CSE 311: Foundations of Computing I

Section : Structural Induction and Regular Expressions Solutions

0. Structural Induction

(a) Recall the recursive definition of a list:

$$\textbf{List} = \texttt{[]} \mid \text{Int} :: \text{List}$$

And the definition of len on lists:

$$\begin{aligned} \text{len}(\texttt{[]}) &= 0 \\ \text{len}(x :: L) &= 1 + \text{len}(L) \end{aligned}$$

Consider the following recursive definition:

$$\begin{aligned} \text{stutter}(\texttt{[]}) &= \texttt{[]} \\ \text{stutter}(x :: L) &= x :: x :: \text{stutter}(L) \end{aligned}$$

Prove that $\text{len}(\text{stutter}(L)) = 2\text{len}(L)$ for all Lists L.

Solution:

We go by structural induction. Let L be a list.

Case $L = \texttt{[]}$**.** Note that $\text{len}(\text{stutter}(\texttt{[]})) = \text{len}(\texttt{[]}) = 0 = 2\text{len}(\texttt{[]})$.

Case $L = x :: L'$**.** Suppose that $\text{len}(\text{stutter}(L')) = 2\text{len}(L')$ for some list L'.

Note that

$$\begin{aligned} \text{len}(\text{stutter}(x :: L')) &= \text{len}(x :: x :: \text{stutter}(L')) && \text{[By Definition of stutter]} \\ &= 1 + \text{len}(x :: \text{stutter}(L')) && \text{[By Definition of len]} \\ &= 1 + 1 + \text{len}(\text{stutter}(L')) && \text{[By Definition of len]} \\ &= 2 + 2\text{len}(L') && \text{[By IH]} \\ &= 2(1 + \text{len}(L')) && \text{[Algebra]} \\ &= 2(\text{len}(x :: L')) && \text{[By Definition of len]} \end{aligned}$$

Thus, the claim is true for all Lists by structural induction.

(b) Consider the recursive definition of a tree:

$$\textbf{Tree} = \texttt{Nil} \mid \texttt{Tree}(\textbf{Integer}, \textbf{Tree}, \textbf{Tree})$$

And the definition of size on trees:

$$\begin{aligned} \text{size}(\texttt{Nil}) &= 0 \\ \text{size}(\texttt{Tree}(x, L, R)) &= 1 + \text{size}(L) + \text{size}(R) \end{aligned}$$

And the definition of height on trees:

$$\begin{aligned} \text{height}(\texttt{Nil}) &= 0 \\ \text{height}(\texttt{Tree}(x, L, R)) &= 1 + \max(\text{height}(L), \text{height}(R)) \end{aligned}$$

Prove that $\text{size}(T) \leq 2^{\text{height}(T)+1} - 1$ for all Trees T.

Solution:

We go by structural induction. Let T be a tree.

Case $T = \texttt{Nil}$. Note that size$(\texttt{Nil}) = 0 \leq 1 = 2^{0+1} - 1 = 2^{\text{height}(\texttt{Nil})+1} - 1$.

Case $T = \texttt{Tree}(x, L, R)$. Suppose that size$(L) \leq 2^{\text{height}(L)+1} - 1$ and size$(R) \leq 2^{\text{height}(R)+1} - 1$ for some trees L and R.

Note that

$$\begin{aligned}
\text{size}(\texttt{Tree}(x, L, R)) &= 1 + \text{size}(L) + \text{size}(R) && \text{[By Definition of size]} \\
&\leq 1 + 2^{\text{height}(L)+1} - 1 + 2^{\text{height}(R)+1} - 1 && \text{[By IH]} \\
&\leq 1 + 2^{\max(\text{height}(L),\text{height}(R))+1} - 1 + 2^{\max(\text{height}(L),\text{height}(R))+1} - 1 && \text{[By } \max\text{]} \\
&\leq 2\left(2^{\text{height}(\texttt{Tree}(x,L,R))}\right) - 1 && \text{[Algebra]} \\
&\leq 2^{\text{height}(\texttt{Tree}(x,L,R))+1} - 1 && \text{[Algebra]}
\end{aligned}$$

Thus, the claim is true for all Trees by structural induction.

1. Regular Expressions

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

Solution:

$$0 \cup ((1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*)$$

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

Solution:

$$0 \cup ((1 \cup 2)(0 \cup 1 \cup 2)^* 0)$$

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

Solution:

$$(01 \cup 001 \cup 1^*)^*(0 \cup 00 \cup \varepsilon)111(01 \cup 001 \cup 1^*)^*(0 \cup 00 \cup \varepsilon)$$