Introduction to Formal Languages, Automata and Computability

Finite State Automata : Characterization, Properties and Decidability

K. Krithivasan and R. Rama

Finite State Automata and Regular Grammars

Theorem If a language L is accepted by a finite nondeterministic automaton, then L can be accepted by a right linear grammar and conversely.
Let L be a language accepted by a finite nondeterministic automaton $M = (K, \Sigma, \delta, q_0, F)$ where $K = \{q_0, \ldots, q_n\}$. If $w \in L$, then w is obtained by the concatenation of symbols corresponding to different transitions starting from q_0 and ending at a finite state. Hence for each transition by M while reading a symbol of w, there must correspond to a production of a right linear grammar G. The construction is as below:

$$G = (\{S_0, S_1, \ldots, S_n\}, \Sigma, P, S_0)$$

contd

where productions in P are
(i) $S_i \rightarrow aS_j$ if $\delta(q_i, a)$ contains q_j for $q_j \notin F$
(ii) $S_i \rightarrow aS_j$ and $S_i \rightarrow a$ if $\delta(q_i, a)$ contains q_j, $q_j \in F$.
To prove $L(G) = L = L(M)$.
From the construction of P, one is able to see that $S_i \Rightarrow aS_j$ if and only if $\delta(q_i, a)$ contains q_j and $S_i \Rightarrow a$ if and only if $\delta(q_i, a) \in F$. Hence if $S_0 \Rightarrow a_1S_1 \Rightarrow a_1a_2S_2 \Rightarrow \cdots \Rightarrow a_1 \ldots a_n$ if and only if $\delta(q_0, a_1)$ contains q_1, $\delta(q_1, a_2)$ contains q_2, $\ldots$, $\delta(q_{n-1}, a_n)$ contains q_n where $q_n \in F$.

Hence $w \in L(G)$ if and only if $w \in L(M)$.

contd

Let $G = (N, T, P, S)$ be a right linear grammar. An equivalent NFSA with ϵ-moves is constructed as below:

Let $M = (K, T, \delta, [S], [\epsilon])$ where

$K = \{[\alpha] | \alpha$ is S or suffix of some right-hand side of a production in P, the suffix need not be proper$\}$.

The transition function δ is defined as follows:

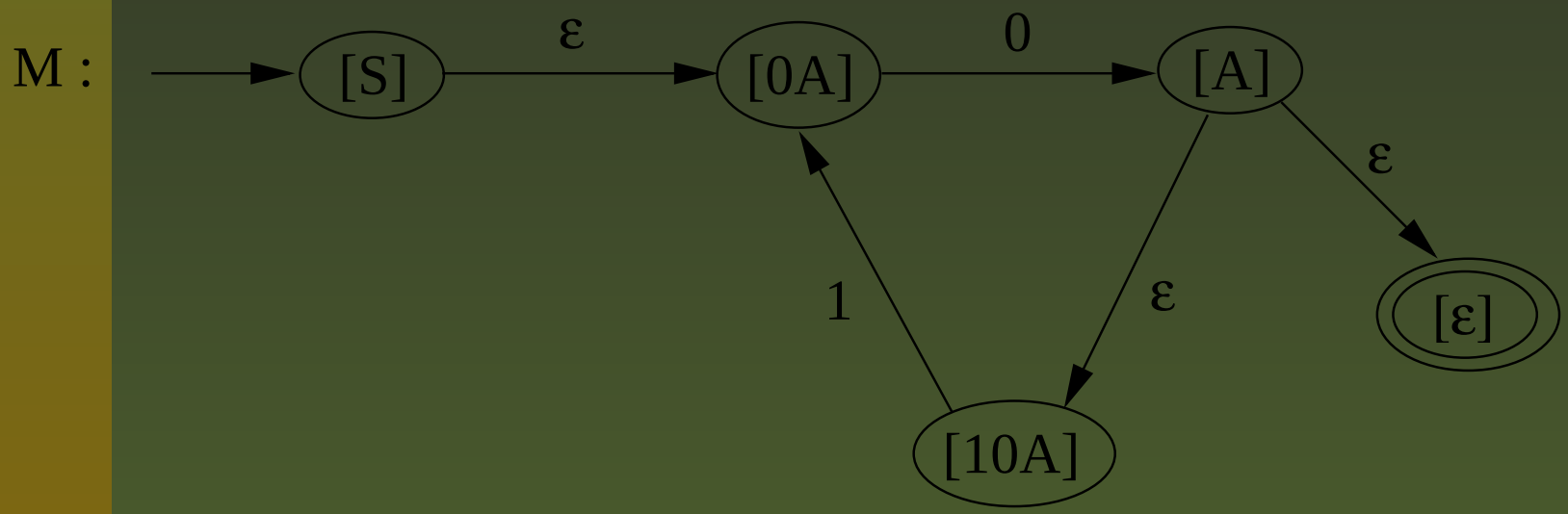
(i) $\delta([A], \epsilon) = \{[\alpha] | A \rightarrow \alpha \in P\}$

