# HW7 Individual

CS40 Fall'21

Bharat Kathi (5938444)

Due: Monday, Nov 22, 2021 at 10:00PM on Gradescope

**In this assignment,**

You will work with recursively defined sets and functions and prove properties about them, practicing induction and other proof strategies.

You will submit this assignment via Gradescope (https://www.gradescope.com) in the assignment called "HW7-Individual".

In your proofs and disproofs of statements below, justify each step by reference to the proof strategies we have discussed so far, and/or to relevant definitions and calculations. We include only induction-related strategies here; you can and should refer to past material to identify others.

**Proof by Structural Induction**: To prove that $\forall x \in X\ P(x)$ where $X$ is a recursively defined set, prove two cases:

| | |
|---|---|
| Basis Step: | Show the statement holds for elements specified in the basis step of the definition. |
| Recursive Step: | Show that if the statement is true for each of the elements used to construct new elements in the recursive step of the definition, the result holds for these new elements. |

**Proof by Mathematical Induction**: To prove a universal quantification over the set of all integers greater than or equal to some base integer $b$:

| | |
|---|---|
| Basis Step: | Show the statement holds for $b$. |
| Recursive Step: | Consider an arbitrary integer $n$ greater than or equal to $b$, assume (as the **induction hypothesis**) that the property holds for $n$, and use this and other facts to prove that the property holds for $n+1$. |

**Proof by Strong Induction** To prove that a universal quantification over the set of all integers greater than or equal to some base integer $b$ holds, pick a fixed nonnegative integer $j$ and then:

| | |
|---|---|
| Basis Step: | Show the statement holds for $b$, $b+1$, ..., $b+j$. |
| Recursive Step: | Consider an arbitrary integer $n$ greater than or equal to $b+j$, assume (as the **strong induction hypothesis**) that the property holds for **each of** $b$, $b+1$, ..., $n$, and use this and other facts to prove that the property holds for $n+1$. |

**RNA related definitions**

Consider the following definitions related to RNA strands:

**Definition** Set of bases $B = \{\texttt{A}, \texttt{C}, \texttt{U}, \texttt{G}\}$. The set of RNA strands $S$ is defined (recursively) by:

| | |
|---|---|
| Basis Step: | $\texttt{A} \in S, \texttt{C} \in S, \texttt{U} \in S, \texttt{G} \in S$ |
| Recursive Step: | If $s \in S$ and $b \in B$, then $sb \in S$ |

where $sb$ is string concatenation.

**Definition** The function *rnalen* that computes the length of RNA strands in $S$ is defined recursively by $rnalen : S \to \mathbb{Z}^+$

| | |
|---|---|
| Basis Step: | If $b \in B$, then $rnalen(b) = 1$ |
| Recursive Step: | If $s \in S$ and $b \in B$, then $rnalen(sb) = 1 + rnalen(s)$ |

**Definition** The function *basecount* that computes the number of a given base $b$ appearing in a RNA strand $s$ is defined recursively by $basecount : S \times B \to \mathbb{N}$

Basis step: If $b_1 \in B$, $b_2 \in B$, $basecount(b_1, b_2) = \begin{cases} 1 & \text{when } b_1 = b_2 \\ 0 & \text{when } b_1 \neq b_2 \end{cases}$

Recursive Step: If $s \in S$, $b_1 \in B$, $b_2 \in B$, $basecount(sb_1, b_2) = \begin{cases} 1 + basecount(s, b_2) & \text{when } b_1 = b_2 \\ basecount(s, b_2) & \text{when } b_1 \neq b_2 \end{cases}$

**Definition** The function $mutate : S \times B \to S$ is defined recursively as:

Basis step: If $b_1 \in B$ and $b_2 \in B$, $mutate(b_1, b_2) = \begin{cases} b_2 & \text{when } b_1 = \texttt{A} \\ b_1 & \text{when } b_1 \neq \texttt{A} \end{cases}$

Recursive Step: If $s \in S$, $b_1 \in B$, $b_2 \in B$, $mutate(sb_1, b_2) = \begin{cases} mutate(s, b_2)b_2 & \text{when } b_1 = \texttt{A} \\ mutate(s, b_2)b_1 & \text{when } b_1 \neq \texttt{A} \end{cases}$

**Linked list related definitions**

**Definition** The set of linked lists of natural numbers $L$ is defined by:

$$\begin{array}{ll} \text{Basis Step:} & [\,] \in L \\ \text{Recursive Step:} & \text{If } l \in L \text{ and } n \in \mathbb{N}, \text{ then } (n, l) \in L \end{array}$$

