Exercise set 4 (Context-free languages)

1. Write CFG's for the following languages over $\Sigma := \{a, b\}$:

(a) $\{\alpha\alpha^R \mid \alpha \in \Sigma^*\}$.

(b) $\{a^k b^k \mid k \in \mathbb{Z}_+\}$.

(c) $\{a^k b^l a^k \mid k, l \in \mathbb{Z}_+\}$.

(d) $\{\alpha \in \Sigma^* \mid \alpha$ has an equal number of a's and b's$\}$.

(e) $\{\alpha \in \Sigma^* \mid \alpha$ has twice as many a's as b's$\}$.

(f) $\{a^k b^l \mid k, l \in \mathbb{Z}_+$ and $k \leqslant 2l\}$.

2. Let $\Sigma := \{a, b, \emptyset, \varepsilon, (,), \cup, ^*\}$. Define a CFG $G = (\Sigma, V, S, R)$ such that $\mathcal{L}(G)$ is the set of all regular expressions over $\{a, b\}$. (**Remark:** In order to avoid confusion we use ε to denote the epsilon of RE's and ϵ to denote the the null string of Σ^*.)

3. Write a CFG to accept the language of the regular expression $a(a \cup b)^*b$ over the alphabet $\Sigma := \{a, b\}$.

4. A CFG is said to be right linear (or regular), if every production of the grammar is of the form $A \to \alpha B$ or $A \to \alpha$, where A and B are non-terminals and α is a string (of length zero or more) of terminals.

* **(a)** Let $L := \mathcal{L}(G)$, where G is right-linear CFG. Show that L is regular. (**Hint:** Construct an ϵ-NFA to accept L.)

(b) Let L be a regular language. Show that there exists a right-linear CFG G with $\mathcal{L}(G) = L$. (**Hint:** Consider a DFA D with $\mathcal{L}(D) = L$.)

5. Let $G = (\Sigma, V, S, R)$ be a CFG with $L := \mathcal{L}(G)$. For $n \in \mathbb{N}$ define L_n to be the set of all strings α in L such that there is a derivation $S \Rightarrow^* \alpha$ with n or fewer steps. Show that L_n is finite for every $n \in \mathbb{N}$.

6. Let G be a CFG and let $\alpha \in \mathcal{L}(G)$ with $|\alpha| = l$ have a derivation under G using n steps.

(a) If there is no production in G of the form $A \to \epsilon$ (for some non-terminal A), show that every parse tree for α has exactly $l + n$ nodes.

(b) Now assume that the productions of G are arbitrary, i.e., those of the form $A \to \epsilon$ are also allowed. Show that every parse tree for α has $\leqslant l + 2n$ nodes.

7. Let L be the language of balanced parentheses. Denote $\Sigma := \{(,)\}$.

(a) Show that the CFG $G := \{\Sigma, \{S\}, S, \{S \to \epsilon \mid SS \mid (S)\}\}$ for L is ambiguous.

(b) Show that the CFG $G' := \{\Sigma, \{S, T\}, S, \{S \to \epsilon \mid T, T \to () \mid (T) \mid TT\}\}$ for L is again ambiguous.

* **(c)** Design an unambiguous CFG G'' with $\mathcal{L}(G'') = L$.

* **8.** Conventionally arithmetic expressions are written in the infix notation in which a binary operator (like $+$ and $*$) is sandwiched between its two operands. In the postfix notation one writes the operands first and then the operator. Thus $a\ b\ +$ in postfix notation stands for $a + b$ in the usual infix notation.

(a) Convert the arithmetic expression $a + (b * (a + b * a) + a * a) * (a + b)$ from the infix notation to the postfix notation.

(b) Convert the arithmetic expression $b\ a + a\ b * a\ a\ b + b * * + b + *$ from the postfix notation to the infix notation. (**Hint:** Use a stack.)

(c) Show that the CFG $(\{a, b, +, *\}, \{P\}, P, R)$ generates all and only arithmetic expressions involving $a, b, +, *$ in the postfix notation, where R consists of the derivations $P \to a \mid b \mid P\ P + \mid P\ P *$.

(d) Prove that the CFG of Part (c) is unambiguous (in spite of the fact that we have not used parentheses).

Remark: Oops! We then need no brackets, no rules of precedence and still can generate unambiguous arithmetic expressions in the postfix notations. True! But to human readers arithmetic expressions in the postfix notation are not as lucid as those in the common-place infix notation. Perhaps you have realized this plane fact quite well, while solving Parts (a) and (b). Incidentally, the printer language PostScript uses the postfix notation for its subroutines, i.e., the arguments are supplied first followed by the subroutine name. The following PostScript code implements the arithmetic expression of Part (b):

```
b a add a b mul a a b add b mul mul add b add mul
```

The following code snippet performs this operation on $a := 97$ and $b := 23$ and displays the result on a page. It's just a one-page printer file. Check that the computation is correct. Also observe how every subroutine call is done in postfix convention.

```
%!PS-Adobe-2.0
/a 97 def                                              %Define a:=97
/b 23 def                                              %Define b:=23
b a add a b mul a a b add b mul mul add b add mul      %Computation
/t exch def                                            %Save result in t
/Helvetica-Bold findfont 12 scalefont setfont          %Set display font
200 200 moveto                                         %Set display location
t 100 string cvs                                       %Convert t to a string
show                                                   %Display
showpage                                               %End of page
```

(e) Analogously define prefix notation, write a grammar for arithmetic expressions in prefix notation and show that this grammar is unambiguous too (without brackets and rules of precedence).

9. Consider the following languages over $\Sigma := \{a, b, c\}$:

$$\begin{aligned} L_1 &:= \{a^k b^k c^l \mid k, l \in \mathbb{N}\} \\ L_2 &:= \{a^k b^l c^l \mid k, l \in \mathbb{N}\} \\ L &:= L_1 \cup L_2 \end{aligned}$$

(a) Prove that L_1, L_2 and L are context-free.

* **(b)** Argue that L is inherently ambiguous, i.e., any CFG for L has to be ambiguous. (**Remark:** Note that a formal proof of this statement is very difficult. Give some heuristic argument instead.)

10. Which of the following languages are context-free?

(a) $\{a^k b^l \mid k \neq l\}$.

(b) $\{\alpha \in \{a, b\}^* \mid \alpha$ has an unequal number of a's and b's$\}$.

(c) $\{ababbabbba \ldots ab^{k-1}ab^k a \mid k \in \mathbb{N}\}$.

(d) $L_1 := \{a^k b^{2k} c^l \mid k, l \in \mathbb{Z}_+\}$.

(e) $L_2 := \{a^k b^l c^{2l} \mid k, l \in \mathbb{Z}_+\}$.

(f) $L_1 \cap L_2$.

(g) $\{\alpha \in \Sigma^* \mid |\alpha|$ is a perfect square$\}$.

(h) $\{\alpha \in \Sigma^* \mid |\alpha|$ is not a perfect square$\}$.

(i) L^R, where L is a CFL.