CSE HW4

Rohan Mukherjee

July 17, 2024

1. (a) We define the following functions:

$$\begin{aligned}
\text{cipher} - \text{encode}(\text{nil}) &= \text{nil} \\
\text{cipher} - \text{encode}(\text{cons}(x, L)) &= \text{cons}(\text{nc}(x), \text{cipher} - \text{encode}(L)) \quad \text{for any L: List}
\end{aligned}$$

and

$$\begin{aligned}
\text{cipher} - \text{decode}(\text{nil}) &= \text{nil} \\
\text{cipher} - \text{decode}(\text{cons}(x, L)) &= \text{cons}(\text{pc}(x), \text{cipher} - \text{decode}(L)) \quad \text{for any L: List}
\end{aligned}$$

2. (a)

$$
\begin{aligned}
\text{crazy\_caps\_encode}(\text{nil}) &= \text{nil} \\
\text{crazy\_caps\_encode}(\text{cons}(x, \text{nil})) &= \text{cons}(x, \text{nil}) \\
\text{crazy\_caps\_encode}(\text{cons}(x, \text{cons}(y, \text{nil}))) &= \text{cons}(x, \text{cons}(y, \text{nil})) \\
\text{crazy\_caps\_encode}(\text{cons}(x, \text{cons}(y, \text{cons}(z, L)))) &= \text{cons}(x, \text{cons}(y, \text{cons}(\text{uc}(z), \\
&\qquad \text{crazy\_caps\_encode}(L)))) \quad \text{for any L: List}
\end{aligned}
$$

$$
\begin{aligned}
\text{crazy\_caps\_decode}(\text{nil}) &= \text{nil} \\
\text{crazy\_caps\_decode}(\text{cons}(x, \text{nil})) &= \text{cons}(x, \text{nil}) \\
\text{crazy\_caps\_decode}(\text{cons}(x, \text{cons}(y, \text{nil}))) &= \text{cons}(x, \text{cons}(y, \text{nil})) \\
\text{crazy\_caps\_decode}(\text{cons}(x, \text{cons}(y, \text{cons}(z, L)))) &= \text{cons}(x, \text{cons}(y, \text{cons}(\text{lc}(z), \\
&\qquad \text{crazy\_caps\_decode}(L)))) \quad \text{for any L: List}
\end{aligned}
$$

3. (a) Let $P(S)$ for a list S be the claim that keep(echo(S)) = S. We will prove this by structural induction. First notice that,

$$\begin{aligned} \text{keep}(\text{echo}(\text{nil})) &= \text{keep}(\text{nil}) && \text{definition of echo} \\ &= \text{nil} && \text{definition of keep} \end{aligned}$$

Now suppose that $P(L)$ holds for some list L. Then,

$$\begin{aligned} \text{keep}(\text{echo}(\text{cons}(a, L))) &= \text{keep}(\text{cons}(a, \text{cons}(a, \text{echo}(L)))) && \text{definition of echo} \\ &= \text{cons}(a, \text{drop}(\text{cons}(a, \text{echo}(L)))) && \text{definition of keep} \\ &= \text{cons}(a, \text{keep}(\text{echo}(L))) && \text{definition of drop} \\ &= \text{cons}(a, L) && \text{inductive hypothesis} \end{aligned}$$

Thus, by structural induction, $P(S)$ holds for all lists S, which completes the proof.

(b) Notice that:

$$\begin{aligned} \text{keep}(\text{cons}(1, \text{cons}(2, \text{echo}(L)))) &= \text{cons}(1, \text{drop}(\text{cons}(2, \text{echo}(L)))) && \text{definition of keep} \\ &= \text{cons}(1, \text{keep}(\text{echo}(L))) && \text{definition of drop} \\ &= \text{cons}(1, L) && \text{by part (a)} \end{aligned}$$

