자동장치이론 HW2

20181593 계인혜

2.4

a. $\{w \mid w\ \boldsymbol{contains\ at\ least\ three\ 1s}\}$

$G = (V, \Sigma, R, S)$ with set of variables V = {S,X}, where S is the start variable.

Finite set is Σ ={0,1} and rules R are as follows.

$S \rightarrow X1X1X1X$

$X \rightarrow 0X \mid 1X \mid \varepsilon$

b. $\{w \mid w\ \boldsymbol{starts\ and\ ends\ with\ the\ same\ symbol}\}$

$G = (V, \Sigma, R, S)$ with set of variables V = {S,X}, where S is the start variable.

Finite set is Σ ={0,1} and rules R are as follows.

$S \rightarrow 0X0 \mid 1X1$

$X \rightarrow 0X \mid 1X \mid \varepsilon$

c. $\{w \mid w\ \boldsymbol{the\ length\ of\ w\ is\ odd}\}$

$G = (V, \Sigma, R, S)$ with set of variables V = {S,X}, where S is the start variable.

Finite set is Σ ={0,1} and rules R are as follows.

$S \rightarrow 0X \mid 1X$

$X \rightarrow 00X \mid 01X \mid 10X \mid 11X \mid \varepsilon$

d. $\{w \mid w\ \boldsymbol{the\ length\ of\ w\ is\ odd\ and\ its\ middle\ symbol\ is\ a\ 0}\}$

$G = (V, \Sigma, R, S)$ with set of variables V = {S}, where S is the start variable.

Finite set is Σ ={0,1} and rules R are as follows.

$S \rightarrow 0S0 \mid 1S1 \mid 1S0 \mid 0S1 \mid 0$

e. $\{w \mid w = w^R, \boldsymbol{that\ is, w\ is\ a\ palindrome}\}$

$G = (V, \Sigma, R, S)$ with set of variables V = {S}, where S is the start variable.

Finite set is Σ ={0,1} and rules R are as follows.

$S \rightarrow 0S0 \mid 1S1 \mid 0 \mid 1 \mid \varepsilon$

f. ***The empty set***

$G = (V, \Sigma, R, S)$ with set of variables V = {S}, where S is the start variable.

Finite set is Σ ={0,1} and rules R are as follows.

$S \to S$

2.9

This language is the union of $A_1 = \{a^i b^j c^k \mid i, j, k \geq 0,\ i = j\}$ and $A_2 = \{a^i b^j c^k \mid i, j, k \geq 0,\ j = k\}$.

So we can create simple grammars for the individual languages and union them.

$G = (V, \Sigma, R, S)$ with set of variables $V = \{S, S_1, S_2, A, B\}$, where S is the start variable.

Rules R are as follows.

$S \to S_1 \mid S_2$

$For\ A_1,\qquad S_1 \to S_1 c \mid A \mid \varepsilon,\qquad A \to aAb \mid \varepsilon$

$For\ A_2,\qquad S_2 \to aS_2 \mid B \mid \varepsilon,\qquad B \to bBc \mid \varepsilon$

In case of A_1, we simply ensure that the number of a's equals the number of b's.

Likewise in A_2, we simply ensure that the number of b's equals the number of c's.

This grammar is ambiguous. For $x = a^n b^n c^n$, we may use either $S_1\ or\ S_2$ to generate x.

2.18

First, we have to describe a language of a CFG G and prove to be ambiguous.

An unambiguous grammar H has to be obtained from the grammar.

Consider the strings produced by the variable T. Let the CFG $I = (V, \Sigma, R, T)\ be : T \to aTb \mid ab$.

The language generated will consist of a sequence of two of more a's followed by the same number of b's because it's either a string of two terminals ab or it's placed between the same two terminals. So, $L(I) = \{a^i b^i \mid i > 1\}$.

And all of the string in this language will be of even length as $|a^i b^i| = 2i$.

As S can be replaced by SS or T, you can think $L(G)$will be the concatenation of one or more occurrences of strings of $L(I)$.

So the language $L(G) = \left\{a^{i^1} b^{i^1} a^{i^2} b^{i^2} \ldots a^{i^k} b^{i^k} \mid i^1, i^2, \ldots, i^k \geq 1, k \geq 1\right\}$.

