CSE 311: Foundations of Computing I

Section 7: Strong Induction and Recursive Sets

1. Cantelli's Rabbits

Xavier Cantelli owns some rabbits. The number of rabbits he has in a given year is described by the function f:

$$
\begin{aligned}
f(0) &= 0 \\
f(1) &= 1 \\
f(n) &= 2f(n-1) - f(n-2) \qquad \text{for } n \geq 2
\end{aligned}
$$

Determine, with proof, the number, $f(n)$, of rabbits that Cantelli owns in year n.

2. Structural Induction

(a) Consider the following recursive definition of strings Σ^* over the alphabet Σ.

Basis Step: ε is a string

Recursive Step: If w is a string and $a \in \Sigma$ is a character, then wa is a string.

Recall the following recursive definition of the function len:

$$
\begin{aligned}
\text{len}(\varepsilon) &= 0 \\
\text{len}(wa) &= 1 + \text{len}(w)
\end{aligned}
$$

Now, consider the following recursive definition:

$$
\begin{aligned}
\text{double}(\varepsilon) &= \varepsilon \\
\text{double}(wa) &= \text{double}(w)aa.
\end{aligned}
$$

Prove that for any string x, $\text{len}(\text{double}(x)) = 2\text{len}(x)$.

(b) 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 $\text{leaves}(T) \geq \text{size}(T)/2$ for all Trees T.

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 as a substring and having at least as many 1s as 0s.

4. Regular Expressions

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

(b) Write a regular expression that matches all non-negative 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".