(ii) For $a \in T$ or $\alpha \in T^*N$, then $\delta([a\alpha], a) = \{[\alpha]\}$. Clearly $[\alpha] \in \delta([\alpha], w)$ if and only if $S \stackrel{*}{\Rightarrow} xA \Rightarrow xy\alpha$ where $A \rightarrow y\alpha \in P$ and $xy = w$. M accepts w if and only if $S \stackrel{*}{\Rightarrow} xA \Rightarrow xy = w$, $w \in T^*$.

Hence the converse follows.

Example

Let $G = (\{S, A\}, \{0, 1\}, \{S \rightarrow 0A, A \rightarrow 10A/\epsilon\}, S)$ be a regular grammar. The corresponding NFSA will be



Clearly $L(G) = L(M) = 0(10)^*$.

Pumping Lemma for Regular Sets

Theorem[Pumping Lemma] Let L be a regular language over T. Then there exists a constant k depending on L such that for each $w \in L$ with $|w| \geq k$, there exists $x, y, z \in T^*$ such that $w = xyz$ and
(i) $|xy| \leq k$
(ii) $|y| \geq 1$
(iii) $xy^iz \in L \ \ \forall i \geq 0$.
Let $M = (K, \Sigma, \delta, q_0, F)$ be a DFSA accepting L. Let $K = \{q_1, \ldots, q_n\}$. Let $w = a_1, \ldots, a_m \in L$ where $a_i \in \Sigma, 1 \leq i \leq m, m \geq k$.

Let the transitions on w be as below:

contd

$$q_1 a_1 \ldots a_m \vdash a_1 q_2 a_2 \ldots a_m \vdash \cdots \vdash a_1 \ldots a_m q_{m+1}$$

where $q_j \in K$, $1 \leq j \leq m+1$. Here $a_1 \ldots a_{i-1} q a_i \ldots a_m$ means the FSA is in state q after reading $a_1 \ldots a_{i-1}$ and the input head is pointing to a_i. Clearly in the above transitions, $m+1$ states are visited, but M has only n states. Hence there exists q_i, q_j such that $q_i = q_j$. Hence for

$$q_1 a_1 \ldots a_m \vdash a_1 q_2 a_2 \ldots a_m \vdash \cdots \vdash (a_1 \ldots a_{i-1} q_i a_i \ldots a_m \ldots$$
$$\vdash a_1 \ldots a_{j-1} q_i a_j \ldots a_m) \vdash \cdots \vdash a_1 \ldots a_m q_{m+1}$$

contd

end at q_i, where the transitions between the brackets start and processing a string α^t for $t \geq 0$. Hence if $x = a_1 \ldots a_{i-1}$, $y = a_i \ldots a_j$, $z = a_{j+1} \ldots a_m$, $xy^t z \in L \ \ \forall t \geq 0$ where $|xy| \leq m$, since q_i is the first state identified to repeat in the transition and $|y| \geq 1$. Hence the lemma.

Example

Let $L = \{a^n b^n | n \leq 1\}$. If L is regular, then by the above lemma there exists a constant 'k' satisfying the pumping lemma conditions. Choose $w = a^k b^k$. Clearly $|w| > k$. Then $w = xyz$, $|xy| \leq k$ and $|y| \geq 1$. If $|x| = p$, $|y| = q$, $|z| = r$, $p + q + r = 2k$ and $p + q \leq k$. Hence xy consists of only a's and since $|y| > 1$, $xz \notin L$ as number of a's in x is less than k and $|z| = k$. Hence pumping lemma is not true for $i = 0$ as $xy^i z$ must be in L for $i \geq 0$. Hence L is not regular.

