Homework3

Youya Xia

1.power function over natural numbers:

We define our property as ,$\forall n \in \mathbb{N}$,p(n):= $\forall x, power\ n\ x = x^n$

Base case:p(0):

Show:$\forall x, power\ 0\ x = x^0$

$power\ 0\ x = 1.0$

(by definition of *power*)

$= x^0$

(by exponential arithmetic)

Inductive case: p(n+1):

Show: $\forall x, power\ (n+1)\ x = x^{(n+1)}$

Given: $\forall x, power\ n\ x = x^n$

$power\ (n+1)\ x = x *. power\ \big((n+1) - 1\big)\ x$

(by definition of *power*)

$= x *. power\ n\ x$

(by arithmetic)

$= x *. x^n$

(by inductive hypothesis)

$= x^{n+1}$

(by exponential arithmetic)

So by induction principle,$\forall n \in \mathbb{N}$, p(n):= $\forall x, power\ n\ x = x^n$ holds

2.Power over structured numbers:

(1): the principle of induction for nat:

$$\forall n \in nat, p(n)\ holds\ if\ p(Zero)\ holds\ and\ p(n) \Rightarrow p(Succ\ n)$$

(2): we define our property as

$\forall n \in nat$, p(n) := $\forall\ x, power\ n\ x = x^{toInt\ n}$

Base case: p(Zero)

Show: $\forall\ x$,power Zero x=$x^{toInt(Zero)}$

$power\ Zero\ x = 1.0$

(by definition of *power*)

$= x^0$

(by properties of exponential arithmetic)

$= x^{toInt(Zero)}$

(by definition of *toInt*)

Inductive case: p(Succ n):

Show:$\forall\ x, power\ (Succ\ n)\ x = x^{toInt(Succ\ n)}$

Given: $\forall\ x, power\ n\ x = x^{toInt\ n}$

$power\ (Succ\ n)\ x = x *. power\ n\ x$

(by definition of *power*)

$= x *. x^{toInt\ n}$

(by inductive hypothesis)

$= x^{toInt\ n+1}$

(by properties of exponential arithmetic)

$$= x^{toInt\ (Succ\ n)}$$

(by definition of *toInt*)

So, we have shown by induction that $\forall n \in nat, p(n) \coloneqq$

$\forall\ x, power\ n\ x = x^{toInt\ n}$ holds.

3.List reverse and append:

In order to show that:

$reverse\ (append\ l1\ l2) = append\ (reverse\ l2)\ (reverse\ l1)$

We need to show that: $\forall l1$,p(l1):

$\forall\ l2, reverse\ (append\ l1\ l2) = append\ (reverse\ l2)\ (reverse\ l1)$

Before proving this statement, we first need to prove two properties of append :

Lemma 3.1:

We define our property as

$\forall\ l1, p(l1) \coloneqq append\ l1\ [\ ] = l1$

Base case: p([ ])

Show:$append\ [\ ][\ ] = [\ ]$

$append\ [\ ]\ [\ ] = [\ ]$ (by definition of *append*)

Inductive case: p(x::xs):

Show:$append\ (x :: xs)\ [\ ] = x :: xs$

Given: $append\ xs\ [\ ] = xs$

$append\ (x :: xs)\ [\ ] = x :: (append\ xs\ [\ ])$

(by definition of *append*)

$= x :: xs$

(by inductive hypothesis)

Thus, by induction, we have shown that

$\forall\ l1, append\ l1\ [\,] = l1$

Lemma 3.2:

$append\ (append\ l1\ l2)\ l3 = append\ l1\ (append\ l2\ l3)$

and we define our property as:$\forall l1, p(l1)$:

$\forall l2, l3, append\ (append\ l1\ l2)\ l3 = append\ l1\ (append\ l2\ l3)$

Base Case: p([ ]):

Show that $\forall\ l2, l3, append\ (append\ [\,]\ l2)\ l3 =$

$append\ [\,]\ (append\ l2\ l3)$

$append\ (append\ [\,]\ l2)\ l3 = append\ l2\ l3$

(by the definition of *append*)

$= append\ [\,](append\ l2\ l3)$

(by definition of *append*)

Inductive case: p(x::xs):

Show: $\forall\ l2, l3, append\ (append\ (x::xs)\ l2)\ l3 = append\ (x ::$

$xs)\ (append\ l2\ l3)$

Given: $\forall\ l2, l3, append\ (append\ xs\ l2)\ l3 =$

$append\ xs\ (append\ l2\ l3)$

$append\ (append\ (x::xs)\ l2)\ l3 = append(x :: append\ xs\ l2)\ l3$

(by definition of *append*)

$= x :: append\ ((append\ xs\ l2)\ l3)$

(by definition of *append*)

$= x :: (append\ xs\ (append\ l2\ l3))$

(by inductive hypothesis)

$= append\ (x :: xs)\ \ (append\ l2\ l3)$

(by definition of *append*)

Therefore, by induction, we have shown

that:$append\ (append\ l1\ l2)\ l3 = append\ l1\ (append\ l2\ l3)$

Now, we can formally begin our proof for this problem:

Base case: p([ ],l2)

Show:

$\forall\ l2, reverse\ (append\ [\ ]\ l2) = append\ (reverse\ l2)\ (reverse\ [\ ])$

$reverse\ (append\ [\ ]\ l2) = reverse\ l2$

(by definition of *append*)

$= append\ (reverse\ l2)\ [\ ]$

(by ***Lemma 3.1*** of the property of *append*)

