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

NFA formal definition

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

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

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

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

NFA : Acceptance

Extend Δ (inductively) to the multi-step version $\hat{\Delta} : 2^Q \times \Sigma^* \rightarrow 2^Q$ using the rules

- $\hat{\Delta}(A, \lambda) = A$ for any $A \subseteq Q$
- $\hat{\Delta}(A, xa) = \bigcup_{q \in \hat{\Delta}(A,x)} \Delta(q, a)$
- Example ??

Language of NFA

An NFA N accepts the language

$$L(N) = \{x \in \Sigma^* \mid \hat{\Delta}(S, x) \cap F \neq \Phi\}$$

Properties of $\hat{\Delta}$

- $\forall x, y \in \Sigma^*$ and $A \subseteq Q$, $\hat{\Delta}(A, xy) = \hat{\Delta}(\hat{\Delta}(A, x), y)$
  proof: by applying induction on $|y|$ and using definition of $\Delta, \hat{\Delta}$
- $\hat{\Delta}$ commutes with $\bigcup$: $\hat{\Delta}(\bigcup_i A_i, x) = \bigcup_i \hat{\Delta}(A_i, x)$, $A_i \subseteq Q$
  proof: by applying induction on $|x|$ and using definition of $\Delta, \hat{\Delta}$

$A \subseteq \Sigma$, $\hat{\Delta}(A, xy) = \hat{\Delta}(\hat{\Delta}(A, x), y)$

- Basis ($y = \lambda$): $\hat{\Delta}(A, x\lambda) = \hat{\Delta}(A, x) = \Delta(\hat{\Delta}(A, x), \lambda))$
- Induction step: For $x, y \in \Sigma^*$, $a \in \Sigma$, let $\hat{\Delta}(A, xy) = \hat{\Delta}(\hat{\Delta}(A, x), y)$

$$
\begin{aligned}
\hat{\Delta}(A, xya) &= \bigcup_{q \in \hat{\Delta}(A, xy)} \Delta(q, a) && \text{by definition of } \hat{\Delta} \\
&= \bigcup_{q \in \hat{\Delta}(\hat{\Delta}(A, x), y)} \Delta(q, a) && \text{induction hypothesis} \\
&= \hat{\Delta}(\hat{\Delta}(A, x), ya)) && \text{by definition of } \hat{\Delta}
\end{aligned}
$$

$\hat{\Delta}(\bigcup_i A_i, x) = \bigcup_i \hat{\Delta}(A_i, x),\ A_i \subseteq Q$

- $\hat{\Delta}(\bigcup_i A_i, \lambda) = \bigcup_i A_i = \bigcup_i \hat{\Delta}(A_i, \lambda)$
-
$$\begin{aligned}
\hat{\Delta}(\bigcup_i A_i, xa) &= \bigcup_{p \in \hat{\Delta}(\bigcup_i A_i, x)} \Delta(p, a) && \text{by definition of } \hat{\Delta} \\
&= \bigcup_{p \in \bigcup_i \hat{\Delta}(A_i, x)} \Delta(p, a) && \text{induction hypothesis} \\
&= \bigcup_i \bigcup_{p \in \hat{\Delta}(A_i, x)} \Delta(p, a) \\
&= \bigcup_i \hat{\Delta}(A_i, xa) && \text{by definition of } \hat{\Delta}
\end{aligned}$$

Formalize NFA $\Leftrightarrow$ DFA using $\hat{\Delta}$

Given NFA $N = (Q_N, \Sigma, \Delta_N, S_N, F_N)$, the **equivalent**[1] DFA $M = (Q_M, \Sigma, \delta_M, s_M, F_M)$ is defined as follows.

- $Q_M = 2^{Q_N}$
- $\delta_M(A, a) = \hat{\Delta}_N(A, a)$
- $s_M = S_N$
- $F_M = \{A \subseteq Q_N \mid A \cap F_N \neq \perp\}$

[1] equivalence is based on language acceptance

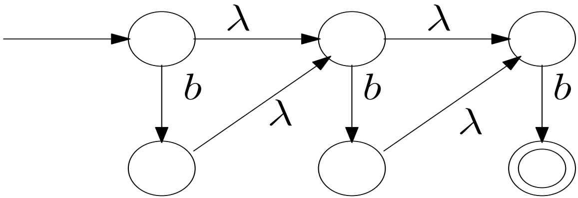
Formalize NFA $\Leftrightarrow$ DFA using $\hat{\Delta}$

Given NFA $N = (Q_N, \Sigma, \Delta_N, S_N, F_N)$, and the **equivalent** DFA $M = (Q_M, \Sigma, \delta_M, s_M, F_M)$,

- $\forall A \subseteq Q_N$ and any $x \in \Sigma^*$, $\hat{\delta}_M(A, x) = \hat{\Delta}_N(A, x)$ - show by induction on $|x|$
- $L(M) = L(N)$

λ transitions

- NFAs can have transitions like $s_1 \xrightarrow{\lambda} s_2$
- The machine can make a move without scanning ANY input.
- NFA + λ transitions $\equiv$ NFA w/o λ transitions, DFA
- $\Delta : Q \times \Sigma \cup \lambda \rightarrow 2^Q$



$L = \{b, bb, bbb\}$. With λ transitions it becomes trivial to construct FA for L^* given FA for L

NFA

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

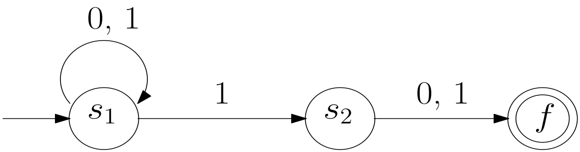


Figure: NFA for L

Let us revisit the NFA-DFA conversion example.

