Tutorial 3

Formal Language and Automata Theory

January 19, 2023

Question 1

For any language L over Σ, the prefix closure of L is defined as

$$Pre(L) = \{x \ \in \ \Sigma^* \mid \exists y \in \ \Sigma^* \text{ such that } xy \ \in \ L\}$$

Prove that if L is regular, then so is $Pre(L)$.

Question 1

For any language L over Σ, the prefix closure of L is defined as

$$Pre(L) = \{x \ \in \ \Sigma^* \mid \exists y \in \ \Sigma^* \text{ such that } xy \ \in \ L\}$$

Prove that if L is regular, then so is $Pre(L)$.

HINT 1: Suppose L is recognized by a DFA A. We need to build a new DFA A_1 from A. A_1 has exact same states, initial state and symbols as A but different final states.

Question 1

For any language L over Σ, the prefix closure of L is defined as

$$Pre(L) = \{x \in \Sigma^* \mid \exists y \in \Sigma^* \text{ such that } xy \in L\}$$

Prove that if L is regular, then so is $Pre(L)$.

HINT 1: Suppose L is recognized by a DFA A. We need to build a new DFA A_1 from A. A_1 has exact same states, initial state and symbols as A but different final states.

HINT 2: For each state s of A, if there's an accepting state reachable from s, then the corresponding state in A_1 will be an accepting state. And A_1 will recognize Pre(L).

Solution:

The idea is: An automaton recognizing L can be transferred into one recognizing Pre(L) if we make all states accepting which lie on a path leading to an accepting state (we have to read y):

Let $A = (Q, \Sigma, \delta, q_0, F)$ a DFA recognizing L, we define F_1 by
$F_1 = \{q \in Q : \exists y \in \Sigma^*, \delta^*(q, y) \in F\}$
Then $A_1 = (Q, \Sigma, \delta, q_0, F_1)$ recognizes $Pre(L)$, as for $x \in \Sigma^*$ we have

$$\begin{aligned} x \in L(A_1) &\iff \delta^*(q_0, x) \in F_1 \\ &\iff \exists y : \delta^*(\delta^*(q_0, x), y) \in F \\ &\iff \exists y : \delta^*(q_0, xy) \in F \\ &\iff \exists y : xy \in L \\ &\iff x \in Pre(L) \end{aligned}$$

Question 2

Prove that $\forall L_1, L_2, (L_1 L_2)^R = L_2^R L_1^R$.

Question 2

Prove that $\forall L_1, L_2, (L_1 L_2)^R = L_2^R L_1^R$.

HINT 1: Show that if $\sigma \in (L_1 L_2)^R$, then $\sigma \in L_2^R L_1^R$ and vice versa.

Question 2

Prove that $\forall L_1, L_2, (L_1 L_2)^R = L_2^R L_1^R$.

HINT 1: Show that if $\sigma \in (L_1 L_2)^R$, then $\sigma \in L_2^R L_1^R$ and vice versa.

HINT 2: Use the principles of reverse and concatenation for the proof.

Question 2

Solution:
Let $\sigma \in (L_1L_2)^R$. Hence $\sigma^R \in L_1L_2$. Let $\sigma^R = xy$ such that $x \in L_1$ and $y \in L_2$. Now $\sigma = (xy)^R = y^Rx^R \in L_2^RL_1^R$.

Let $\sigma \in L_2^RL_1^R$. Let $\sigma = xy$ such that $x \in L_2^R$ and $y \in L_1^R$. Hence $\sigma^R = y^Rx^R \in L_1L_2$. Hence $\sigma \in (L_1L_2)^R$. Thus proved.

Hence $\sigma \in (L_1L_2)^R$. Thus proved.

Question 3

(a) For languages A and B, the shuffle of A and B is the language
$L = \{\omega \mid \omega = a_1 b_1 .....a_k b_k\}$ where $a_1, ..., a_k \in A$ and
$b_1, ..., b_k \in B, \forall a_i, b_i \in \Sigma^*$
Prove that the class of regular languages is closed under Shuffle operation.

Question 3

(a) For languages A and B, the shuffle of A and B is the language $L = \{\omega \mid \omega = a_1 b_1 .....a_k b_k\}$ where $a_1, ..., a_k \in A$ and $b_1, ..., b_k \in B, \forall a_i, b_i \in \Sigma^*$
Prove that the class of regular languages is closed under Shuffle operation.

HINT 1: Consider two DFAs M_A, M_B and try to construct the NFA N that represents the Shuffle operation.

Question 3

(a) For languages A and B, the shuffle of A and B is the language $L = \{\omega \mid \omega = a_1 b_1 .....a_k b_k\}$ where $a_1, ..., a_k \in A$ and $b_1, ..., b_k \in B, \forall a_i, b_i \in \Sigma^*$

Prove that the class of regular languages is closed under Shuffle operation.

HINT 1: Consider two DFAs M_A, M_B and try to construct the NFA N that represents the Shuffle operation.

