Formal Language and Automata Theory (CS21004)

Soumyajit Dey
CSE, IIT Kharagpur

Table of Contents

- The slide is just a short summary
- Follow the discussion and the boardwork
- Solve problems (apart from those we dish out in class)

Table of Contents

Finite Automata

- A finite automata is a mathematical model of a system with discrete inputs and outputs
- system can have finite no. of internal configs / states
- Simple Ex : Elevator controller : does not remember requests that are already serviced. Remembers 1) current floor, 2) current direction of motion, 3) set of requests to be serviced
- Complex Ex: a digital computer $\rightarrow$ infinite memory version is the Turing machine
- Day to day use by you : think of a text editor program (modes of operation : save file, open file, update user input with file in memory)

Finite Automata

$M = \langle Q, \Sigma, \delta, q, F \rangle$

- Q : finite set of **internal states**
- Σ : finite set of **input alphabet**
- $\delta : Q \times \Sigma \rightarrow Q$: is the **transition function**
- $q \in Q$: **initial state**
- $F \subseteq Q$ is the set of **final/accept states**

FA example

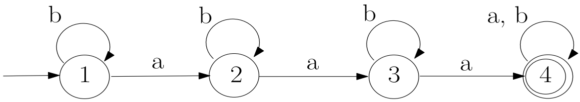


Figure: A sample FA

$$\Sigma = \{a, b\},\ Q = \{1, 2, 3, 4\},\ q = 1$$
$$\delta(1, a) = 2, \delta(1, b) = 1, \delta(2, a) = 3, \cdots$$

i/p is processed left to right

Acceptance by FA

Given $\delta : Q \times \Sigma \to Q$, define $\hat{\delta} : Q \times \Sigma^* \to Q$ inductively as follows

- $\hat{\delta}(q, \lambda) = q$
- For $x \in \Sigma^*, a \in \Sigma$, $\hat{\delta}(q, xa) = \delta(\hat{\delta}(q, x), a)$
- $\hat{\delta}$ is nothing but the multistep version of δ
- A string σ is **accepted** by $M = \langle Q, \Sigma, \delta, q_0, F \rangle$, if $\hat{\delta}(q, \sigma) \in F$
- $L(M) = \{\sigma \in \Sigma^* \mid \hat{\delta}(q, \sigma) \in F\}$
- A language A is regular iff it is accepted by some FA
- Regular sets are closed under $\cup, \cap, \neg, \circ, ^*$

Product Machines

Given $M_1 = \langle Q_1, \Sigma, \delta_1, q_1, F_1 \rangle$, $M_2 = \langle Q_2, \Sigma, \delta_2, q_2, F_2 \rangle$ let $M_3 = \langle Q_3, \Sigma, \delta_3, q_3, F_3 \rangle$ be defined as follows.

- $Q_3 = Q_1 \times Q_2$,
- $\delta_3 : Q_3 \times \Sigma \rightarrow Q_3$ where $\delta_3((p,q),a) = (\delta_1(p,a), \delta_2(q,a))$.
- $F_3 = F_1 \times F_2$
- $q_3 = (q_1, q_2)$

$L(M_3) = L(M_1) \cap L(M_2)$: Prove that $\forall \sigma \in \Sigma^*, \forall (p,q) \in Q_3, \hat{\delta}_3((p,q),\sigma) = (\hat{\delta}_1(p,\sigma), \hat{\delta}_2(q,\sigma))$ by induction on length of input string $\sigma \in \Sigma^*$. Use $\hat{\delta}_3$ to define $L(M_3)$ as desired.

With $F_3 = (F_1 \times Q_2) \cup (Q_1 \times F_2)$, we have an automata for $\cup$

Product Example

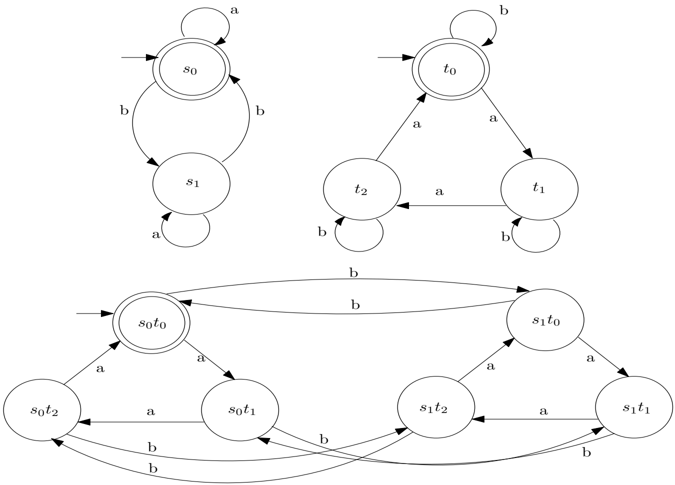
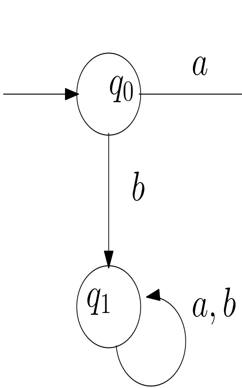