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/ps 0,1

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

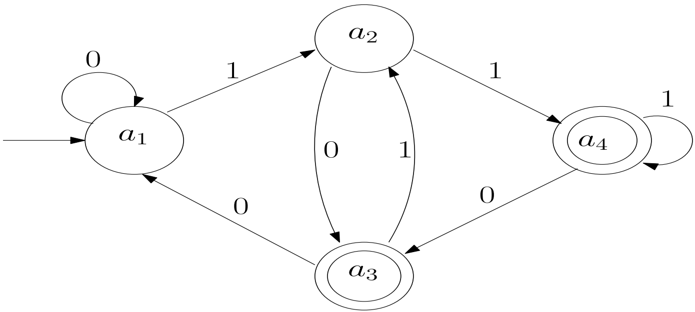
δ	0	1
$\{s_1\}$	$\{s_1\}$	$\{s_1, s_2\}$
$\{s_1, s_2\}$	$\{s_1, f\}$	$\{s_1, s_2, f\}$
$\{s_1, f\}$	$\{s_1\}$	$\{s_1, s_2\}$
$\{s_1, s_2, f\}$	$\{s_1, f\}$	$\{s_1, s_2, f\}$

$$\downarrow$$

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

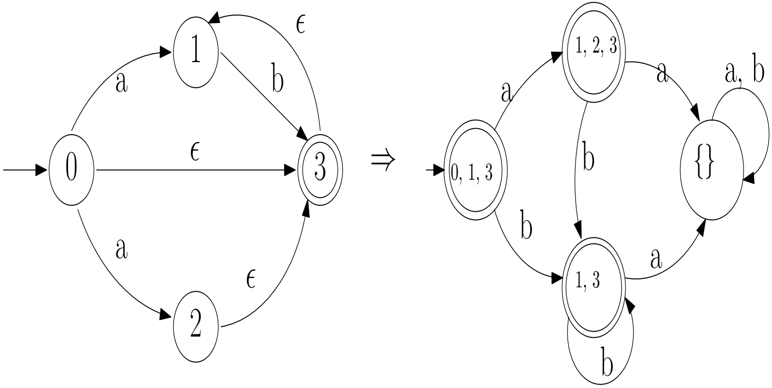


NFA to DFA : handle λ transitions

λ-closure of set S of states in NFA $=$ collection of states reachable from any state in S using only λ transitions $=$ λ-close(S) say.

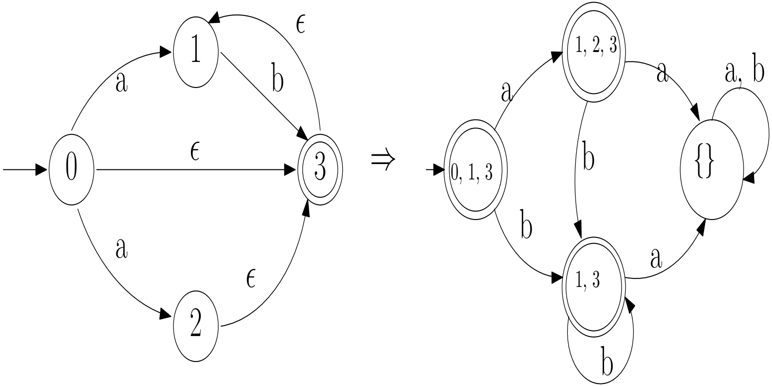
- Compute λ-close(S) where S is NFA initial state - this is DFA initial state I
- Keep computing λ-closures for every reachable state

NFA to DFA : handle λ transitions



$$I = \lambda - close(0) = \{0, 1, 3\}$$

NFA to DFA : handle λ transitions

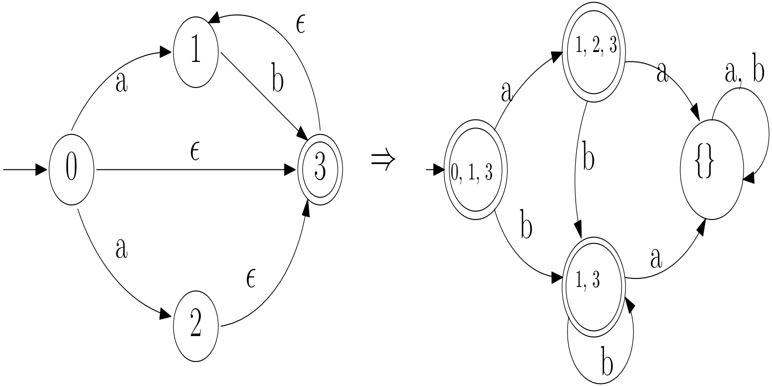


$$\lambda\!-\!close(\{\Delta(0,a),\Delta(1,a),\Delta(3,a)\}) = \lambda\!-\!close(\{1,2\}) = \{1,2,3\}$$

Hence

$$\{0,1,3\} \xrightarrow{a} \{1,2,3\}$$

NFA to DFA : handle λ transitions



$$\lambda - close(\{\Delta(1, a), \Delta(2, a), \Delta(3, a)\}) = \lambda - close(\{\}) = \{\}$$

Hence

$$\{1, 2, 3\} \xrightarrow{a} \{\}$$