Section 07: Solutions

1. Structural Induction

(a) Consider the following recursive definition of strings.

Basis Step: "" is a string

Recursive Step: If X is a string and c is a character then $\mathsf{append}(c, X)$ is a string.

Recall the following recursive definition of the function len:

$$\begin{aligned} \mathsf{len}(\texttt{""}) &= 0 \\ \mathsf{len}(\mathsf{append}(c, X)) &= 1 + \mathsf{len}(X) \end{aligned}$$

Now, consider the following recursive definition:

$$\begin{aligned} \mathsf{double}(\texttt{""}) &= \texttt{""} \\ \mathsf{double}(\mathsf{append}(c, X)) &= \mathsf{append}(c, \mathsf{append}(c, \mathsf{double}(X))). \end{aligned}$$

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

Solution:

For a string X, let $\mathsf{P}(X)$ be "$\mathsf{len}(\mathsf{double}(X)) = 2\mathsf{len}(X)$". We prove $\mathsf{P}(X)$ for all strings X by structural induction on X.

Base Case (X = ""): By definition, $\mathsf{len}(\mathsf{double}(\texttt{""})) = \mathsf{len}(\texttt{""}) = 0 = 2 \cdot 0 = 2\mathsf{len}(\texttt{""})$, so $\mathsf{P}(\texttt{""})$ holds

Inductive Hypothesis: Suppose $\mathsf{P}(X)$ holds for some arbitrary string X.

Inductive Step: Goal: Show that $\mathsf{P}(\mathsf{append}(c, X))$ holds for any character c.

$$\begin{aligned} \mathsf{len}(\mathsf{double}(\mathsf{append}(c, X))) &= \mathsf{len}(\mathsf{append}(c, \mathsf{append}(c, \mathsf{double}(X)))) && \text{[By Definition of double]} \\ &= 1 + \mathsf{len}(\mathsf{append}(c, \mathsf{double}(X))) && \text{[By Definition of len]} \\ &= 1 + 1 + \mathsf{len}(\mathsf{double}(X)) && \text{[By Definition of len]} \\ &= 2 + 2\mathsf{len}(X) && \text{[By IH]} \\ &= 2(1 + \mathsf{len}(X)) && \text{[Algebra]} \\ &= 2(\mathsf{len}(\mathsf{append}(c, X))) && \text{[By Definition of len]} \end{aligned}$$

This proves $\mathsf{P}(\mathsf{append}(c, X))$.

Conclusion: $\mathsf{P}(X)$ holds for all strings X by structural induction.

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

Basis Step: • 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:

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

Also, recall the definition of size on trees:

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

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

Solution:

For a tree T, let P be $\mathsf{leaves}(T) \geq \mathsf{size}(T)/2 + 1/2$. We prove P for all trees T by structural induction on T.

Base Case (T = •): By definition of $\mathsf{leaves}(\bullet)$, $\mathsf{leaves}(\bullet) = 1$ and $\mathsf{size}(\bullet) = 1$. So, $\mathsf{leaves}(\bullet) = 1 \geq 1/2 + 1/2 = \mathsf{size}(\bullet)/2 + 1/2$, so $\mathsf{P}(\bullet)$ holds.

Inductive Hypothesis: Suppose $\mathsf{P}(L)$ and $\mathsf{P}(R)$ hold for some arbitrary trees L, R.

Inductive Step: Goal: Show that $\mathsf{P}(\mathtt{Tree}(\bullet, L, R))$ holds.

$$\begin{aligned}
\mathsf{leaves}(\mathtt{Tree}(\bullet, L, R)) &= \mathsf{leaves}(L) + \mathsf{leaves}(R) && \text{[By Definition of } \mathsf{leaves}\text{]} \\
&\geq (\mathsf{size}(L)/2 + 1/2) + (\mathsf{size}(R)/2 + 1/2) && \text{[By IH]} \\
&= (1/2 + \mathsf{size}(L)/2 + \mathsf{size}(R)/2) + 1/2 && \text{[By Algebra]} \\
&= \frac{1 + \mathsf{size}(L) + \mathsf{size}(R)}{2} + 1/2 && \text{[By Algebra]} \\
&= \mathsf{size}(T)/2 + 1/2 && \text{[By Definition of } \mathsf{size}\text{]}
\end{aligned}$$

This proves $\mathsf{P}(\mathtt{Tree}(\bullet, L, R))$.

Conclusion: Thus, $\mathsf{P}(T)$ holds for all trees T by structural induction.

(c) Prove the previous claim using strong induction. Define $P(n)$ as "all trees T of size n satisfy $\mathsf{leaves}(T) \geq \mathsf{size}(T)/2 + 1/2$". You may use the following facts:

- For any tree T we have $\mathsf{size}(T) \geq 1$.
- For any tree T, $\mathsf{size}(T) = 1$ if and only if $T = \bullet$.

If we wanted to prove these claims, we could do so by structural induction.

Note, in the inductive step you should start by letting T be an arbitrary tree of size $k + 1$.

Solution:

Let $P(n)$ be "all trees T of size n satisfy $\mathsf{leaves}(T) \geq \mathsf{size}(T)/2 + 1/2$". We show $P(n)$ for all integers $n \geq 1$ by strong induction on n.

Base Case: Let T be an arbitrary tree of size 1. The only tree with size 1 is $\bullet$, so $T = \bullet$. By definition, $\mathsf{leaves}(T) = \mathsf{leaves}(\bullet) = 1$ and thus $\mathsf{size}(T) = 1 = 1/2 + 1/2 = \mathsf{size}(T)/2 + 1/2$. This shows the base case holds.

Inductive Hypothesis: Suppose that $P(j)$ holds for all integers $j = 1, 2, \ldots, k$ for some arbitrary integer $k \geq 1$.

Inductive Step: Let T be an arbitrary tree of size $k+1$. Since $k+1 > 1$, we must have $T \neq \bullet$. It follows from the definition of a tree that $T = \mathtt{Tree}(\bullet, L, R)$ for some trees L and R. By definition, we have $\mathsf{size}(T) = 1 + \mathsf{size}(L) + \mathsf{size}(R)$. Since sizes are non-negative, this equation shows $\mathsf{size}(T) > \mathsf{size}(L)$ and $\mathsf{size}(T) > \mathsf{size}(R)$ meaning we can apply the inductive hypothesis. This says that $\mathsf{leaves}(L) \geq \mathsf{size}(L)/2 + 1/2$ and $\mathsf{leaves}(R) \geq \mathsf{size}(R)/2 + 1/2$.

We have,

$$\begin{aligned}
\mathsf{leaves}(T) &= \mathsf{leaves}(\texttt{Tree}(\bullet, L, R)) \\
&= \mathsf{leaves}(L) + \mathsf{leaves}(R) && \text{[By Definition of leaves]} \\
&\geq (\mathsf{size}(L)/2 + 1/2) + (\mathsf{size}(R)/2 + 1/2) && \text{[By IH]} \\
&= (1/2 + \mathsf{size}(L)/2 + \mathsf{size}(R)/2) + 1/2 && \text{[By Algebra]} \\
&= \frac{1 + \mathsf{size}(L) + \mathsf{size}(R)}{2} + 1/2 && \text{[By Algebra]} \\
&= \mathsf{size}(T)/2 + 1/2 && \text{[By Definition of size]}
\end{aligned}$$

This shows $P(k+1)$.

Conclusion: $P(n)$ holds for all integers $n \geq 1$ by the principle of strong induction.

Note, this proves the claim for all trees because every tree T has some size $s \geq 1$. Then $P(s)$ says that all trees of size s satisfy the claim, including T.

2. 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:

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

3. CFGs

(a) All binary strings that end in 00.
Solution:

$$\mathbf{S} \to 0\mathbf{S} \mid 1\mathbf{S} \mid 00$$

(b) All binary strings that contain at least three 1's.
Solution:

$$\mathbf{S} \to \mathbf{TTT}$$
$$\mathbf{T} \to 0\mathbf{T} \mid \mathbf{T}0 \mid 1\mathbf{T} \mid 1$$

(c) 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.)
Solution:

We can do this by slightly modifying the grammar from lecture.

$$\mathbf{S} \to 2\mathbf{T} \mid \mathbf{T}2 \mid \mathbf{ST} \mid \mathbf{TS} \mid 0\mathbf{S}1 \mid 1\mathbf{S}0$$
$$\mathbf{T} \to \mathbf{TT} \mid 0\mathbf{T}1 \mid 1\mathbf{T}0 \mid \varepsilon$$

T is the grammar from lecture. It generates all binary strings with the same number of 1s and 0s.