HINT 2: N will be obtained by a modified cross-product construction. Think about how to formulate the transition function of N according to the problem definition.

Question 3

Solution:
Let $M_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ be a DFA recognizing A and $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ be a DFA recognizing B. The NFA for shuffle of A and B will simulate both M_A and M_B on the input, while non-deterministically choosing which machine to run on a particular input symbol. So the NFA N will be obtained by a modified cross-product construction. Formally, let $N = (Q, \Sigma, \delta, q_0, F)$, where

1. $Q = Q_A \times Q_B$
2. $q_0 = (q_A, q_B)$
3. $F = F_A \times F_B$
4. For $a \in \Sigma$, δ is given as
   $\delta((p_A, p_B), a) = \{(\delta_A(p_A, a), p_B), (p_A, \delta_B(p_B, a))\}$
   In all other cases, δ is ϕ

Question 3

At each step, the machine changes p_A according to δ_A or p_B according to δ_B. It reaches a state in $F = F_A \times F_B$ if and only if the moves according to δ_A take it from q_A to a state in F_A, and the ones according to δ_B take it from q_B to a state in F_B. Hence N accepts exactly the language Shuffle(A,B).

Question 3

(b) Let B and C be languages over $\Sigma = \{0, 1\}$. We have defined a language $L = B \leftarrow C$ as $L = \{\omega \ \in \ B|$ for some $y \ \in \ C$, strings ω and y contain equal numbers of 1's $\}$. Show that the class of regular languages is closed under the $\leftarrow$ operation.

Question 3

(b) Let B and C be languages over $\Sigma = \{0, 1\}$. We have defined a language $L = B \leftarrow C$ as $L = \{\omega \ \in \ B|$ for some $y \ \in \ C$, strings ω and y contain equal numbers of 1's $\}$. Show that the class of regular languages is closed under the $\leftarrow$ operation.

HINT 1: Consider two DFAs M_B, M_C and try to construct the NFA M that represents the $\leftarrow$ operation.

Question 3

(b) Let B and C be languages over $\Sigma = \{0, 1\}$. We have defined a language $L = B \leftarrow C$ as $L = \{\omega \ \in \ B|$ for some $y \ \in \ C$, strings ω and y contain equal numbers of 1's $\}$. Show that the class of regular languages is closed under the $\leftarrow$ operation.

HINT 1: Consider two DFAs M_B, M_C and try to construct the NFA M that represents the $\leftarrow$ operation.

HINT 2: To decide whether its input ω is in $B \leftarrow C$, the machine M checks that $\omega \ \in \ B$, and in parallel, non-deterministically guesses a string y that contains the same number of 1's as contained in ω and checks that $y \ \in \ C$.

Question 3

Solution: Let $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ and $M_C = (Q_C, \Sigma, \delta_C, q_C, F_C)$ be DFAs recognizing B and C respectively. Construct NFA $M = (Q, \Sigma, \delta, q_0, F)$ that recognizes $B \leftarrow C$ as follows.

1. $Q = Q_B \times Q_C$
2. For $(q, r) \in Q$ and $a \in \Sigma$ define $\delta((q, r), a)$
   $\{(\delta_B(q, 0), r)\}$ if a=0
   $\{(\delta_B(q, 1), \delta_C(r, 1))\}$ if a=1
   $\{(q, \delta_C(r, 0))\}$ if a=ϵ
3. $q_0 = (q_B, q_C)$
4. $F = F_B \times F_C$

Question 3

(c) A homomorphism is a mapping h with domain Σ^* for some alphabet Σ which preserves concatenation: $h(v.w) = h(v).h(w)$. Prove that the class of regular languages is closed under Homomorphism operation.

Question 3

Hint : Regular languages are closed under homomorphism, i.e., if L is a regular language and h is a homomorphism, then $h(L)$ is also regular.

- Define homomorphism as an operation on regular expressions
- Show that $L(h(R)) = h(L(R))$
- Let R be such that $L_1 = L(R)$. Let $R' = h(R)$. Then $h(L_1) = L(R')$.

Question 3

Solution:
For a regular expression R, let $h(R)$ be the regular expression obtained by replacing each occurrence of $a \in \Sigma$ in R by the string $h(a)$.
Example:
If $R = (0 \cup 1)^*001(0 \cup 1)^*$ and $h(0) = ab$ and $h(1) = bc$ then
$h(R) = (ab \cup bc)^*ababbc(ab \cup bc)^*$
Formally $h(R)$ is defined inductively as follows.

$$\begin{array}{ll} h(\phi) = \phi & h(R_1R_2) = h(R_1)h(R_2) \\ h(\epsilon) = \epsilon & h(R_1 \cup R_2) = h(R_1) \cup (R_2) \\ h(a) = h(a) & h(R^*) = h((R))^* \end{array}$$

Question 3

Claim: For any regular expression $R, L(h(R)) = h(L(R))$.
Proof: By induction on the number of operations in R

- Base Cases: For $R = \ \epsilon \ or \ \phi, h(R) = R$ and $h(L(R)) = L(R)$. For $R = a$, $L(R) = \{a\}$ and $h(L(R)) = \{h(a)\} = L(h(a)) = L(h(R))$. So claim holds.
- Induction Step: For $R = R_1 \cup R_2$, observe that $h(R) = h(R_1) \cup h(R_2)$ and $h(L(R)) = h(L(R_1) \cup L(R_2)) = h(L(R_1)) \cup h(L(R_2))$. By induction hypothesis, $h(L(R_i)) = L(h(R_i))$ and so
$h((L(R))) = L(h(R_1) \cup h(R_2))$
Other cases ($R = R_1 R_2$ and $R = R_1^*$) are similar.