Take the string $s = abaabbaaabbb$. There are two ways to derive string s.

1) $S \to SS \to TS \to abS \to abSS \to abTS \to abaTbS \to abaabbS \to abaabbT \to$

$abaabbaTb \to abaabbaaTbb \to abaabbaaabbb$

2) $S \to SS \to ST \to SaTb \to SaaTbb \to Saaabbb \to SSaaabbb \to STaaabbb \to$

$SaTbaaabbb \to Saabbaaabbb \to Taabbaaabbb \to abaabbaaabbb$

So the grammar G is ambiguous. The rules for the start symbol $S \to SS \mid T$.

Because of the rule, it's possible to get different derivations for a string in the language $L(G)$.

The grammar G can be converted into unambiguous grammar $H = (V, \sum, R', S)$.

The rules are modified as follows.

$S \to TS \mid T$

$T \to aTb \mid ab$

Now we have to check whether $L(G) = L(H)$. It means that a string s lies in $L(G) \leftrightarrow$ $the\ string\ s$ lies in $L(H)$.

($\to$)

The string ab lies in $L(G)$. It also be derived in $L(H)$, that is, $S \to T \to ab.$

All strings whose length is equal to or less than n of $L(G)$ lie in the language $L(H).$

Consider an arbitrary w string of length $n + 2$ lying in the language $L(G)$. There are three cases possible:

1) The string w starts with an ab. It can be derived in $L(H)$ to a string of length n as follows.

$$S \to SS \to TS \to abS \to abx$$

2) An ab in the middle of the string w. The derivation for this string as follows.

$$S \to SS \to SSS \to STS \to SaTbS \to xaybz$$

3) The string w ends with an ab. The derivation for this string as follows.

$$S \to SS \to ST \to Sab \to xab$$

The string can be derived in all the cases into a string whose length is at most.

Therefore, it has been proven if a string lies in $L(G)$ then it also lies in $L(H)$.

($\leftarrow$)

The string ab lies in $L(H)$. It also be derived in $L(G)$, that is, $S \to T \to ab.$

Consider an arbitrary w string of length $n + 2$ lying in the language $L(H)$. There are three cases possible:

1) The string w starts with an ab. It can be derived in $L(G)$ to a string of length n as follows.

$$S \rightarrow SS \rightarrow TS \rightarrow abS \rightarrow abx$$

2) An ab in the middle of the string w. The derivation for this string as follows.

$$S \rightarrow SS \rightarrow SSS \rightarrow STS \rightarrow SaTbS \rightarrow xaybz$$

3) The string w ends with an ab. The derivation for this string as follows.

$$S \rightarrow SS \rightarrow ST \rightarrow Sab \rightarrow xab$$

For the every cases, the string can be derived in grammar G to a string whose length $\leq n.$

So a string s lies in $L(G)$ if it lies in $L(H)$. Hence, it has been proven that the languages $L(H)$ and $L(G)$ are equivalent.

A grammar is unambiguous if there is only one leftmost derivation of a string in the grammar. An induction on the length of the string can be used to prove this and showing that is only one way to derive an arbitrary string into a string of smaller length.

2.43

Assume that B is context-free language and let p be its pumping length.

Let $s = 0^p1^{2p}0^p \in B$ with $|s| \geq$ p. By the pumping lemma, we can write x as $x = uvxyz$, such that $|vy| > 0, |vxy| \leq p \; and \; uxz \in B.$

If v and y both consist entirely of 1's, then $uxz = 0^p1^{2p-|vy|}0^p$ doesn't have the same number of 0s and 1s. Otherwise, v or y contains a nonzero amount of symbols m from one of the sets of 0s and some amount n from the set of 1s, but none from the other set of 0s, since $|vxy| \leq p.$

We have four cases as follows.

