

Finite Automata Overview

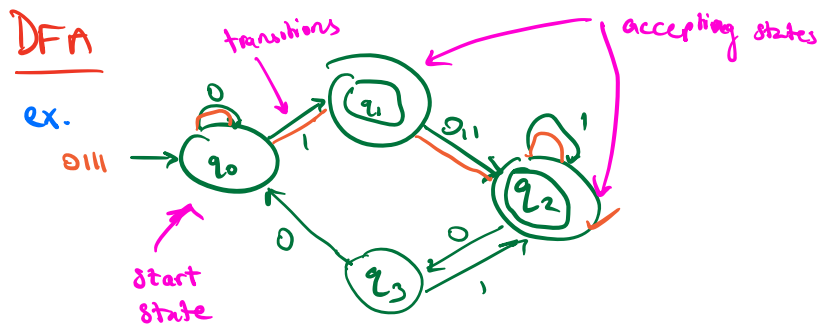
DFA Deterministic Finite Automata
recognizes (computes) regular languages (RL's)

NFA Nondeterministic Finite Automata
Arise from proving closure properties of RL's
Similar to DFA's, but allowed to make
"educated guesses"

Regular Expressions Define languages via closure properties
RE's recognize RL's, so do DFA + NFA

Later topics of focus

- Pumping Lemma
- Myhill-Nerode Theorem
- Streaming algorithms
- Communication complexity
- Randomness



Automaton accepts string if end on accepting state.
Otherwise, it rejects.

Why study DFA's?

- Constant-size memory
- Read-once input
- Relevant to streaming algos
which can only read once

vocabulary

alphabet Σ : finite set (i.e., $\Sigma = \{0, 1\}$)

string over Σ : finite length sequence of elements of Σ

Σ^* : set of all strings over Σ

For string x : $|x|$ length of x

ϵ : empty string

Language over Σ : set of strings over Σ
(subset of Σ^*)

OR boolean function over strings

Languages L correspond to functions

$f: \Sigma^* \rightarrow \{0, 1\}$

where $f(x) = 1$ if $x \in L$ else 0

Formalization

DFA is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$

where

Q : set of states (finite)

Σ : alphabet (finite)

$\delta: Q \times \Sigma \rightarrow Q$: transition fn

$q_0 \in Q$: start state

$F \subseteq Q$: set of accepting states

M accepts string $w = w_1 \dots w_n \in \Sigma^*$

if $\exists r_0 \dots r_n \in Q$ where

$r_0 = q_0$

$\delta(r_i, w_{i+1}) = r_{i+1} \quad i = 0 \dots n-1$

$r_n \in F$

$L(M)$: set of strings accepted/recognized by M

Regular Languages

Language L' regular if

L' recognized by a DFA

(\exists DFA M where $L' = L(M)$)

Closure properties of RL's

Union: $A \cup B = \{w \mid w \in A \text{ or } w \in B\}$

Intersection: $A \cap B = \{w \mid w \in A \text{ or } w \in B\}$

Complement: $\neg A = \{w \in \Sigma^* \mid w \notin A\}$

Reverse: $A^R = \{w_k \dots w_1 \mid w_k \dots w_1 \in A, w_i \in \Sigma\}$

Concatenation: $A \cdot B = \{vw \mid v \in A, w \in B\}$

Star: $A^* = \{s_1 \dots s_k \mid k \geq 0, s_i \in A\}$

If A, B regular, so are

$A \cup B$

$A \cap B$

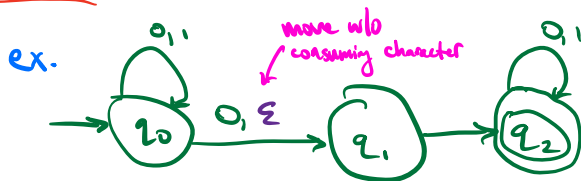
$\neg A$

A^R \Leftarrow this proof leads to define NFA's

$A \cdot B$

A^*

NFA's



accepts strings that contain 0

Allows multiple start states

(can use ϵ transitions to reduce to single start state)

formalization

NFA is a 5-tuple $N = (Q, \Sigma, \delta, Q_0, F)$

where

Q : set of states

Σ : alphabet

$\delta: Q \times \Sigma_{\epsilon} \rightarrow 2^Q$ transition function

↑
augment Σ to Σ_{ϵ} ← set of all subsets of Q

$Q_0 \subseteq Q$: set of start states

$F \subseteq Q$: set of accepting states

N accepts $w = w_1 \dots w_n \in \Sigma^* \cup \{\epsilon\}$

if $\exists r_0 \dots r_n \in Q$

where

$r_0 \in Q_0$

$r_{i+1} \in \delta(r_i, w_{i+1}) \quad i = 0 \dots n$

$r_n \in F$

differences w/ DFA

→ generally NFA much simpler

→ DFA accepts or rejects; NFA only needs one path to accept in order to accept