CS 303 Introduction to Theory of Computing, VCU
Fall 2018, Assignment 4

Turned in electronically in PDF, PNG or Word format

Total marks: 63 marks + 3 bonus marks for LaTeX.

Unless otherwise noted, the alphabet for all questions below is assumed to be $\Sigma = \{0, 1\}$.

1. [12 marks] This question develops a basic understanding of CFGs and parse trees. Consider the grammar G below.

$$A \rightarrow A + B \mid B \quad (1)$$
$$B \rightarrow B \times C \mid C \quad (2)$$
$$C \rightarrow (A) \mid 5 \quad (3)$$

For each string below, give a parse tree of a derivation in G.

(a) 5

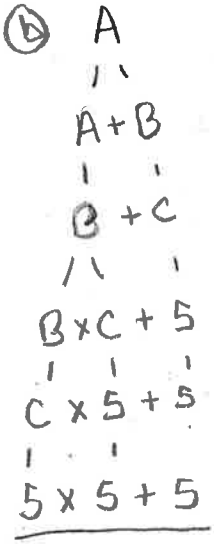
(b) $5 \times 5 + 5$

(c) $((5 \times 5) + (5 \times 5))$

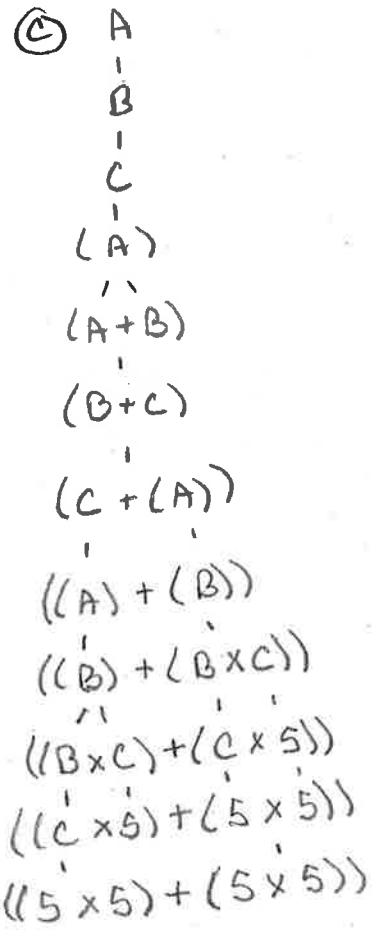
Solution:

(a)

(b)



(c)



2. [5 marks] We have seen in class that the sets of both regular and context-free languages are closed under the union, concatenation, and star operations. We have also seen in A2 that the regular languages are closed under complement. In this question, you will investigate whether context-free languages are closed under intersection. Use the languages $A = \{a^m b^n c^n \mid m, n \geq 0\}$ and $B = \{a^n b^n c^m \mid m, n \geq 0\}$ to show that the class of context-free languages is not closed under intersection. You may use the fact that the language $C = \{a^n b^n c^n \mid n \geq 0\}$ is not context-free.

Solution: Assume, for sake of contradiction, that the context-free languages are closed under intersection. Then if A and B are context-free, so is $A \cap B = C$. But we know C is not regular. Hence, if we can show that A and B are indeed context-free, we have a contradiction. To show that A and B are context-free, we demonstrate the following CFGs for A and B, respectively:

$$(A) \quad S \rightarrow MN \tag{4}$$
$$M \rightarrow aM \mid \epsilon \tag{5}$$
$$N \rightarrow bNc \mid \epsilon \tag{6}$$

$$(B) \quad S \rightarrow MN \tag{7}$$
$$M \rightarrow aMb \mid \epsilon \tag{8}$$
$$N \rightarrow cN \mid \epsilon \tag{9}$$

3. [12 marks] This question develops your ability to design CFGs. For each of the following languages, give a CFG. Assume the alphabet is $\Sigma = \{0, 1\}$. Justify your answers briefly.

