CSC 512	Compiler Construction	
Fall 2022	Xu Liu	HW 2

1

The following grammar is not suitable for a top-down predictive parser. Identify the problem and correct it by rewriting the grammar. Show that your new grammar satisfies the $LL(1)$ condition.

$$L \to Ra|Qba$$

$$R \to aba|caba|Rbc$$

$$Q \to bbc|bc$$

There is left recursion in the nonterminal R. We can correct it as follows:

$$L \to Ra|Qba$$

$$R \to abaT|cabaT$$

$$T \to bcT|\epsilon$$

$$Q \to bZ$$

$$Z \to bc|c$$

Now the solution has no left recursion. We must also formally prove that the grammar satisfies the $LL(1)$ grammar s.t. $A \to \alpha$ and $A \to \beta$ implies $FIRST^{+}(\alpha) \cap FIRST^{+}(\beta) = \varnothing$:

For L, R, T, Q, Z:

$$FIRST^{+}(Ra) \cap FIRST^{+}(Qba) = \varnothing$$

$$FIRST^{+}(abaT) \cap FIRST^{+}(cabaT) = \varnothing$$

$$FIRST^{+}(bcT) \cap FIRST^{+}(\epsilon) = \varnothing$$

$$FIRST^{+}(bZ) = \varnothing$$

$$FIRST^{+}(bc) \cap FIRST^{+}(c) = \varnothing$$

Simplified:

$$a, c \cap b = \varnothing$$

$$a \cap c = \varnothing$$

$$b \cap \epsilon = \varnothing$$

$$b \cap c = \varnothing$$

2

Consider the following grammar:

$$A \to Ba$$
$$B \to dab|Cb$$
$$C \to cB$$
$$C \to Ac$$

(a) Does this grammar has left recursions? If you believe so, try to rewrite the grammar to remove left recursions; the new grammar should describe the same set of expressions as the original grammar does.

Yes, there is left recursion. From $A \to Ba, B \to Cb, C \to Ac$

We can fix this grammar:

$$A \to dabaD|cBbaD$$
$$B \to dab|Cb$$
$$C \to cB|Ac$$
$$D \to cbaD|\epsilon$$

(b) Is the original grammar a $LL(1)$ grammar? Justify your answer.

The original grammar has a left recursion, thus it doesn't satisfy $LL(1)$. Even after eliminating left recursion in a, the grammar still doesn't satisfy $LL(1)$ due to disjoint sets.

For A, B, C, D:

$$FIRST^{+}(dabaD) \cap FIRST^{+}(cBbaD) = \varnothing$$
$$FIRST^{+}(dab) \cap FIRST^{+}(Cb) = \varnothing$$
$$FIRST^{+}(cB) \cap FIRST^{+}(Ac) = \varnothing$$
$$FIRST^{+}(cbaD) \cap FIRST^{+}(\epsilon) = \varnothing$$

Simplified:

$$d \cap c = \varnothing$$
$$d \cap c, d \neq \varnothing$$
$$c \cap c, d \neq \varnothing$$
$$c \cap \epsilon = \varnothing$$

3

Write a grammar to describe all binary numbers that are multiples of four.

Note: 0 is a multiple of 4. The binary number is unsigned (i.e., it contains no sign bit). It should not contain leading 0s. For instance 00100 is not legal, but 100 is.

$$N \to 0|1M$$

$$M \to 1M|0M|00$$

Multiples of 4 in binary are represented with trailing two zeros, `00`. The grammar becomes ambiguous, so it will have to look ahead