COL352
Problem Sheet 3

Himanshi Ghai (2019CS50433)
Mallika Prabhakar (2019CS50440)
Sayam Sethi (2019CS10399)

January 2022

Contents

1 Question 1

Question 1

Question. *We say that a context-free grammar G is self-referential if for some non-terminal symbol X we have $X \longrightarrow^* \alpha X \beta$, where $\alpha, \beta \neq \epsilon$. Show that a CFG that is not self-referential is regular.*

Proof. Since it is given that G is not self-referential, we can construct a DAG on G. For each production, $A \to \alpha_1 A_1 \alpha_2 A_2 \dots \alpha_n A_n \alpha_{n+1}$, where A_i are non-terminals and α_i are terminals, we add an edge $A \to A_i \forall i : 1 \leq i \leq n$. Since G is not self-referential, the graph generated will be a DAG (this can be easily proved using contradiction). We now propose the following algorithm to generate a NFA for $L(G)$:

Algorithm 1 Algorithm to generate NFA for L(G)

procedure NFA(G)
  $D \leftarrow graph(G)$
  $done \leftarrow$ boolean array initialised with 0 for each non-terminal
  while $\exists A : \neg done[A] \wedge (\forall B \in child[A] : done[B])$ **do** ▷ construct NFA for A using concatenation of regular languages
    $nfa(A) \leftarrow (Q_A, \Sigma_A, \delta_A, q_A, F_A)$
    $Q_A \leftarrow \{q_{\alpha_i} | 1 \leq i \leq n+1\} \cup (\bigcup_{i=1}^n Q_{A_i})$
    $\Sigma_A \leftarrow \{\alpha_i | 1 \leq i \leq n+1\} \cup (\bigcup_{i=1}^n \Sigma_{A_i})$
    $\delta_A \leftarrow \{\delta_A(q_{\alpha_i}, \alpha_i) = q_{A_i} | 1 \leq i \leq n\} \cup \{\delta_A(f_i, \alpha_{i+1}) = q_{\alpha_{i+1}} | 1 \leq i \leq n \wedge f_i \in F_{A_i}\} \cup (\bigcup_{i=1}^n \delta_{A_i})$
    $q_A \leftarrow q_{\alpha_1}$
    $F_A \leftarrow \{q_{\alpha_{n+1}}\}$
    $done[A] \leftarrow 1$
  end while
  return $nfa(S_0)$ ▷ S_0 is the start symbol of G
end procedure

In the above algorithm, we construct NFAs sequentially starting from those non-terminals which directly yield a terminal and have no outgoing edges. We then move up the graph to non-terminals that produce other non-terminals whose NFAs have already been constructed. We construct the NFA using concatenation of the NFAs for the non-terminals and terminals on the right hand side. Finally, we return the NFA produced for the start symbol. $\square$

2 Question 2

Question 2

Question. *Prove that the class of context-free languages is closed under intersection with regular languages. That is, prove that if L_1 is a context-free language and L_2 is a regular language, then $L_1 \cap L_2$ is a context-free language. Do this by starting with a DFA*

Solution. Let there be a PDA $P = (Q_P, \Sigma_P, \tau_P, \delta_P, q_{\circ_P}, F_P)$ for context free language L_1 and a DFA $D = (Q_D, \Sigma_D, \delta_D, q_{\circ_D}, F_D)$ for regular language L_2. To prove that $L_1 \cap L_2$ is a context-free language we need to show that there exists a PDA which accepts this language. Now consider the PDA $P' = (Q', \Sigma', \tau', \delta', q'_\circ, F')$ defined below.

1. $Q' = Q_P \times Q_D$
2. $\Sigma' = \Sigma_P \cup \Sigma_D$
3. $\tau' = \tau_P$
4. $\delta'((x, y), a, b) = \{(x', y')$ where $x' \in \delta_P(x, a, b)$ and $y' = \delta_D(y, b)\}$.
   Here $a \in \tau'$(stack top element), $b \in \Sigma'$(input element) and $(x, y) \in Q'$.
5. $q'_\circ = (q_{\circ_P}, q_{\circ_D})$
6. $F' = F_P \times F_D$

Here states in P' are pair of states where first state is from P and second state if from D. P' begins with empty stack from state $q'_\circ$ which is pair of start states of P and D. On any state (x, y) in P' on reading next symbol b from input string where $b \in \Sigma'$ it will transition into a state which is pair of states where P and D would have transition into on reading input symbol b and stack top element b.
Notice that here b can be ϵ i.e a transition in which the next symbol is not read. In this case D would not make any transition and will stay in same state whereas state and stack of P may be modified.