   (a) $\{x \mid x \text{ is a palindrome}\}$. Recall a palindrome is a string that looks the same forwards and backwards. Examples of palindromes are "madam" and "racecar".
   Solution:

$$S \rightarrow 0S0 \mid 1S1 \mid T \mid \epsilon \tag{10}$$
$$T \rightarrow 0 \mid 1 \tag{11}$$

   (b) $\{x \mid \text{the length of x is odd}\}$.
   Solution:

$$S \rightarrow 0 \mid 1 \mid 0T \mid 1T \tag{12}$$
$$T \rightarrow 00T \mid 01T \mid 10T \mid 11T \mid \epsilon \tag{13}$$

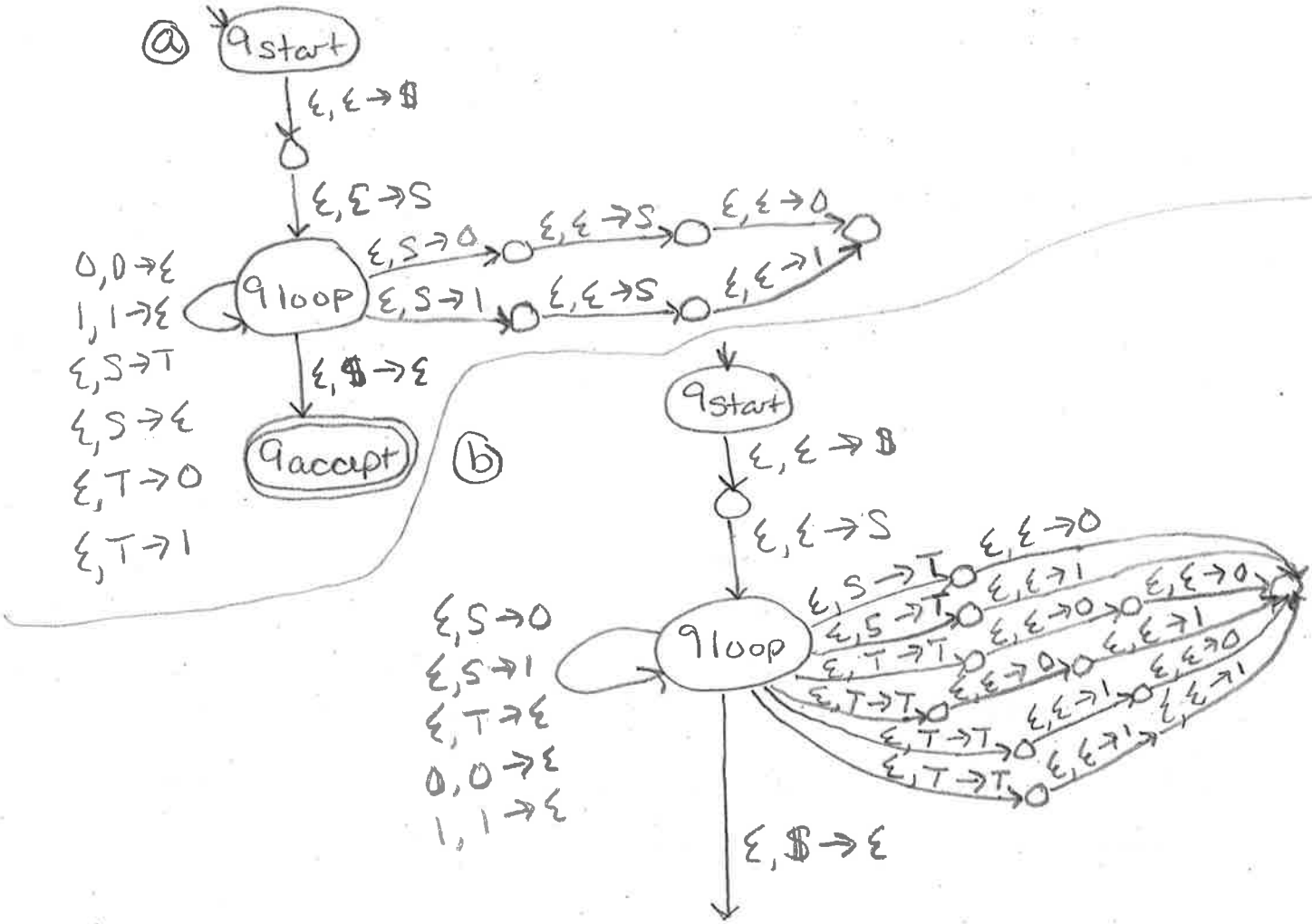
   (c) $\emptyset$.
   Solution: To generate the empty set, we have a grammar with no rules. Hence, no strings can be generated. (One can still define variables, though.)

   (d) $\{x \mid x \text{ does not contain any substrings of 1's of odd length}\}$. For example, ϵ, 011, 111100011 are in the language, but 10011 is not.
   Solution:

$$S \rightarrow \epsilon \mid 0S \mid 11S \tag{14}$$

4. [12 marks] This question develops your ability to design PDAs. For parts (a), (b), (c) of question 3 above, give state diagrams of pushdown automata. For each automata, include a brief description of the idea behind its design.