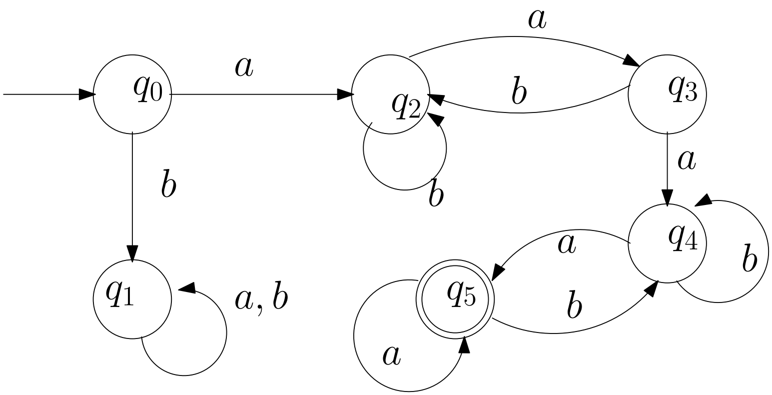
Figure: M_1 : even no. of b, M_2 : no of a is 3n. Technically they are sensitive to different alphabets, self loops highlight that

DFA for $L = \{awa \mid w \in \{a, b\}^*\}$



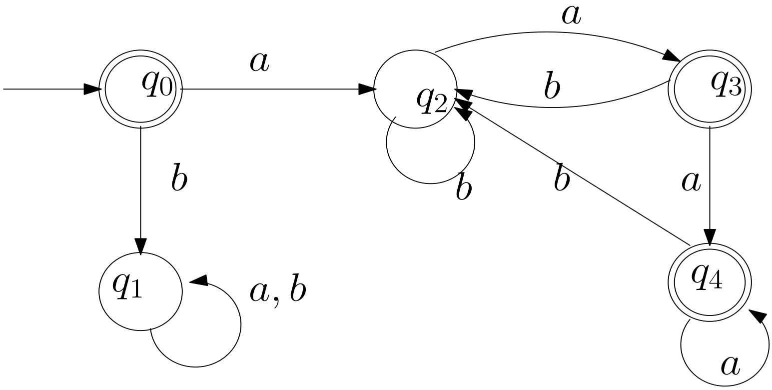
DFA for L^2

With $L = \{awa \mid w \in \{a,b\}^*\}$,
$L^2 = \{aw_1aaw_2a \mid w_1, w_2 \in \{a,b\}^*\}$

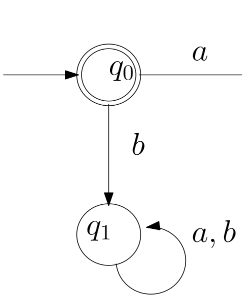


$L^n = \{(aw_ia)^n \mid w_1, \cdots, w_n \in \{a,b\}^*\}$

DFA for L^*



Smaller DFA for L^*



Why ? Note $L^n \subseteq L^{n-1} \subseteq \cdots \subseteq L^3 \subseteq L^2 \subseteq L$. So, $L^* = L^0 \cup L^1 \cup L^2 \cup \cdots = L^0 \cup L = \{\epsilon\} \cup L$. Just make initial state as accept state in machine for L

Table of Contents

NFA

- DFA is deterministic by construction since δ is function, (for all situations/states, how the machine will exactly react is uniquely specified)
- May be cumbersome to specify for complex systems
- May even be unknown
- NFA provides a nice way of ‘abstraction of information’ - a concept applied in so many aspects of computing