4. (a) We define:

$$\begin{aligned}
\text{prefix}(0, \text{L}) &= \text{nil} & \\
\text{prefix}(\text{n}+1, \text{nil}) &= \text{undefined} & \text{for any } n \in \mathbb{N} \\
\text{prefix}(\text{n}+1, \text{cons}(\text{x}, \text{L})) &= \text{cons}(\text{x}, \text{prefix}(\text{n}, \text{L})) & \text{for any } n \in \mathbb{N}, \text{ L: List}
\end{aligned}$$

We continue to define:

$$\begin{aligned}
\text{suffix}(0, \text{L}) &= L & \\
\text{suffix}(\text{n}+1, \text{nil}) &= \text{undefined} & \text{for any } n \in \mathbb{N} \\
\text{suffix}(\text{n}+1, \text{cons}(\text{x}, \text{L})) &= \text{suffix}(\text{n}, \text{L}) & \text{for any } n \in \mathbb{N}, \text{ L: List}
\end{aligned}$$

5. (a) We prove this first claim by cases. If $L = \text{nil}$, then we have that:

$$\begin{aligned} \text{concat(L, cons(b, nil))} &= \text{concat(nil, cons(b, nil))} & \\ &= \text{cons(b, nil)} & \text{def of concat} \\ &\neq \text{nil} & \end{aligned}$$

Otherwise, we can write $L = \text{cons(a, R)}$ for some list R. Then we have,

$$\begin{aligned} \text{concat(L, cons(b, nil))} &= \text{concat(cons(a, R), cons(b, nil))} & \\ &= \text{cons(a, concat(R, cons(b, nil)))} & \text{def of concat} \\ &\neq \text{nil} & \end{aligned}$$

and this last list is not nil since it at least contains a. Thus, the claim holds.

(b) Let $P(S)$ for a list S be the claim that $\text{last(concat(S, cons(b, nil)))} = \text{b}$. We prove the claim by structural induction on S. First, if $S = \text{nil}$, then we have that:

$$\begin{aligned} \text{last(concat(S, cons(b, nil)))} &= \text{last(concat(nil, cons(b, nil)))} & S = \text{nil} \\ &= \text{last(cons(b, nil))} & \text{def of concat} \\ &= b & \text{def of last} \end{aligned}$$

So the base case holds. Now suppose that the claim holds for some list L. Then we have that:

$$\begin{aligned} \text{last(concat(cons(a, L), cons(b, nil)))} &= \text{last(cons(a, concat(L, cons(b, nil))))} & \text{def of concat} \\ &= \text{last(concat(L, cons(b, nil)))} & \text{def of last} \\ &= b & \text{inductive hypothesis} \end{aligned}$$

This completes the proof by structural induction. I don't see where you were meant to use part (a) in the above proof. I believe it might be used to show that the first statement is not undefined, since the inner list is not nil, but I don't see how that is necessary if you order the arguments in this way (you basically repeat part (a) in the proof).

(c) Notice that,

$$\begin{aligned} \text{last(rev(cons(a, R)))} &= \text{last(concat(rev(R), cons(a, nil)))} & \text{def of rev} \\ &= a & \text{by part (b)} \end{aligned}$$

6. (a) We define the function as follows:

$$
\begin{aligned}
\text{worm-latin-encode(L)} &= \text{nil} && L = \text{"bird"} \\
\text{worm-latin-encode(L)} &= \text{L} && \text{L: List}, cc(L) = -1 \\
\text{worm-latin-encode(L)} &= \text{concat}([\text{"w"}, \text{a}, \text{"orm"}], \text{R}) && \text{A} \\
\text{worm-latin-encode(L)} &= \text{concat}(\text{suffix}(\text{cc(L)}, \text{L}), \\
&\qquad \text{concat}(\text{prefix}(\text{cc(L)}, \text{L}), [\text{"orm"}])) && \text{L: List}, cc(L) \geq 1
\end{aligned}
$$

Where A stands for "L: List, cc(L) = 0, L = cons(a, R) for some R : List". Note that if $cc(L) = 0$, then L has a vowel in the first position, so it is not nil and thus of the above form: cons(a, R) for some other list R.