Programming Languages

Homework 1

Amit Moshcovitz and Ari Feiglin

1.1 Exercise

For the relevant code, prove that for every x, y of type `sequence`,

$$\texttt{len}(\texttt{append}\ x\ y) = \texttt{len}(x) + \texttt{len}(y)$$

Proof: by induction on x. For $x = \texttt{Empty}$, $\texttt{append}\ x\ y = y$ and so $\texttt{len}(\texttt{append}\ x\ y) = \texttt{len}(y)$. And $\texttt{len}(x) = 0$ so $\texttt{len}(x) + \texttt{len}(y) = \texttt{len}(y) = \texttt{len}(\texttt{append}\ x\ y)$ as required.

Now, if $x = \texttt{Cons}(v, x')$ then $\texttt{append}\ x\ y = \texttt{Cons}(v, \texttt{append}\ x'\ y)$ and so $\texttt{len}(\texttt{append}\ x\ y) = 1+\texttt{len}(\texttt{append}\ x'\ y)$ which by induction is equal to $1 + \texttt{len}(x') + \texttt{len}(y)$. And $\texttt{len}(x) = 1 + \texttt{len}(x')$, so $\texttt{len}(\texttt{append}\ x\ y) = \texttt{len}(x) + \texttt{len}(y)$ as required.

1.2 Exercise

For the relevant code, show that for every t of type `btree`, the height of t is at least the length of the longest path from the root of t to a leaf.

Proof: by induction on t. If $t = \texttt{Empty}$ then $\texttt{height}(t) = 0$ and the length of the path from the root of t to a leaf is also 0, so the inequality is satisfied. Now if $t = \texttt{Node}(v, t_1, t_2)$ then $\texttt{height}(t) = 1 + \max\{\texttt{height}(t_1), \texttt{height}(t_2)\}$, and so by induction this is at least $\geq 1 + \max\{|P| \mid P \text{ is a path in } t_i\}$, where the maximum is taken over all paths starting from a root of some t_i and ending at a leaf. Where $|\text{path in } t_i|$ is the length of a path from the root of t_i to a leaf. A path from the root of t to a leaf must be contained (other than the root) in t_1 or t_2, and so has a length of $1 + |P|$ for some path P from the root of a t_i to a leaf, and thus has a length bound by $1 + \max_P\{|P|\} \leq \texttt{height}(t)$ as required.

1.3 Exercise

For the relevant code, prove or disprove

(1) for every `exp` of type `bool_expr`,

$$\texttt{num\_of\_vars}(\texttt{exp}) = \texttt{num\_of\_connectives}(\texttt{exp}) + 1$$

(2) for every `exp` of type `bool_expr` not containing `Not`,

$$\texttt{num\_of\_vars}(\texttt{exp}) = \texttt{num\_of\_connectives}(\texttt{exp}) + 1$$

(1) This is false: take $\texttt{exp} = \texttt{Not}(\texttt{"}x\texttt{"})$. Then $\texttt{num\_of\_vars}(\texttt{exp}) = 1$ and $\texttt{num\_of\_connectives}(\texttt{exp}) = 1$.

(2) This is true: proof by induction on `exp`. For $\texttt{exp} = \texttt{Var}(\texttt{"}x\texttt{"})$, $\texttt{num\_of\_vars}(\texttt{exp}) = 1$ and we get $\texttt{num\_of\_connectives}(\texttt{exp}) = 0$ as required. Otherwise, $\texttt{exp} = \circ(\texttt{e}_1, \texttt{e}_2)$ for $\circ \in \{\texttt{And}, \texttt{Or}\}$. And so by induction

$$\begin{aligned}\texttt{num\_of\_vars}(\texttt{exp}) &= \texttt{num\_of\_vars}(\texttt{e}_1) + \texttt{num\_of\_vars}(\texttt{e}_2) \\ &= 2 + \texttt{num\_of\_connectives}(\texttt{e}_1) + \texttt{num\_of\_connectives}(\texttt{e}_2)\end{aligned}$$

And

$$\begin{aligned}\texttt{num\_of\_connectives}(\texttt{exp}) &= 1 + \texttt{num\_of\_connectives}(\texttt{e}_1) + \texttt{num\_of\_connectives}(\texttt{e}_2) \\ &= 1 + \texttt{num\_of\_vars}(\texttt{exp})\end{aligned}$$

as required.