CS5000 HW4

Chase Mortensen A01535275

September 2019

1

Let L be a language over $\{a, b\}$ generated by the following grammar G:

1. $S \to aSa$
2. $S \to \epsilon$

Answer the following questions:

1. Is G linear? Explain why or why not.
   G is not linear. If G were right linear, it would be of the form $S \to aaS$. If it were left linear, it would be $S \to Saa$. Since we are inserting between two characters and not at either the left or right end, G is not linear.
2. Is L linear? Explain why or why not.
   L is linear. The language generated by G results in strings such as ϵ, aa, $aaaa$, $aaaaaa$, etc. Although G is not linear, the language it generates is identical to languages generated by other grammars such as $S \to aaS$ and $S \to Saa$. Since L can be generated by a linear grammar, L is linear.
3. Are all languages generated by non-linear grammars necessarily non-linear?
   Not necessarily. As shown above, a non-linear grammar can generate linear languages because the language generated by the non-linear grammar is identical to a language generated by a linear grammar.

2

Let L_1 and L_2 be languages over some alphabet Σ. The quotient of L_1 and L_2, $\frac{L_1}{L_2}$, is defined as $\{x | \exists y \in L_2 \text{ such that } xy \in L_1\}$. Let L_1 be a regular language over Σ and L_2 a finite language over Σ. Show that $\frac{L_1}{L_2}$ is regular.

$\frac{L_1}{L_2}$ is regular. L_1 is a regular language and L_2 is a finite language. Thinking about $\frac{L_1}{L_2}$ as a prefix in L_1 (since L_1 is xy), y is a postfix in L_1 that happens to be in L_2. $L_2(y)$ is also a finite language and therefore also regular. Removing

a regular postfix from a regular language results in a regular language. Our class notes state "A language L is regular if and only if there exists a DFA M such that $L(M) = L$." If There is a DFA for L_1 which consists of a subsection x connected to a subsection y, removing the y results in a DFA, which has a corresponding regular language.

3

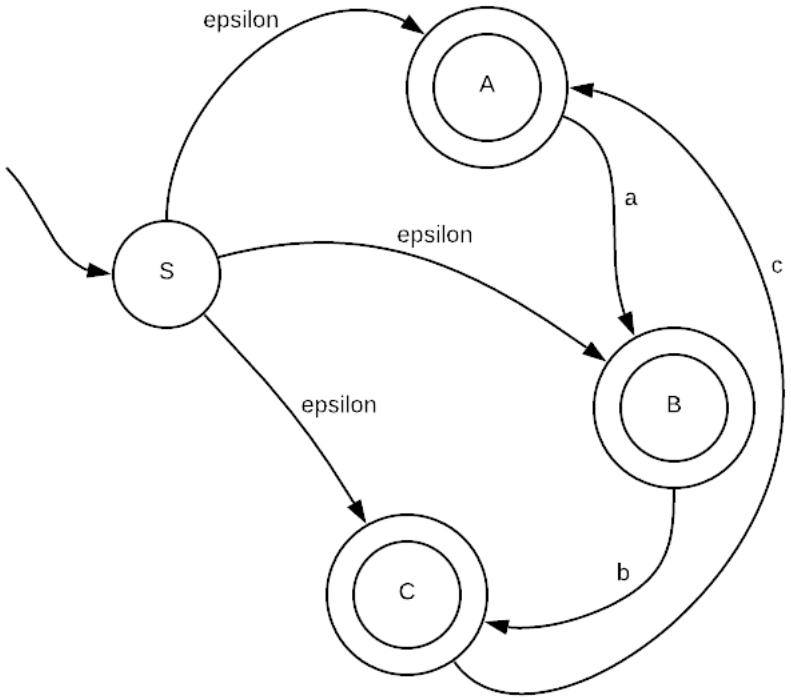
Construct a finite state machine for the following grammar:

1. $S \rightarrow A|B|C$;
2. $A \rightarrow aB|\epsilon$;
3. $B \rightarrow bC|\epsilon$;
4. $C \rightarrow cA|\epsilon$.

The transitions in the grammar above are represented in the finite state machine transition table below.

	S	A	B	C
a	$\emptyset$	$\{B\}$	$\emptyset$	$\emptyset$
b	$\emptyset$	$\emptyset$	$\{C\}$	$\emptyset$
c	$\emptyset$	$\emptyset$	$\emptyset$	$\{A\}$
ϵ	$\{A, B, C\}$	$\emptyset$	$\emptyset$	$\emptyset$

A diagram of the complete FSM is shown below. The epsilon transitions in the given grammar are translated to accepting states in the FSM.



4

Write a context-free grammar G that generates all strings of balanced left and right parentheses, e.g., (), (()), ()(), (()()),((()(()))), etc.

G is a CFG with equal numbers of left and right parentheses. It is not necessarily a palindrome, but for each production that contains a left parentheses, it must also contain a right parentheses and vice versa.

1. $S \rightarrow (S)$;
2. $S \rightarrow)S($;
3. $S \rightarrow ()S$;
4. $S \rightarrow)(S$;
5. $S \rightarrow S()$;
6. $S \rightarrow S)($;

7. $S \to \epsilon$

This grammar G may contain extra productions which aren't necessary, but it isn't required to have a minimum number of productions or to be right linear or left linear. Each of the example strings can be created using a combination of the productions listed above.

5

Recall that a palindrome is a string that reads the same left to right and right to left, e.g., *aba*, *abba*, *abaaba*, etc. Write a context-free grammar G for the language of palindromes over $\{a, b\}$.

The language above contains two of the productions necessary in this grammar. However, this grammar also requires productions of single characters since there isn't always an even number of each character (such as *aba*).

1. $S \to aSa$;
2. $S \to bSb$;
3. $S \to aTa$;
4. $S \to bTb$;
5. $S \to T$
6. $T \to aU$;
7. $T \to bU$;
8. $T \to U$;
9. $U \to \epsilon$.

Again, this grammar may not be written perfectly concisely, but it does handle the requirement of creating a palindrome over $\{a, b\}$.