Solution:

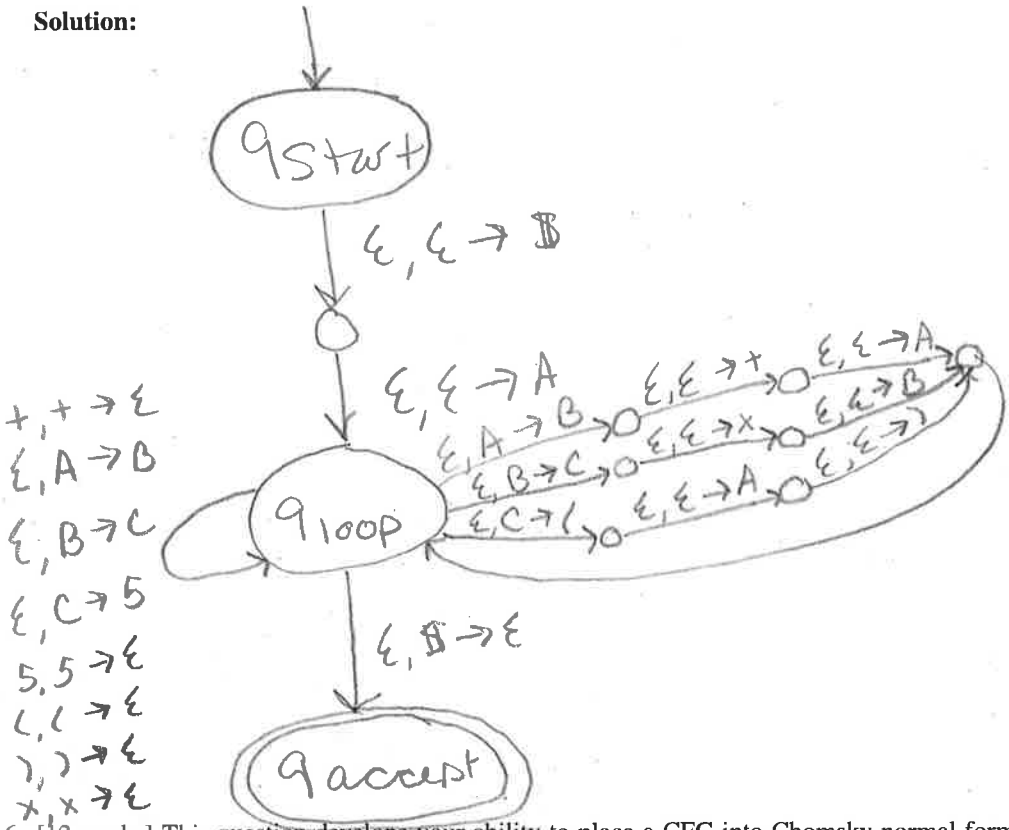


(c)

or just

5. [10 marks] This question forces you to practice the generic construction for mapping a CFG to a PDA. Specifically, for the grammar G from question 1, use the construction from Theorem 2.20 to construct an equivalent PDA P.

Solution:



6. [12 marks] This question develops your ability to place a CFG into Chomsky normal form and to understand properties that result from that form.

   (a) Convert G so that it is a CFG in Chomsky normal form. Where each rule is in the form: $A \to BC$ or $A \to a$

   Solution:

$A_0 \to UB \mid YC \mid ZR \mid 5$

$A \to UB \mid YC \mid ZR \mid 5$

$B \to YC \mid ZR \mid 5$

$U \to AP$

$P \to +$

$Y \to BT$

$T \to X$

$Z \to QA$

$Q \to ($

$R \to)$

(b) Show that if G is a CFG in Chomsky normal form, then for any string $w \in L(G)$ of length $n \geq 1$, exactly $2n - 1$ steps are required for any derivation of w.

Solution: To begin, since $n \geq 1$, we can assume WLOG that we do not have a rule of the form $S \to \epsilon$, for S the start variable. This means no variables can map to ϵ. In other words, since G is in Chomsky normal form, whatever intermediate string w' of variables and terminals we currently have, applying a substitution rule can do one of two things: (1) Leave the length of w' invariant, in which case a variable in w' must have been replaced with a terminal (rule of type $A \to a$), and (2) increase the length of w' by one (rule of type $A \to BC$). It follows that the only way we to achieve a final string length of n for w is to use $n - 1$ rules of type $A \to BC$ (which grow our string from size 1 to n) and n rules of type $A \to a$ (which replace all n variables with n terminals). The order in which these rules are applied naturally depends on which string w we are deriving, and is unimportant for this proof. We conclude that a total of exactly $2n - 1$ steps are required to derive a string of length n.