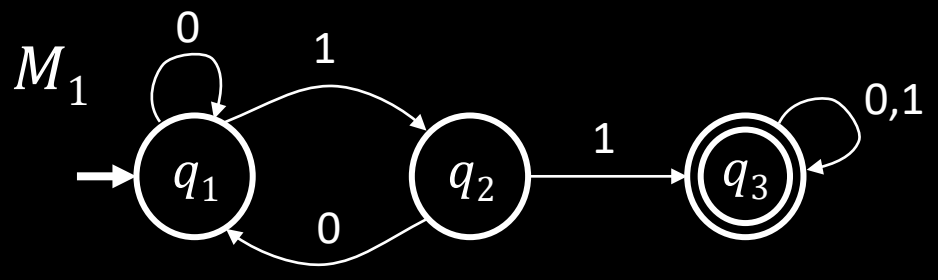
Intro to the Theory of Computation

Instructor: Mike Sipser

From lecture notes
https://ocw.mit.edu/courses/18-404j-theory-of-computation-fall-2020/

Finite Automata



States: $q_1\ q_2\ q_3$

Transitions: 1

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 $A = L(M_1)$.

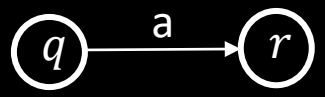
Finite Automata – Formal Definition

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

Q finite set of states

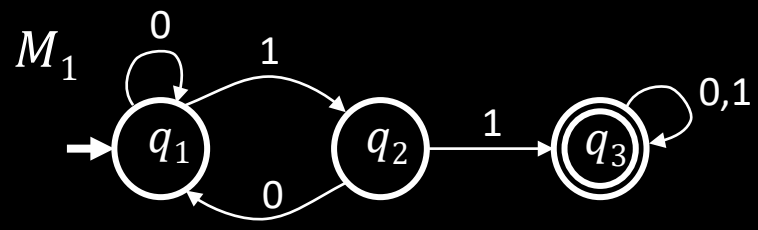
Σ finite set of alphabet symbols

δ transition function $\delta\colon Q \times \Sigma \rightarrow Q$

q_0 start state $\delta(q, a) = r$ means 

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

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 $\emptyset$ is the set with no strings

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

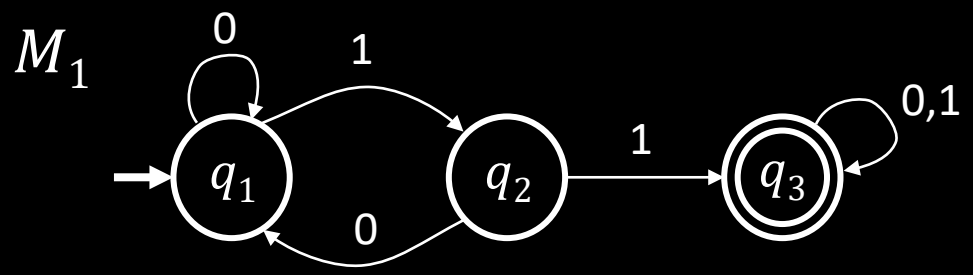
- $r_0 = q_0$
- $r_i = \delta(r_{i-1}, w_i)$ for $1 \leq i \leq n$
- $r_n \in F$

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



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

Therefore A is regular

More examples:

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

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

Goal: Understand the regular languages

Regular Expressions

Regular operations. Let A, B be languages:

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

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

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

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

Regular expressions

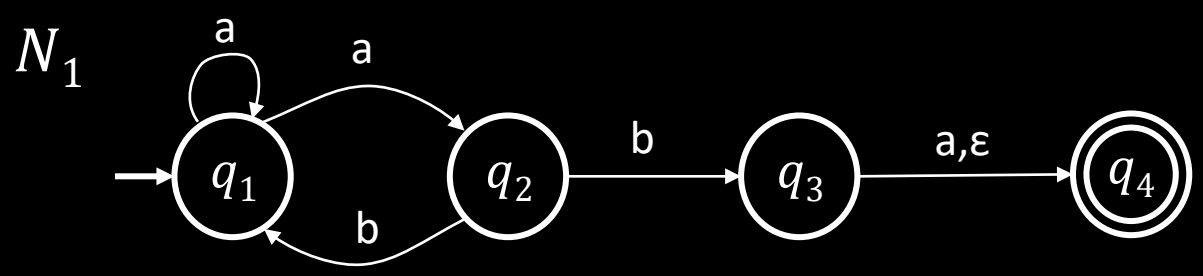
- Built from Σ, members $\Sigma, \emptyset, \varepsilon$ [Atomic]
- By using $\cup, \circ, *$ [Composite]

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

Nondeterministic Finite Automata



New features of nondeterminism:

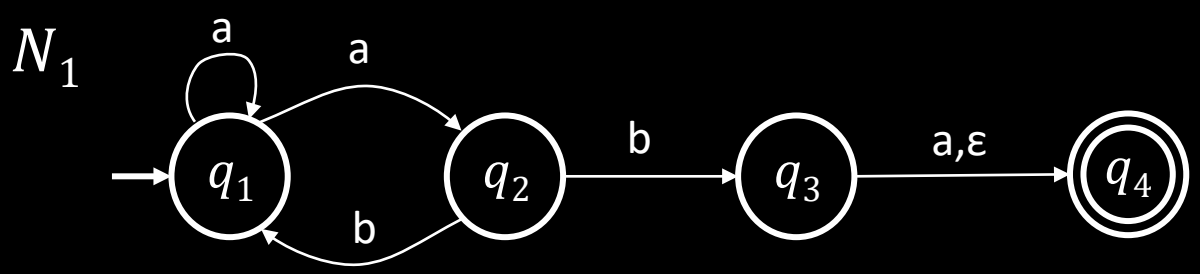
- multiple paths possible (0, 1 or many at each step)
- ε-transition is a "free" move without reading input
- Accept input if some path leads to accept ◎

Example inputs:

- ab accept
- aa reject
- aba accept
- abb reject

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

NFA – Formal Definition



Defn: A nondeterministic finite automaton (NFA)

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
- $\delta: Q \times \Sigma_\varepsilon \to \mathcal{P}(Q) = \{R | R \subseteq Q\}$
  - $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$
  - power set
- In the N_1 example: $\delta(q_1, \text{a}) = \{q_1, q_2\}$
  $\delta(q_1, \text{b}) = \emptyset$

Ways to think about nondeterminism:

Computational: Fork new parallel thread and accept if any thread leads to an accept state.

Mathematical: Tree with branches. Accept if any branch leads to an accept state.

Magical: Guess at each nondeterministic step which way to go. Machine always makes the right guess that leads to accepting, if possible.

Regular Expressions → NFA

Theorem: If R is a regular expr and $A = L(R)$ then A is regular

Proof: Convert R to equivalent NFA M:

If R is atomic: Equivalent M is:

$R = a$ for $a \in \Sigma$ 

$R = \varepsilon$ 

$R = \emptyset$ 

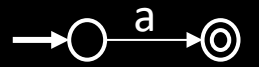
If R is composite:

$R = R_1 \cup R_2$
$R = R_1 \circ R_2$
$R = R_1^*$

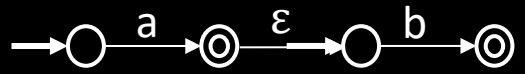
} Use closure constructions

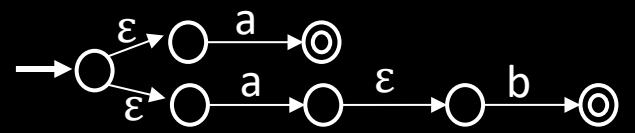
Example:

Convert $(\text{a} \cup \text{ab})^*$ to equivalent NFA

a: 

b: 

ab: 

a ∪ ab: 

$(\text{a} \cup \text{ab})^*$: