Quiz Section 8: CFGs, Regex, Recursive definitions, Structural Induction

Task 1 – Seeing double

Consider the following recursive definition of strings.

Basis Step: $\varepsilon \in \Sigma^*$

Recursive Step: for any $a \in \Sigma$, if $w \in \Sigma^*$, then $wa \in \Sigma^*$.

Recall the following recursive definition of the function len:

$$\begin{aligned} \text{len}(\varepsilon) &:= 0 \\ \text{len}(wa) &:= \text{len}(w) + 1 \text{ for } x \in \Sigma^*, a \in \Sigma \end{aligned}$$

The double operator is defined recursively as follows:

$$\begin{aligned} \text{double}(\texttt{""}) &:= \texttt{""} \\ \text{double}(wa) &:= wwa \end{aligned}$$

Let Σ be a fixed alphabet. Prove that $\forall w \in \Sigma^*$, $\text{len}(\text{double}(w)) = 2\text{len}(w)$ by induction.

Task 2 – Leafy Trees

Consider the following definition of a (binary) **Tree**:

Basis Step: $\bullet$ is a **Tree**.

Recursive Step: If L is a **Tree** and R is a **Tree** then $\texttt{Tree}(\bullet, L, R)$ is a **Tree**.

The function leaves returns the number of leaves of a **Tree**. It is defined as follows:

$$\begin{aligned} \text{leaves}(\bullet) &= 1 \\ \text{leaves}(\texttt{Tree}(\bullet, L, R)) &= \text{leaves}(L) + \text{leaves}(R) \end{aligned}$$

Also, recall the definition of size on trees:

$$\begin{aligned} \text{size}(\bullet) &= 1 \\ \text{size}(\texttt{Tree}(\bullet, L, R)) &= 1 + \text{size}(L) + \text{size}(R) \end{aligned}$$

Prove that $\text{leaves}(T) \geqslant \text{size}(T)/2 + 1/2$ for all Trees T.

Task 3 – Recursively Defined Sets of Strings

For each of the following, write a recursive definition of the sets satisfying the following properties. Briefly justify that your solution is correct.

a) Binary strings of even length.

b) Binary strings not containing 10.

c) Binary strings not containing 10 as a substring and having at least as many 1s as 0s.

d) Binary strings containing at most two 0s and at most two 1s.

Task 4 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

c) Write a regular expression that matches all binary strings that contain the substring "111", but not the substring "000".

Task 5 – CFGs

Give CFGs for each of the following languages.

"Document" all the non-start variables in your grammar with an English description of the set of strings it generates. (You do not need to document the start variable because it is documented by the problem statement.)

a) All binary strings that start with 11.

b) All binary strings that contain at most one 1.

c) All binary strings that end in 00.

d) All binary strings that contain at least three 1's.

e) All strings over $\{0,1,2\}$ with the same number of 1s and 0s and exactly one 2.

Hint: Try modifying the grammar from lecture for binary strings with the same number of 1s and 0s. (You may need to introduce new variables in the process.)