COMPSCI 2AC3 Assignment 4

Prakhar Saxena

6 April 2024

Contents

Question 1

Part A

This language is not regualr. We will prove that using the Pumping Lemma Theorem.

For every CFL A, there exists $k \geq 0$ such that every $z \in A$ of length at least k can be broken up into five sub-strings: $z = uvwxy$, such that $uv^iwx^iy \in A$ and $|vwx| \leq k$, there exists an $i \geq 0$, $uv^iwx^iy \notin A$.

Essentially, the demon will pick some $k \geq 0$, we get to choose some $z \in A$ that is at least length k, and then, the demon will pick $uvwxy$ and we get to choose i such that $uv^iwx^iy \notin A$.

Use the Pumping Lemma for CFLs to show that $A = \{a^p \mid p \text{ is prime}\}$ is not context free. Once the demon picks k, we pick $z = a^k$ in which k is a prime number, which is in A, and $|z| \geq k$. The demon will then pick some u, v, w, x, y such that $z = uvwxy$, $vx \neq \epsilon$, and $|vwx| < k$. Next, if we find an i where $uv^iwx^iy \notin A$, let say we pick $i = 3$. This makes it uv^2wx^2y, $aaaaaaaaa$ which is not in A since, 9 is not prime.

Part B

The set is a CFL.
Below is the grammar for the CFL:

$$
\begin{aligned}
A &\to aA \mid \epsilon \\
B &\to bB \mid \epsilon \\
C &\to cC \mid \epsilon \\
Y &\to aYb \mid aA \mid bB \\
X &\to AZ \mid BY \\
Z &\to bZc \mid bB \mid cC
\end{aligned}
$$

Question 2

Let's first write a CFG for the given language, which is as follows.

$$
\begin{aligned}
S &\to A \mid B \mid AB \mid BA \\
A &\to a \mid CAC \\
B &\to b \mid CBC \\
C &\to a \mid b
\end{aligned}
$$

Now, let's just convert our grammar to NPDA.
Let Q be the set of states, Σ be the input alphabet, Γ be the stack alphabet, δ be the transition function, $s \in Q$ be the start state, $\perp \in \Gamma$ be the initial stack symbol, $F \subseteq Q$ be the set of final states.

$$
\begin{aligned}
Q &= \{q1, q2, q3\} \\
\Sigma &= \{a, b\} \\
\Gamma &= \{\perp, a, b, A, B\} \\
\perp &= S \\
F &= \{q3\}
\end{aligned}
$$

Finally, following is the δ -

$$
\begin{aligned}
\delta(q_0, \epsilon, \_) &= (q_1, S\perp) \\
\delta(q_1, \epsilon, S) &= (q_1, AB) \\
\delta(q_1, \epsilon, S) &= (q_1, BA) \\
\delta(q_1, \epsilon, S) &= (q_1, A) \\
\delta(q_1, \epsilon, S) &= (q_1, B) \\
\delta(q_1, a, A) &= (q_1, \epsilon) \\
\delta(q_1, b, B) &= (q_1, \epsilon) \\
\delta(q_1, a, C) &= (q_1, \epsilon) \\
\delta(q_1, b, B) &= (q_1, \epsilon) \\
\delta(q_1, \epsilon, \perp) &= (q_3, \perp)
\end{aligned}
$$

Question 3

Part A

Starting with:

$$S \rightarrow aSb \mid T$$
$$T \rightarrow bTaa \mid SS \mid \epsilon$$

First, we remove the epsilon and since, we are not allowed to have nonterminal generations, we take the values of T and put them in S. Then, we add the nonterminals A, B, X, Y and, Z to get.

$$A \rightarrow a$$
$$B \rightarrow b$$
$$X \rightarrow SB$$
$$Y \rightarrow AA$$
$$Z \rightarrow TY$$
$$S \rightarrow AX \mid BY \mid SS \mid BZ \mid AB$$
$$T \rightarrow BY \mid SS \mid BZ$$

Part B

Consider the following grammar written in CNF form:

$$A \rightarrow a$$
$$B \rightarrow b$$
$$X \rightarrow SB$$
$$Y \rightarrow AA$$
$$Z \rightarrow TY$$
$$S \rightarrow AX \mid BY \mid SS \mid BZ \mid AB$$
$$T \rightarrow BY \mid SS \mid BZ$$

Check to see if the string 'abbaa' can be derived from the above CNF.