One can see that not only regular languages satisfy the pumping lemma property, but also some nonregular languages do so.

example

Let $L = \{a^n b^n | n \leq 1\}$. We know L is nonregular. Consider $L_{\#} = (\#^+ L) \cup \{a, b\}^*$ where $\# \notin \{a, b\}$. $L_{\#}$ satisfies all the properties of pumping lemma with $k = 1$. For any $w \in \#^+ L$, let $x = \lambda$, $y = \#$ and for any $w \in \Sigma^*$, $x = \lambda$ and y is the first letter of w. However $L_{\#}$ is not regular, which can be seen as below. Let h be a homomorphism defined as below: $h(a) = a$ for each $a \in \Sigma$ and $h(\#) = \lambda$. Then $L = h(L_{\#} \cap \#^+ \Sigma^*)$. Clearly $\#^+ \Sigma^*$ regular. If $L_{\#}$ is regular, then $L_{\#} \cap \#^+ \Sigma^*$ is regular as regular languages are closed under intersection. Also regular languages are closed under homomorphism and hence L is regular which is a contradiction. Hence $L_{\#}$ is nonregular.

Closure Properties

Theorem The family of regular languages is closed under the following operations (1) union (2) intersection (3) complementation (4) catenation (5) star and (6) reversal.

The six closure properties will be proved either through finite automaton or regular grammars.

Union : Let L_1 and L_2 be two regular languages generated by two right linear grammars $G_1 = (N_1, T_1, P_1, S_1)$ and $G_2 = (N_2, T_2, P_2, S_2)$ (say). Without loss of generality let $N_1 \cap N_2 = \phi$. $L_1 \cup L_2$ is generated by the right linear grammar. $G' = (N_1 \cup N_2 \cup \{S\}, T_1 \cup T_2, P_1 \cup P_2 \cup \{S \rightarrow S_1, S \rightarrow S_2, S\})$. $L(G') = L(G_1) \cup L(G_2)$ because, the new start symbol of G' is S from which we reach S_1

contd

or S_2 using the rules $S \to S_1, S \to S_2$. After this step one can use only rules from P_1 or P_2, hence deriving words in L_1 or L_2 or in both.

Intersection : Let L_1, L_2 be any two regular languages accepted by two DFSA's $M_1 = (K_1, \Sigma_1, \delta_1, q_1, F_1)$ and $M_2 = (K_2, \Sigma_2, \delta_2, q_2, F_2)$. Then the DFSA M constructed as below accepts $L_1 \cap L_2$.

Let $M = (K, \Sigma, \delta, q_0, F)$ where $K = K_1 \times K_2$, $q_0 = (q_1, q_2)$, $F = F_1 \times F_2$, $\delta : K \times \Sigma \to K$ is defined by $\delta((p_1, p_2), a) = (\delta_1(p_1, a), \delta_2(p_2, a))$.

One can see that for each input word w, M runs M_1 and M_2 parallely, starting from q_1, q_2 respectively. Having finished reading the input, M accepts only if both M_1, M_2 accept. Hence $L(M) = L(M_1) \cap L(M_2)$.

contd

Complementation : Let L_1 be a regular language accepted by DFSA $M = (K, \Sigma, \delta, q_0, F)$. Then clearly the complement of L is accepted by the DFSA $M^c = (K, \Sigma, \delta, q_0, K - F)$.

Concatenation : We prove this property using the concept of regular grammar. Let $L_1$and L_2 and G_1 and G_2 be defined as in proof of union of this theorem. Then the type 3 grammar G constructed as below satisfies the requirement that $L(G) = L(G_1).L(G_2)$. $G = (N_1 \cup N_2, T_1 \cup T_2, S_1, P_2 \cup P)$ where $P = \{A \rightarrow aB/A \rightarrow aB \in P_1\} \cup \{A \rightarrow aS_2 | A \rightarrow a \in P_1\}$.

contd

Clearly $L(G) = L(G_1).L(G_2)$ because any derivation starting from S_1 derives a word $w \in L_1$ and for G, $S_1 \stackrel{*}{\Rightarrow} wS_2$. Hence if $S_2 \stackrel{*}{\Rightarrow} w'$ by G_2, then $S_1 \stackrel{*}{\Rightarrow} ww'$ by G.

