Recursion and Structural Induction Worksheet

Benjamin Cosman, Patrick Lin and Mahesh Viswanathan

Fall 2020

TAKE-AWAYS

- Functions and sets can be defined *recursively*
- Recursive function definitions are especially convenient when the domain is a recursively-defined set
- *Structural induction* is a variant of induction used for recursively-defined sets

Problem 1. Consider a set S defined recursively by $-3 \in S$, $2 \in S$, and if x and y are in S then $x + y$ is in S. [1] Answer each of the following, and briefly justify your answers:

[1] Note that x and y don't have to be different, so e.g. $2 + 2 = 4 \in S$

a) What is the set S? (Give an equivalent non-recursive description, e.g. in set-builder notation or using sets we've named in the past.)

b) More generally, let $R_{a,b}$ be the set defined recursively by $a \in R_{a,b}$, $b \in R_{a,b}$, and if x and y are in $R_{a,b}$ then $x + y$ is in $R_{a,b}$. (So S is just $R_{-3,2}$.) Find a negative c and positive d *other than* -3 and 2 such that $R_{c,d} = S$. [2]

[2] Recall that two sets are the same if they have the same elements, even if their definitions look different.

c) With $R_{a,b}$ defined as above, find a negative integer e and positive integer f such that $R_{e,f} \neq S$.

Problem 2. Recall that the set of binary strings $\{0,1\}^*$ was defined recursively by $\lambda \in \{0,1\}^*$, and if $w \in \{0,1\}^*$, then $0w \in \{0,1\}^*$ and $1w \in \{0,1\}^*$.

a) A *palindrome* is a string that is the same when reversed. For example, 010, 11, and λ (the empty string) are palindromes, but 01 is not. Give a recursive definition for the set of palindromic binary strings. [3]

[3] Hint: What are all the "small" palindromes, and then how can you build a bigger palindrome when given a smaller one?

b) Define a function $countZeros : \{0,1\}^* \to \mathbb{N}$ which uses recursion to count the number of zeros in a (not-necessarily-palindromic) string, e.g. $countZeros(00101) = 3$.

c) Prove by structural induction that $countZeros(st) = countZeros(s) + countZeros(t)$.

Problem 3. We will call a binary tree "slightly balanced" if every vertex has either 0 or 2 children (so no vertex has exactly 1 child).

a) Define the set $SBBT$ of slightly balanced binary trees recursively. [4]

b) Define a recursive function $f : SBBT \to \mathbb{N}$ which counts the number of vertices with 0 children ("final" vertices).

c) Define a recursive function $b : SBBT \to \mathbb{N}$ which counts the number of vertices with 2 children ("branching" vertices). [5]

d) Prove by structural induction that for every $T \in SBBT$, $f(T) = b(T) + 1$.

Problem 4. In propositional logic, any proposition can be expressed as an equivalent proposition that uses only "and"s ($\wedge$) and "not"s ($\neg$). Problem 7.25 in our MCS textbook (pdf p286, printed page number 278) explores whether the same is true for "xor"s ($\oplus$) and "and"s. Solve Problem 7.25.

[4] Hint: Take our definition for *all* binary trees and delete a piece.

[5] Hint: Every vertex is either "final" or "branching", so if you compute $f(T)$ and $b(T)$ on any example tree T, the sum of the results should equal the total number of vertices in T.