+b)^*$

(c) $(a+b)^*(aaa+bbb)(a+b)^*$.

Nondeterministic Finite Automata



Check-in 2.1

What does N_1 do on input aab ?

(a) Accept

(b) Reject

(c) Both Accept and Reject

New features of nondeterminism:

- multiple paths possible (0, 1 or many at each step)
- ε-transition is a "free" move without reading input
- Accept input if <u>some</u> path leads to accept

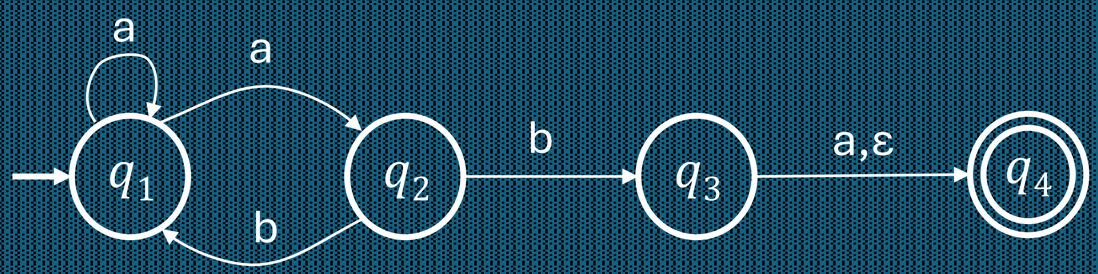
Exa[redacted] inputs:

- ab [redacted]
- aa [redacted]
- aba <u>accept</u>
- abb <u>reject</u>

Nondeterminism doesn't correspond to a physical machine we can build. However, it is useful mathematically.

NFA – Formal Definition

N_1



Defn: A <u>nondeterministic finite automaton (NFA)</u> N is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$

states
alphabet
transition function
start state
accept states

- all same as before except δ
- $\delta : Q \times \Sigma_\varepsilon \rightarrow \boldsymbol{\mathcal{P}}(Q) = \{R | R \subseteq Q\}$

  Σ_ε: $\Sigma \cup \{\varepsilon\}$ $\boldsymbol{\mathcal{P}}(Q)$: power set

- In the N_1 example: $\delta(q_1, \mathrm{a}) = \{q_1, q_2\}$

  $\delta(q_1, \mathrm{b}) = \emptyset$

Method for Proving language is regular

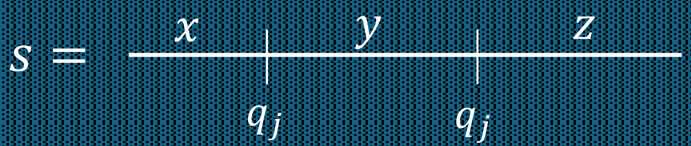
Pumping Lemma: For every regular language A, there is a number p (the "pumping length") such that if $s \in A$ and $|s| \geq p$ then $s = xyz$ where

1) $xy^iz \in A$ for all $i \geq 0$
2) $y \neq \varepsilon$
3) $|xy| \leq p$

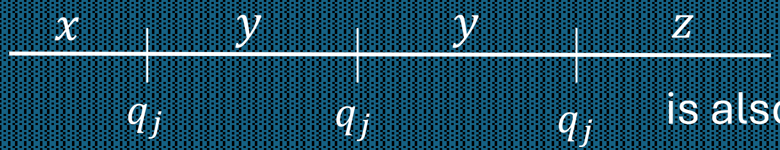
$$y^i = \underbrace{yy\cdots y}_{i}$$

Informally: A is regular → every long string in A can be pumped and the result stays in A.

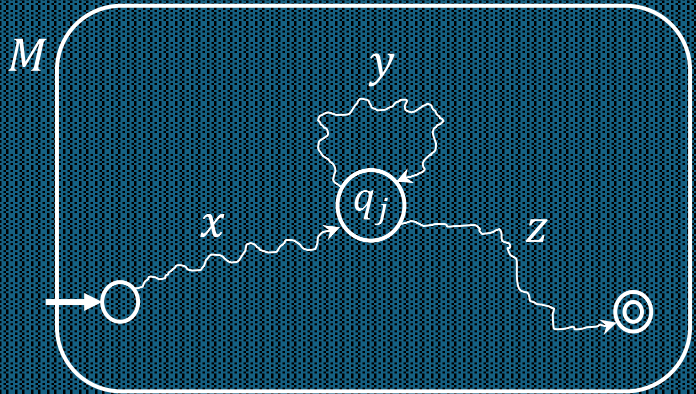
Proof: Let DFA M recognize A. Let p be the number of states in M. Pick $s \in A$ where $|s| \geq p$.



M will repeat a state q_j when reading s because s is so long.



is also accepted



The path that M follows when reading s.