**Definition** The function *removeTail* : $L \to L$ that removes the last node of a linked list (if it exists) is defined by:

$$\begin{array}{ll} & \mathit{removeTail} : L \to L \\ \text{Basis Step:} & \mathit{removeTail}([\,]) = [\,] \\ \text{Recursive Step:} & \text{If } l \in L \text{ and } n \in \mathbb{N}, \text{ then} \\ & \mathit{removeTail}(\ (n, l)\ ) = \begin{cases} [\,], \text{when } l = [\,] \\ (n, \mathit{removeTail}(l)\ ), \text{when } l \neq [\,] \end{cases} \end{array}$$

**Definition** The function *remove* : $L \times \mathbb{N} \to L$ that removes a single node containing a given value (if present) from a linked list is defined by:

$$\begin{array}{llll} & & \mathit{remove} : L \times \mathbb{N} & \to L \\ \text{Basis Step:} & \text{If } m \in \mathbb{N} \text{ then} & \mathit{remove}([\,], m) & = [\,] \\ \text{Recursive Step:} & \text{If } l \in L, n \in \mathbb{N}, m \in \mathbb{N}, \text{ then} & \mathit{remove}((n, l), m) & = \begin{cases} l & \text{when } n = m \\ (n, \mathit{remove}(l, m)) & \text{when } n \neq m \end{cases} \end{array}$$

**Definition:** The function *prepend* : $L \times \mathbb{N} \to L$ that adds an element at the front of a linked list is defined by:

$$\mathit{prepend}(l, n) = (n, l)$$

**Definition** The function *append* : $L \times \mathbb{N} \to L$ that adds an element at the end of a linked list is defined by:

$$\begin{array}{llll} & & \mathit{append} : L \times \mathbb{N} & \to L \\ \text{Basis Step:} & \text{If } m \in \mathbb{N} \text{ then} & \mathit{append}([\,], m) & = (m, [\,]) \\ \text{Recursive Step:} & \text{If } l \in L \text{ and } n \in \mathbb{N} \text{ and } m \in \mathbb{N}, \text{ then} & \mathit{append}((n, l), m) & = (n, \mathit{append}(l, m)) \end{array}$$

# Assigned Questions

Definitions related to RNA and linked list are listed on the previous two pages.

1. (*Graded for correctness*) Calculate the following function applications. Include all intermediate steps, with justifications.

   ---

   *Sample response that can be used as reference for the detail expected in your answer for this part:*

   Calculating $append(\ (1,(2,[\,]))\ ,3)$, we have

   $$\begin{aligned} append(\ (1,(2,[\,]))\ ,3) &= (1, append(\ (2,[\,])\ ,3)) && \text{By recursive step of } append\text{: } n=1,\ l=(2,[\,]),\ m=3 \\ &= (1,\ (2,\ append(\ [\,]\ ,3))) && \text{By recursive step of } append\text{: } n=2,\ l=[\,],\ m=3 \\ &= (1,\ (2,\ (3,[\,]))) && \text{By basis step of } append\text{: } m=3 \end{aligned}$$

   ---

   (a) Calculate $removeTail(\ append(\ (2,(3,[\,])\ ),1\ )\ )$

   (b) Calculate $prepend(\ remove(\ (1,(2,(2,(3,[\,])))),2\ ),3\ )$

2. Consider the following statement and attempted proof:

   $$\forall l \in L\, \exists n \in \mathbb{N}\ (\ append(removeTail(l),n) = l\ )$$

   **Attempted proof:** By structural induction on $L$, we have two cases:

   **Basis Step**: Consider $l = [\,]$ and choose the witness $n = 0$ (in the domain $\mathbb{N}$ since it is a nonnegative integer). We need to show that $append(removeTail(\ (0,[\,])\ ),0) = (0,[\,])$. By the definition of $removeTail$, using the recursive step with $l = [\,]$ and $n = 0$, we have $removeTail(\ (0,[\,])\ ) = [\,]$. By the definition of $append$, using the basis step with $l = [\,]$ and $n$, we have $append([\,],0) = (0,[\,])$ as required.

   **Recursive Step** Consider an arbitrary list $l = (x,l')$, $l' \in L$, $x \in \mathbb{N}$, and we assume as the **induction hypothesis** that:

   $$\exists n \in \mathbb{N}\ (\ append(removeTail(l'),n) = l'\ )$$

   Our goal is to show that $\exists n \in \mathbb{N}\ (\ append(removeTail((x,l')),n) = (x,l')\ )$. Choose the witness $n = x$, a nonnegative integer so in the domain. We need to show that

   $$append(removeTail((x,l')),x) = (x,l')$$

   Applying the definitions:

   $$\begin{aligned} LHS &= append(removeTail((x,l')),x) \\ &= append((x,removeTail(l')),x) && \text{by recursive step of } removeTail\text{, with } l=l' \text{ and } n=x \\ &= (x,l') && \text{by the recursive definition of } append\text{, with } l=l' \text{ and } n=x \\ &= RHS \end{aligned}$$

   as required.

   Thus, the recursive step is complete and we have finished the proof by structural induction. ■

(a) (*Graded for correctness*) Demonstrate that this attempted proof is invalid by providing and justifying a **counterexample** (disproving the statement).

(b) (*Graded for fair effort completeness*) Explain why the attempted proof is invalid by identifying in which step(s) a definition or proof strategy is used incorrectly, and describing how the definition or proof strategy was misused.

3. (*Graded for correctness*) Prove the statement

$$\forall l \in L \, \forall n \in \mathbb{N}( \; removeTail(append(l, n)) = l \; ).$$

4. (*Each part graded for correctness in evaluating statement and for fair effort completeness in the justification)* Statements like these are used to build the specifications for programs, libraries, and data structures (API) which spell out the expected behavior of certain functions and methods. In this HW question, you're analyzing whether and how order matters for the *remove* and *prepend* functions.

   (a) Prove or disprove the following statement:

$$\forall l \in L \, \forall m \in \mathbb{N} \; ( \; prepend( \; remove(l, m) \; , \; m) = l \; ).$$

   (b) Prove or disprove the following statement:

$$\exists l \in L \, \exists m \in \mathbb{N} \; ( \; prepend( \; remove(l, m) \; , m) = l \; ).$$

5. Write the first 6 terms of the sequence that is described by each of the recurrence relations below:

   (a) $f_1 = 0, f_2 = 2$, and $f_n = 5f_{n-1} - 2f_{n-2}$ for $n \geq 3$.

   **Answer:** .

   $f_1 = 0$
   $f_2 = 2$
   $f_3 = 5(2) - 2(0) = 10$
   $f_4 = 5(10) - 2(2) = 46$
   $f_5 = 5(46) - 2(10)) = 210$
   $f_6 = 5(210) - 2(46) = 958$

   $f_n = 0, 2, 10, 46, 210, 958, ...$

   (b) $g_1 = 2$ and $g_2 = 1$. The rest of the terms are given by the formula $g_n = ng_{n-1} + g_{n-2}$.

   **Answer:** .

   $g_1 = 2$
   $g_2 = 1$
   $g_3 = 3(1) + 2 = 5$
   $g_4 = 4(5) + 1 = 21$
   $g_5 = 5(21) + 5 = 110$
   $g_6 = 6(110) + 21 = 681$

   $g_n = 2, 1, 5, 21, 110, 681, ...$

6. (*Graded for correctness*) Prove that

$$\exists n_0 \in \mathbb{N}\, \forall n \in \mathbb{Z}^{\geq n_0} \text{ ( } 3n \leq (n+1)! \text{ )}$$

**Answer:** .

$(n+1)! - 3n$
$= (n+1)(n)(n-1)! - 3n$
$= n((n+1)(n-1)! - 3)$

$n \geq 2 \rightarrow (n-1)! \geq 1$
$(n+1) \geq 3$
$(n+1)(n-1)! \geq 3$
$(n+1)(n-1)! - 3 \geq 0$
$n((n+1)(n-1)! - 3) \geq 0$
$(n+1)! - 3n \geq 0$
$3n \leq (n+1)!$

So there exists some $n_0 = 2$ such that $\forall n \geq 2 \rightarrow 3n \leq (n+1)!$

7. Prove that any amount of postage worth 24 cents or more can be made from 7-cent or 5-cent stamps

**Answer:** .

Solve using strong induction.
Let $P(n)$ be the statement that postage of n cents can be performed using 7 cent and 5 cent stamps.
Basic Step: The propositions $P(24), P(26), P(28)$ are true since:
$24 = 5 + 5 + 7 + 7$
$26 = 5 + 7 + 7 + 7$
$28 = 7 + 7 + 7 + 7$
Inductive Step: The inductive hypothesis is the statement $P(i)$ is true for $24 \leq i \leq j$ where $i$ is an integer with $j \geq 28$. Assuming this is true, we need to show that $P(j+1)$ is true. Using our inductive hypothesis we know that $P(j-4)$ is true since $j - 4 \geq 24$. Therefore, we can create the postage for $j - 4$ using 7 and 5 cent stamps. By adding 1 more 5 cent stamp to our solution for $j - 4$ cents and we will get the solution for $j + 1$ cents. Therefore, we can know that $P(j+1)$ is true.
Since we completed the basic step and the inductive step for the proof, we can say that by strong induction, $P(n)$ is true for all integers $n \geq 24$.

8. Write a recursive algorithm to compute the maximum of a sequence of numbers. Then, use induction to prove that your algorithm outputs the correct value for every non-empty input sequence.

## Attributions