We claim that language accepted by P' is $L_1 \cap L_2$.

Proof. Let s be a string accepted by P'. Look at the derivation of s in PDA in P' and let say that

$$\{(q_{\circ_P}, q_{\circ_D}), (u_1, v_1), (u_2, v_2).......(u_k, v_k), (q_{F_P}, q_{F_D})\}$$

be the transition states of the derivation. It is similar to saying that P and D are transitioning on input string s in parallel. That is P has derivation

$$\{q_{\circ_P}, u_1, u_2, ..., u_k, q_{F_P}\}$$

and D has a derivation

$$\{q_{\circ_D}, v_1, v_2, ...v_k, q_{F_D}\}$$

for input string s.
From the construction we know that all the transitions of first state if defined by transition

function of P and that of second state by transition function of D and transition of second state if independent of stack of P. So the derivations above are valid transitions in P and D for input string s and are in final states after consuming complete input. So we can say that the string s is accepted by both P and D.
Notice here that it is not possible that s is accepted by only one of P and D. This is because the final states say (x, y) in F' of P' is pair final states of P and D.

Thus we can say that language accepted by P' is $L_1 \cap L_2$. $\square$

$\square$

3 Question 3

Question 3

Question. *Given two languages L, L', denote by*

$$L||L' := \{x_1y_1x_2y_2 \dots x_ny_n \mid x_1x_2 \dots x_n \in L, y_1y_2 \dots y_n \in L'\}$$

.

Show that if L is a CFL and L' is regular, then $L||L'$ is a CFL by constructing a PDA for $L||L'$. Is $L||L'$ a CFL if both L and L' are CFLs? Justify your answer.

Solution. We will construct a PDA for the language $L||L'$, assuming that the PDA for L is $P = (Q_L, \Sigma_L, \Gamma, \delta_L, (q_0)_L, \perp, F_L)$ and the NFA for L' is $D = (Q_{L'}, \Sigma_{L'}, \delta_{L'}, (q_0)_{L'}, F_{L'})$, as $S = (Q_L \times Q_{L'} \times \{1, 2\}, \Sigma_L \cup \Sigma_{L'}, \delta, ((q_0)_L, (q_0)_{L'}, 1), F)$ as follows:

$$\begin{aligned} F &= \{(f_1, f_2, 2) | f_1 \in F_L \text{ and } f_2 \in F_{L'}\} \\ \delta((q_1, q_2, 1), a, A) &= \{((q_1', q_2, 2), A') | (q_1', A') \in \delta_L(q_1, a)\} \\ \delta((q_1, q_2, 2), a, A) &= \{((q_1, q_2', 1), A) | q_2' \in \delta(q_2, a)\} \end{aligned} \tag{1}$$

We construct a PDA that alternates the simulation on the PDA and on the DFA. Therefore, the final accepting states are those where we reach the final state on both machines and we have just moving on the DFA (since it ends with y_n).
For the second part of the question, we show with a counter example that if both L and L' are CFLs then $L||L'$ is not a CFL. Consider the following languages:

$$\begin{aligned} L &= 0^n1^{2n} \\ L' &= 0^{2n}1^n \\ \implies L||L' &= 0^{2n}(10)^n1^{2n} \end{aligned} \tag{2}$$

It is easy to see that both L and L' are CFG's. We just modify the PDA for 0^n1^n by inserting new states to recognize two consecutive 1 for L and two consecutive 0 for L'. However, the language that we get as $L||L'$ is not a CFL. We can easily show this using Pumping Lemma. Let the pumping length be n. Now consider the string $z = 0^{2n}(10)^n1^{2n}$. For all possible breakups of z as $uvwxy$ where $|vwx| \le n$ and $vx \ne \epsilon$. We have the following cases:

1. **Case 1:** ($vwx = 0^n$) any pumping will lead to a word which will not be in $L||L'$
2. **Case 2:** ($vwx = 0^a(10)^{\frac{n-a}{2}}$) for any breakup of this string, we will have more 0 than 1 in the string. Therefore, no pumped string will be in $L||L'$
3. **Case 3:** ($vwx = (10)^{\frac{n-a}{2}}1^a$) for any breakup of this string, we will have more 1 than 0 in the string. Therefore, no pumped string will be in $L||L'$