Catenation Closure : Here also we prove the closure using regular grammar. Let L_1 be a regular grammar generated by $G_1 = (N_1, T_1, P_1, S_1)$. Then the type 3 grammar $G = (N_1 \cup \{S_0\}, T_1, S_0, \{S_0 \rightarrow \epsilon, S_0 \rightarrow S_1\} \cup \{A \rightarrow aS_1 | A \rightarrow a \in P_1\} \cup P_1)$. Clearly G generates L_1^*.

Reversal : The proof is given using the NFSA model. Let L be a language accepted by a NFSA with ϵ-transitions which has exactly one final state.

contd

(Exercise : For any NFSA, there exists an equivalent NFSA with ϵ-transitions with exactly one final state). Let it be $M = (K, \Sigma, \delta, q_0, \{q_f\})$. Then the reversal automaton $M' = (K, \Sigma, \delta', q_f, \{q_0\})$ where δ' is defined as $\delta'(q, a)$ contains p if $\delta(p, a)$ contains q for any $p, q \in K, a \in \Sigma \cup \{\epsilon\}$. One can see that if $w \in L(M)$ then $w^R \in L(M')$ as the in the modified automaton M' each transition takes a backward movement on w.

contd

Theorem Regular languages are closed under homomorphism.

Let $\boldsymbol{r}$ be the regular expression for the regular language L and h be the homomorphism. $h(\boldsymbol{r})$ is an expression obtained by substituting $h(a)$ for each symbol a in $\boldsymbol{r}$. Clearly $h(a)$ is a regular expression. Hence $h(\boldsymbol{r})$ is a regular expression. For every $w \in L(\boldsymbol{r})$ the corresponding $h(w)$ will be in $L(h(\boldsymbol{r}))$ and conversely.

Theorem Let L_1 and L_2 be two regular languages. Then L_1/L_2 is also regular.

Let L_1 be accepted by a DFSA $M_1 = (K_1, \Sigma_1, \delta_1, q_1, F_1)$.

Let $M_i = (K_1, \Sigma_1, \delta_1, q_i, F_1)$ be a DFSA with q_i as

contd

its initial state, for each $q_i \in K_1$. Construct an automaton $\overline{M}$ that accepts $L_2 \cap L(M_i)$. If there is a successful path from the initial state of this automaton $\overline{M}$ to its final states, then $L_2 \cap L(M_i)$ is not empty. If so add q_i to $\overline{F}$.

Let $\overline{M} = (K_1, \Sigma_1, \delta, q_0, \overline{F})$. One can see that $L(\overline{M}) = L_1/L_2$ for if $x \in L_1/L_2$ whenever for any $y \in L_2$, $\delta_1(q_0, xy) \in F_1$. Hence $\delta(q_0, x) = q$ and $\delta(q, y) \in F_1$.

Conversely if $x \in L(\overline{M})$, then $\delta(q_0, x) = q$, $q \in \overline{F}$. By construction there exists a $y \in L_2$ such that $\delta(q, y) \in F_1$, implying $xy \in L_1$ and $x \in L_1/L_2$. Hence the proof.

contd

Theorem Let $h : \Sigma_1^* \rightarrow \Sigma_2^*$ be a homomorphism. If $L' \subseteq \Sigma_2^*$ is regular, then $h^{-1}(L') = L \subseteq \Sigma_1^*$ will be regular.

Let $M = (K, \Sigma_2, \delta, q_0, F)$ be a DFSA such that $L(M) = L'$. We construct a new FSA M' for $h^{-1}(L') = L$ from M as below:

Let $M' = (K, \Sigma_1, \delta', q_0, F)$ be such that K, q_0, F are as in M.