The first step is to split this string into sub-parts by creating n + 1 lines to split apart each of the characters in the string. In this case, n is the length of string, which is 5 in this case, so we will need to create 6 lines to partition of the string. The string can be split like so: $|a|b|b|a|a|$

For $0 \leq i < j \leq n$, we say that $x_{i,j}$ denotes a sub-string from position i to j, for example, $x_{0,2}$ is the sub-string 'ab'. Now, construct a table T with $\binom{n}{2}$ entries, with entries for each possible $x_{i,j}$ pairs.

0					
–	1				
–	–	2			
–	–	–	3		
–	–	–	–	4	
–	–	–	–	–	5

To start, we fill in the top diagonal of the table with the non terminals that result in the terminals in the string. For each sub-string $s = x_{i,i+1}$, if there is a non terminal production X such that $X \to s \in G$, then we write that non terminal in that spot of the table.

0					
A	1				
–	B	2			
–	–	B	3		
–	–	–	A	4	
–	–	–	–	A	5

Now, we check to see if sub-strings of size 2 can be created. These are the strings in the spots $x_{i,i+2}$. To see if this sub-string can be formed, we look at $x_{i,i+1}$ and $x_{i+1,i+2}$ in the table. Let X represent the non-terminal in $x_{i,i+1}$ and Y represent the non-terminal in $x_{i+1,i+2}$. If there is a production in G such that $Z \to XY$, then we write the non-terminal Z in the position $x_{i,i+2}$ in the table. If there is no such production, then we write $\emptyset$.

0					
A	1				
S	B	2			
$-$	$\emptyset$	B	3		
$-$	$-$	$\emptyset$	A	4	
$-$	$-$	$-$	Y	A	5

Now, for creating strings of length 3 and above, the sections that need to be checked are different. We first check the first entry in the column, and the first entry in the row (entry next to sub string being created). If it is possible to create that sub-string, then we put the associated non-terminal in that spot, otherwise, we check the second term in the column and row. If one of the terms is $\emptyset$, then a sub string can not be created. If it is not possible to create a sub string using any of the pairings, then we write down $\emptyset$ in the table.
For example, for $x_{0,3}$, we check $x_{0,1}$ and $x_{1,3}$, $x_{0,2}$ and $x_{2,3}$. If we can create a sub-string using any of these pairings, then we write the non-terminal in that spot, so we write down C in the spot $x_{0,3}$. Similarly, for $x_{1,5}$ we check $x_{1,2}$ and $x_{2,5}$, $x_{1,3}$ and $x_{3,5}$. So we write B in the spot $x_{1,2}$ and S in$x_{2,5}$.

0					
A	1				
S	B	2			
X	$\emptyset$	B	3		
$\emptyset$	$\emptyset$	$\emptyset$	A	4	
$-$	$\emptyset$	S	Y	A	5

For $x_{0,5}$ we check $x_{0,1}$ and $x_{1,5}$, $x_{0,2}$ and $x_{2,5}$, $x_{0,3}$ and $x_{3,5}$, $x_{0,4}$ and $x_{4,5}$. So we write S in the spot $x_{0,5}$.

0					
A	1				
S	B	2			
C	$\emptyset$	B	3		
$\emptyset$	$\emptyset$	$\emptyset$	A	4	
S	$\emptyset$	S	F	A	5

By having the bottom left entry have an S, this implies that the string 'abbaa'

from $x_{0,n}$ can be derived from the start state, which implies that it can be created from the grammar.