Therefore, we have shown that the language produced by $L||L'$ when both L and L' are CFLs is not a CFL. Therefore CFLs are not closed under this operation. □

4 Question 4

Question 4

Question. *For* $A \subseteq \Sigma^*$, *define*

$$cycle(A) = \{yx \mid xy \in A\}$$

For example if $A = \{aaabc\}$, *then*

$$cycle(A) = \{aaabc, aabca, abcaa, bcaaa, caaab\}$$

. *Show that if* A *is a CFL then so is* $cycle(A)$

Solution. □

5 Question 5

Question 5

Question. *Let*

$$A = \{wtw^R \mid w, t, \in \{0,1\}^* \quad and \quad |w| = |t|\}$$

. Show that A is not a CFL.

Solution. Let's assume that A is a context-free language. Let p be the pumping length and take $s = 0^p(01)^{p/2}0^p$ which belongs to A and has length $> p$. Here $w = 0^p$ and $t = (01)^{p/2}$. From pumping lemma we can say that s can be pumped.
Let's divide the string into $uvxyz$ such that $|vy| > 0$ i.e. either v or y in non empty and $|vxy| \le p$. This is to satisfy the conditions of pumping lemma. Now according to pumping lemma for there will exist such $uvxyz$ satisfying above conditions and also $\forall i \ge 0$ $uv^ixy^iz \in A$.

Consider the following cases of dividing the string s into $uvxyz$.

Case 1: Both $v, y \in t$. Let $v = (01)^a$, $x = (01)^b$, and $y = (01)^c$ such that $a + b + c \le p/2$. WLOG let $u = 0^p(01)^{(p/2-a-b-c)}$ and $z = 0^p$.
For $i = 2$, we have pumped string $uv^2xy^2z = 0^p(01)^{p/2+a+c}0^p$. We can here see that only up to length p prefix is reverse of suffix. And length of pumped string is $> 3p$. As it is not of the form wtw^R where $|w| = |t|$, so $uv^2xy^2z \notin A$. This is a contradiction to our assumption that A is CFL and holds the pumping lemma.

Case 2: $v \in w$ and $y \in t$. Let $v = 0^a$, $x = 0^b(01)^c$ and $y = (01)^d$ such that $a+b+2c+2d < p$. Then $u = 0^{p-a-b}$ and $z = (01)^e0^p$ such that $c + d + e = p/2$.
For $i = 2$, we have pumped string $uv^2xy^2z = 0^{p+a}(01)^{c+2d+e}0^p$. Clearly here also up to length p only prefix is reverse of suffix and length of pumped string is $> 3p$. So $uv^2xy^2z \notin A$ which is a contradiction.

Case 3: $v \in t$ and $y \in z$. This case is similar to **Case 2** and here also result in contradiction.

There cannot be any other case as then the length $|vxy|$ would be greater than p.

As in every case of splitting s into $uvxyz$ has resulted into violation of pumping lemma, thus our assumption that A is a CFL is wrong.

Hence A is not a CFL. □

6 Question 6

Question 6

Question. *Prove the following stronger version of pumping lemma for CFLs: If A is a CFL, then there is a number k where if s is any string in A of length at least k then s may be divided into five pieces $s = uvxyz$, satisfying the conditions:*

- *for each $i \geq 0$, $uv^ixy^iz \in A$*
- *$v \neq \varepsilon$, and $y \neq \varepsilon$, and*
- *$|vxy| \leq k$.*

Solution. Found the trick to finish the proof. We just need to increasing the size of the pump. Let v be the number of variables in a Chompsky normal form grammar. If 2^v is the ordinary pumping length. Consider a new length 2^{3v}. In the parse tree of a long string (of minimal length) of length greater than 2^{3v}, there must be a path of height 3v. Thus some variable must occur 3 times. Then use the minimality of the parse tree to conclude that the stepwise retiteration can be pumped. □

7 Question 7

Question 7

Question. *Give an example of a language that is not a CFL but nevertheless acts like a CFL in the pumping lemma for CFL (Recall we saw such an example in class while studying pumping lemma for regular languages).*

Solution. □