NFA

Consider $L = \{x \in \{0,1\}^* \mid \text{2nd symbol from the right is } 1\}$

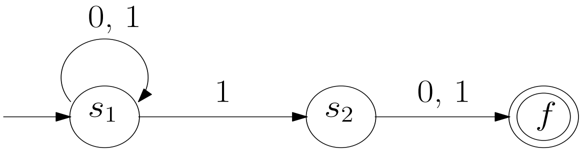
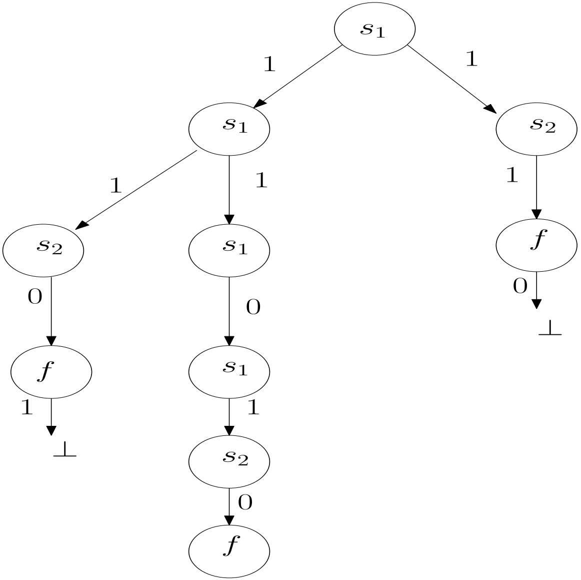


Figure: NFA for L

- the movement from s_1 when input is '1' is nondeterministic
- For an input string, NFA will create a 'computation tree' rather than a 'computation sequence' in case of DFA
- An NFA accepts a string if any one of the paths in the tree leads to some accept state

Computation tree for $\sigma = 11010$



NFA to DFA

Compute all possible **subsets** :
$\{\{\}, \{s_1\}, \{s_2\}, \{f\}, \{s_1, s_2\}, \{s_2, f\}, \{s_1, f\}, \{s_1, s_2, f\}\}$

- compute single step reachability among subsets for i/p-s 0,1 with same initial state

δ	0	1
$\{\}$	$\{\}$	$\{\}$
$\{s_1\}$	$\{s_1\}$	$\{s_1, s_2\}$
$\{s_2\}$	$\{f\}$	$\{f\}$
$\{f\}$	$\{\}$	$\{\}$
$\{s_1, s_2\}$	$\{s_1, f\}$	$\{s_1, s_2, f\}$
$\{s_1, f\}$	$\{s_1\}$	$\{s_1, s_2\}$
$\{s_2, f\}$	$\{f\}$	$\{f\}$
$\{s_1, s_2, f\}$	$\{s_1, f\}$	$\{s_1, s_2, f\}$

States $\{s_2, f\}$, $\{s_2\}$, $\{f\}$, $\{\}$, are *unreachable*

NFA to DFA

Reduced transition table with unreachable states removed

δ	0	1
$\{s_1\}$	$\{s_1\}$	$\{s_1, s_2\}$
$\{s_1, s_2\}$	$\{s_1, f\}$	$\{s_1, s_2, f\}$
$\{s_1, f\}$	$\{s_1\}$	$\{s_1, s_2\}$
$\{s_1, s_2, f\}$	$\{s_1, f\}$	$\{s_1, s_2, f\}$

DFA should have 4 states :
$\{a_1, a_2, a_3, a_4\} = \{\{s_1\}, \{s_1, s_2\}, \{s_1, f\}, \{s_1, s_2, f\}\}$
Initial state is same.

NFA to DFA

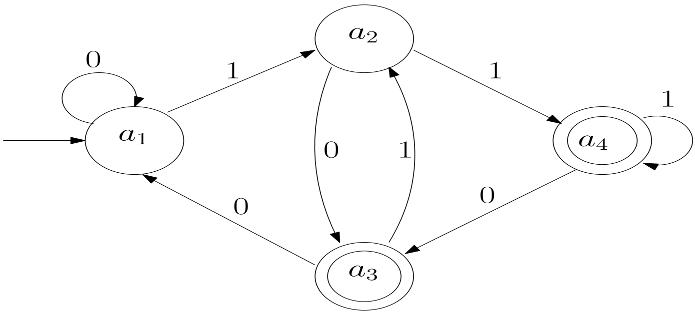
δ	0	1
$\{s_1\}$	$\{s_1\}$	$\{s_1, s_2\}$
$\{s_1, s_2\}$	$\{s_1, f\}$	$\{s_1, s_2, f\}$
$\{s_1, f\}$	$\{s_1\}$	$\{s_1, s_2\}$
$\{s_1, s_2, f\}$	$\{s_1, f\}$	$\{s_1, s_2, f\}$

$$\downarrow$$

δ	0	1
a_1	a_1	a_2
a_2	a_3	a_4
a_3	a_1	a_2
a_4	a_3	a_4

NFA to DFA

δ	0	1
a_1	a_1	a_2
a_2	a_3	a_4
a_3	a_1	a_2
a_4	a_3	a_4



NFA formal definition

$(Q, \Sigma, \Delta, S, F)$

- Q : set of states
- Σ : Alphabet
- Δ : transition relation (function) defined as

$$\Delta : Q \rightarrow 2^Q$$

  $2^Q = \{A | A \subseteq Q\}$: captures multiple possible reactions to an input
- $S \subseteq Q$: **set of initial states**