Question 4

Consider $\Sigma = \left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}$. A string $\sigma \in \Sigma^*$ can be interpreted as two binary numbers, for example

$$\sigma = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 101100 \\ 010011 \end{bmatrix} = \begin{bmatrix} 2^0 + 2^2 + 2^3 \\ 2^1 + 2^4 + 2^5 \end{bmatrix} = \begin{bmatrix} 13 \\ 50 \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}$$

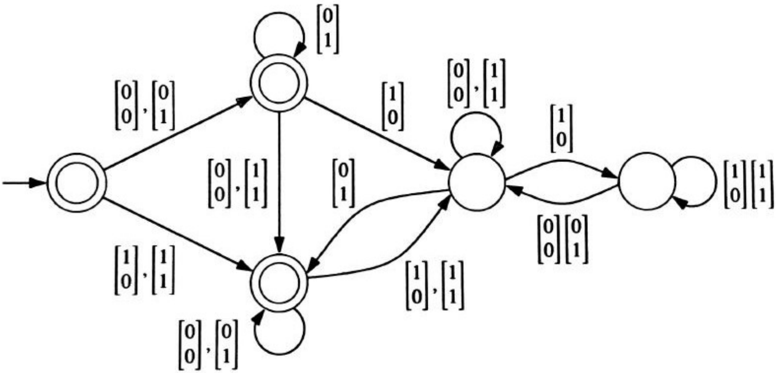
where $x, y \in \{0,1\}^*$. Design a DFA which accepts strings in Σ^* such that $2x - y \leq 2$. Note that for such a DFA, transitions will be labelled with elements from $\Sigma = \left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}$.

Question 4

HINT 1: Starting from the initial state, consider two paths to the next state, such that x starting with zeros will go to a state and x starting with ones will go to a different state. And continue from there onwards.

Solution:

The DFA is:



Question 5

Provide grammars for the following languages.

(a) $L_1 = \{a^n b^m \mid n \geq 0, m > n\}$

Question 5

(a) $L_1 = \{a^n b^m \mid n \geq 0, m > n\}$

HINT 1: Absence of ‘a’ is possible, but at least one ’b’ should be present.

Question 5

(a) $L_1 = \{a^n b^m \mid n \geq 0, m > n\}$

HINT 1: Absence of 'a' is possible, but at least one 'b' should be present

HINT 2: For each 'b' add an 'a' to keep track of the number of 'a'

Question 5

(a) $L_1 = \{a^n b^m \mid n \geq 0, m > n\}$

HINT 1: Absence of ‘a’ is possible, but at least one ‘b’ should be present

HINT 2: For each ‘b’ add an ‘a’ to keep track of the number of ‘a’

HINT 3: An infinite number of ‘b’ is possible

Question 5

Solution:

(a) Language: $L_1 = \{a^n b^m \mid n \geq 0, m > n\}$

$$
\begin{aligned}
S &\rightarrow Ab \\
A &\rightarrow \lambda \\
A &\rightarrow aAb \\
A &\rightarrow Ab
\end{aligned}
$$

Question 5

(b) $L_2 = \{a^n b^{2n} \mid n \geq 0\}$

Question 5

(b) $L_2 = \{a^n b^{2n} \mid n \geq 0\}$

HINT: For each ‘a’ two ‘b’ should be present. An empty string is also possible.

Question 5

Solution:

(b) Language: $L_2 = \{a^n b^{2n} \mid n \geq 0\}$

$$S \rightarrow \lambda$$
$$S \rightarrow aSbb$$

Question 5

(c) $L_1 \setminus \overline{L}$ where $L = \{a^n b^{n-3} \mid n \geq 3\}$

Question 5

(c) $L_1 \setminus \overline{L}$ where $L = \{a^n b^{n-3} \mid n \geq 3\}$

HINT: Given $L_1 = \{a^n b^m \mid n \geq 0, m > n\}$, try to understand the language $L_1 \setminus \overline{L}$. What should be the resulting strings from the language $L_1 \setminus \overline{L}$?

Question 5

(c) Language: $L_1 \setminus \overline{L}$ where $L = \{a^n b^{n-3} \mid n \geq 3\}$

Solution: Since $L_1 = \{a^n b^m \mid n \geq 0, m > n\}$, so $L_1 \setminus \overline{L} = \phi$. Thus, it represents an empty language.

$S \rightarrow \lambda$