1) v : consists of m 0's and n 1's, y : consists entirely of 1's and m 0's are from the first zero set.

$$uxz = 0^{p-m}1^{2p-n-|y|}0^p$$

2) v : consists entirely of 0's, y : consists of m 0's and n 1's and m 0's are from the first zero set.

$$uxz = 0^{p-m-|v|}1^{2p-n}0^p$$

3) v : consists entirely of 1's, y : consists of m 0's and n 1's and m 0's are from the second zero set.

$$uxz = 0^p1^{2p-n-|v|}0^{p-m}$$

4) v : consists of m 0's and n 1's, y : consists entirely of 0's and m 0's are from the second zero set.

$$uxz = 0^p1^{2p-n}0^{p-m-|y|}$$

None of these is a palindrome since m,n are positive integers. In any case $uxz \notin B$.
Since we have a contradiction, it must be that B is not context free.

2.44

For a contradiction, suppose that C is context free.

So, C has a pumping length p. Let $s = 1^p 3^p 2^p 4^p \in C$ with |s|≥p.

By the pumping lemma, we can write x as $x = uvxyz$, such that $|vy| > 0, |vxy| \leq p\ and\ uv^i xy^i z \in C\ for\ all\ i \geq 0$.

Since $|vxy| \leq p$, there are 3 possible cases as follows.

1) vxy is substring of $1^p 3^p$

   In uv^2xy^2z, either the number of 1s is greater than the number of 2s, or the number of 3s is greater than the number of 4s, or both. Thus, $uv^2xy^2z \notin C$.

2) vxy is substring of $3^p 2^p$

   In uv^2xy^2z, either the number of 2s is greater than the number of as, or the number of 3s is greater than the number of 4s, or both. Thus, $uv^2xy^2z \notin C$.

3) vxy is substring of $2^p 4^p$

   In uv^2xy^2z, either the number of 2s is greater than the number of 1s, or the number of 4s is greater than the number of 3s, or both. Thus, $uv^2xy^2z \notin C$.

Therefore, in any case $uv^2xy^2z \notin C$. Since we have a contradiction, it must be that C is not context free.

2.56

Consider the two regular languages A and B over the input alphabet Σ. For the language A◇B, if PDA is constructed then it can be said that it is a CFL.

Consider the DFA $D_A = (Q_A, \Sigma, \delta_A, q_A, F_A)\ and\ D_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ for the languages A and B respectively. Construct the PDA $M = (Q, \Sigma, \Gamma, \delta, q_{start}, F)$ for the language A◇B formed by concatenating equal length strings from A and B.

- From the start state q_{start}, push the symbol '\$' into the stack to know the bottom of the stack.
- For every symbol from the language A, push 1 into the stack. It guesses the end of the string that belongs to A when it reaches to the final state F_A.
- For every symbol from the language B, pop 1 from the stack. When the string reaches the final state F_B, it moves to the final state F if and only if the top of the stack is '\$'.

The formal definition of the PDA $M = (Q, \sum, \Gamma, \delta, q_{start}, F)$ is as follows.

- $Q = Q_A \cup Q_B \cup \{q_{start}, F\}$
- $\sum$ is the input alphabet for A and B.
- q_{start} is the start state.
- F is the final state.
- $\Gamma = \{\$, 1\}$, where \$ is the empty marker and 1 is pushed every time into the stack when the symbol read from the language A.

The transition function is defined as follows.

$\delta(q_{start}, \varepsilon, \varepsilon) = \{q_A, \$\}$

$\delta(q, a, \varepsilon) = \{\delta_A(q, a), 1\} \qquad if\ q \in Q_{A,}\ a \in \sum$

$\delta(q, \varepsilon, \varepsilon) = \{q_B, \varepsilon\} \qquad if\ q \in F_A$

$\delta(q, a, 1) = \{\delta_B(q, a), \varepsilon\} \qquad if\ q \in Q_{B,}\ a \in \sum$

$\delta(q, \varepsilon, \$) = \{F, \varepsilon\} \qquad if\ q \in F_B$

Except these transitions, every other transitions will be rejected.

The PDA non-deterministically guesses the end of the string from D_A and transitions to the start symbol of D_B if it's in a final state of D_A. The PDA M accepts the string when it's in accepting state of D_B while hitting the empty stack. This shows that L(M) = A◇B. Therefore, for any two regular languages A and B, A◇B is a CFL.