The construction of the transition function δ' is defined as:

$$\delta'(q, a) = \delta(q, h(a)) \text{ for } a \in \Sigma_1.$$

i.e., Here $h(a)$ is a string over Σ_2.

contd

For if $x \in \Sigma_1^*$, and $x = \epsilon$,
$\delta'(q, x) = \delta(q, h(x))$
i.e., $\delta'(q, \epsilon) = q = \delta(q, h(\epsilon)) = \delta(q, \epsilon)$.
If $x \neq \epsilon$, let $x = x'a$, then

$$\begin{aligned}
\delta'(q, x'a) &= \delta'(\delta'(q, x'), a) \\
&= \delta'(\delta(q, h(x')), a) \\
&= \delta(\delta(q, h(x')), h(a)) \\
&= \delta(\delta(q, h(x')h(a)) \\
&= \delta(q, h(x'a)).
\end{aligned}$$

Hence one can see that $L(M') = h^{-1}(L(M))$ for any

contd

input $x \in \Sigma_1$. i.e.,

$$\begin{aligned} x \in L(M') \quad & \text{iff} \quad \delta'(q_0, x) \in F \\ & \text{iff} \quad \delta(q_0, h(x)) \in F \\ & \text{iff} \quad h(x) \in L(M) \\ & \text{iff} \quad x \in h^{-1}(L(M)). \end{aligned}$$

Any family of languages which is closed under the six basic operations of union, concatenation, Kleene closure, ϵ-free homomorphism, intersection with regular sets and inverse homomorphism is called an Abstract Family of Languages (AFL).

contd

The family of regular sets is an AFL. This is seen from the above closure properties. If a family is closed under union, concatenation, Kleene closure, arbitrary homomorphism, intersection with regular sets and inverse homomorphism, it called a full AFL. If a family of languages is closed under intersection with regular set, inverse homomorphism and ϵ-free homomorphism, it is called a trio. If a family of languages is closed under all homomorphisms, as well as inverse homomorphism and intersection with a regular set, then it is said to be a full trio. The family of regular sets is a full trio and a full AFL.

Decidability Theorems

Theorem Given a regular language L over T and $w \in T^*$, there exists an algorithm for determining whether or not w is in L.

Let L be accepted by a DFSA M (say). Then for input w one can see whether w is accepted by M or not. The complexity of this algorithm is $O(n)$ where $|w| = n$. Hence membership problem for regular sets can be solved in linear time.

Theorem There exists an algorithm for determining whether a regular language L is empty, finite or infinite.

Let M be a DFSA accepting L. In the state diagram representation of M with inaccessible states from the initial state removed, one has to check whether there is a simple

contd

directed path from the initial state of M to a final state. If so, L is not empty. Consider a DFSA M' accepting L, where inaccessible states from the initial state are removed and also states from which a final state cannot be reached are removed.
If in the graph of the state diagram of the DFSA, there are no cycles, then L is finite. Otherwise L is infinite.

One can see that the automaton accepts sentences of length less than n, (where n is the number of states of the DFSA) if and only if $L(M)$ is nonempty. One can prove this statement using pumping lemma.

contd

That is $|w| < n$ for if w were the shortest and $|w| \geq n$ then $w = xyz$ and xz is shorter than w that belong to L.

Also L is infinite if and only if the automaton M accepts at least one word of length l where $n \leq l < 2n$. One can prove this by using pumping lemma. If $w \in L(M)$, $|w| \geq n$ and $|w| \leq 2n$, directly from pumping lemma, L is infinite. Conversely if L is infinite, we show that there should be a word in L whose length is l where $n \leq l < 2n$. If there is no word whose length is l, where $n \leq l < 2n$, let w be the word whose length is at least

contd

$2n$, but as short as any word in $L(M)$ whose length is greater than or equal to $2n$. Then by pumping lemma, $w = w_1w_2w_3$ where $1 \leq |w_2| \leq n$ and $w_1w_3 \in L(M)$. Hence either w was not shortest word of length $2n$ or more or $|w_1w_3|$ is between n and $2n - 1$, which is a contradiction.

Theorem For any two regular languages L_1 and L_2, there exists an algorithm to determine whether or not $L_1 = L_2$.

Consider $L = (L_1 \cap \bar{L}_2) \cup (\bar{L}_1 \cap L_2)$. Clearly L is regular by closure properties of regular languages.

contd

Hence there exists a DFSA M which accepts L. Now by the previous theorem one can determine whether L is empty or not. L is empty if and only if $L_1 = L_2$. Hence the theorem.