S matches a 2 at the beginning or end. The rest of the string must then match **T** since it cannot have another 2. If neither the first nor last character is a 2, then it falls into the usual cases for matching 0s and 1s, so we can mostly use the same rules as **T**. The main change is that **SS** becomes **ST** | **TS** to ensure that exactly one of the two parts contains a 2. The other change is that there is no ε since a 2 must appear somewhere.

4. Walk the Dawgs

Suppose a dog walker takes care of $n \geq 12$ dogs. The dog walker is not a strong person, and will walk dogs in groups of 3 or 7 at a time (every dog gets walked exactly once). Prove the dog walker can always split the n dogs into groups of 3 or 7.

Solution:

Let $P(n)$ be "a group with n dogs can be split into groups of 3 or 7 dogs." We will prove $P(n)$ for all natural numbers $n \geq 12$ by strong induction.

Base Cases $n = 12, 13, 14,$ or 15: $12 = 3 + 3 + 3 + 3$, $13 = 3 + 7 + 3$, $14 = 7 + 7$, So $P(12)$, $P(13)$, and $P(14)$ hold.

Inductive Hypothesis: Assume that $P(12), \ldots, P(k)$ hold for some arbitrary $k \geq 14$.

Inductive Step: Goal: Show $k + 1$ dogs can be split into groups of size 3 or 7.
We first form one group of 3 dogs. Then we can divide the remaining $k-2$ dogs into groups of 3 or 7 by the assumption $P(k-2)$. (Note that $k \geq 14$ and so $k-2 \geq 12$; thus, $P(k-2)$ is among our assumptions

$P(12), \ldots, P(k)$.)

Conclusion: $P(n)$ holds for all integers $n \geq 12$ by by principle of strong induction.

5. Reversing a Binary Tree

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

Basis Step `Nil` is a **Tree**.

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

The sum function returns the sum of all elements in a **Tree**.

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

The following recursively defined function produces the mirror image of a **Tree**.

$$\begin{aligned} \mathsf{reverse}(\texttt{Nil}) &= \texttt{Nil} \\ \mathsf{reverse}(\texttt{Tree}(x, L, R)) &= \texttt{Tree}(x, \mathsf{reverse}(R), \mathsf{reverse}(L)) \end{aligned}$$

Show that, for all **Tree**s T that

$$\mathsf{sum}(T) = \mathsf{sum}(\mathsf{reverse}(T))$$

Solution:

For a **Tree** T, let $P(T)$ be "$\mathsf{sum}(T) = \mathsf{sum}(\mathsf{reverse}(T))$". We show $P(T)$ for all **Tree**s T by structural induction.

Base Case: By definition we have $\mathsf{reverse}(\texttt{Nil}) = \texttt{Nil}$. Applying sum to both sides we get $\mathsf{sum}(\texttt{Nil}) = \mathsf{sum}(\mathsf{reverse}(\texttt{Nil}))$, which is exactly $P(\texttt{Nil})$, so the base case holds.

Inductive Hypothesis: Suppose $P(L)$ and $P(R)$ hold for some arbitrary **Tree**s L and R.

Inductive Step: Let x be an arbitrary integer. Goal: Show $P(\texttt{Tree}(x, L, R))$ holds.

We have,

$$\begin{aligned} \mathsf{sum}(\mathsf{reverse}(\texttt{Tree}(x, L, R))) &= \mathsf{sum}(\texttt{Tree}(x, \mathsf{reverse}(R), \mathsf{reverse}(L))) && \text{[Definition of reverse]} \\ &= x + \mathsf{sum}(\mathsf{reverse}(R)) + \mathsf{sum}(\mathsf{reverse}(L)) && \text{[Definition of sum]} \\ &= x + \mathsf{sum}(R) + \mathsf{sum}(L) && \text{[Inductive Hypothesis]} \\ &= x + \mathsf{sum}(L) + \mathsf{sum}(R) && \text{[Commutativity]} \\ &= \mathsf{sum}(\texttt{Tree}(x, L, R)) && \text{[Definition of sum]} \end{aligned}$$

This shows $P(\texttt{Tree}(x, L, R))$.

Conclusion: Therefore, $P(T)$ holds for all **Tree**s T by structural induction.