$= append\ (reverse\ l2)\ (reverse\ [\ ])$

(by the definition of *reverse*)

Inductive case: p(x::xs)

Show:

$\forall\ l2, reverse\ (append\ (x :: xs)\ l2) =$

$append\ (reverse\ l2)\ (reverse\ (x :: xs))$

Given:

$\forall\ l2, reverse\ (append\ xs\ l2) = append\ (reverse\ l2)\ (reverse\ xs)$

$reverse\ (append\ (x :: xs)\ l2) = reverse\ (x :: (append\ xs\ l2))$

(by definition of *append*)

$= append\ \big(reverse\ (append\ xs\ l2)\big)\ [x]$

(by definition of *reverse*)

$= append\ \ (append\ (reverse\ l2)\ \ (reverse\ xs))\ \ [x]$

(by inductive hypothesis)

$= append(reverse\ l2)\ \ (append\ (reverse\ xs)\ [x])$

(by ***Lemma 3.2*** of append property)

$= append\ (reverse\ l2)\ \ (reverse\ (x :: xs))$

(by definition of *reverse*)

So, we have shown by induction that: $\forall l1, p(l1) \coloneqq \forall l2,$

$reverse\ (append\ l1\ l2) = append\ (reverse\ l2)\ (reverse\ l1)$ holds

4.List processing:

In order to show that :

$someupper\ (l1\ @l2) = someupper\ l1||someupper\ l2$

We need to show that: $\forall l1$, p(l1):

$\forall l2, someupper\ (l1@l2) = someupper\ l1||someuppper\ l2$

Base case: p([ ])

Show:

$\forall l2, someupper\ ([\,]@l2) = someupper\ [\,]||someuppper\ l2$

$someupper\ ([\,]@l2) = someupper\ l2$

(by definition of [ ] and @)

$= false\ ||someupper\ l2$

(by property of disjunction)

$= someupper\ [\,]||someupper\ l2$

(by definition of *someupper*)

Inductive case:p(x::xs)

Show:

$\forall l2, someupper\ \big((x :: xs)@l2\big) = someupper\ (x ::$

$xs)||someuppper\ l2$

Given:

$\forall l2, someupper\ (xs@l2) = someupper\ xs||someuppper\ l2$

$someupper\ \big((x :: xs)@l2\big) = someupper\big(x :: (xs@l2)\big)$

(by definition of @ and : :)

$= isupper\ x||someuppper\ (xs@l2)$

(by definition of *someupper*)

$= isupper\ x\ ||(someupper\ xs||someupper\ l2)$

(by inductive hypothesis)

$= (isupper\ x\ ||someupper\ xs)||\ someupper\ l2$

(by property of disjunction)

$= someupper\ (x :: xs)||someupper\ l2$

(by definition of *someupper*)

So, we have shown by induction that $\forall l1, p(l1) :=$

$\forall l2, someupper\ (l1\ @l2) = someupper\ l1 || someupper\ l2$ holds

5.List processing and folds:

We define our property as $\forall l1$, p(chs) :=

$someupper\ chs = foldupper\ chs$

Base case: p([ ])

$someupper\ [\,] = foldupper\ [\,]$

$foldupper\ [\,] = foldr\ upperor\ [\,]\ false$

(by definition of *foldupper*)

$= false$

(by definition of *foldr*)

$= someupper\ [\,]$

(by definition of *someupper*)

Inductive case: p(x::xs)

Show:$someupper\ (x :: xs) = foldupper\ (x :: xs)$

Given: $someupper\ xs = foldupper\ xs$

$foldupper\ (x :: xs) = foldr\ upperor\ (x :: xs)\ false$

(by definition of *foldupper*)

$= upperor\ x\ (foldr\ upperor\ xs\ false)$

(by definition of *foldr*)

$= upperor\ x\ (foldupper\ xs)$

(by definition of *foldupper*)

$= isupper\ x || foldupper\ xs$

(by definition of *upperor*)

$= isupper\ x\ || someupper\ xs$

(by inductive hypothesis)

$= someupper\ (x :: xs)$

(by definition of *someupper*)

So, we have shown by induction that $\forall l1, p(l1) \coloneqq\ someupper\ chs = foldupper\ chs$ holds

6.Tree processing:

We define our property as $\forall t \in\ 'a\ tree$, p(t) :=

$mintree\ t = fold\_mintree\ t$

Base case: p(Leaf a)

Show:$mintree\ (Leaf\ \ a)$ =fold_mintree $(Leaf\ \ a)$

$fold\_mintree\ \ (Leaf\ \ a) = tfold\ (fun\ x \rightarrow x)$ min$(Leaf\ \ a)$

(by definition of *fold_mintree*)

$= a$

(by definition of *tfold* and identity function)

$= mintree\ (Leaf\ \ a)$

(by definition of *mintree*)

Inductive case: p($Branch\ left,\ \ right$):

Show:$mintree\ Branch\ (left, right) =$

$fold\_mintree\ Branch\ (left, right)$

Given: $mintree\ right = fold\_mintree\ right$ and

$mintree\ left = fold\_mintree\ left$

$fold\_mintree\ Branch\ (left, right)$

$= tfold\ (fun\ x \rightarrow x)\ \text{min}\ Branch\ (left, right)$

(by definition of *fold_mintree*)

$= min\ (tfold\ (fun\ x \rightarrow x)\ \text{min}\ left)\ (tfold\ (fun\ x \rightarrow x)\ \text{min}\ right)$

(by definition of *tfold*)

$= \text{min}(fold\_\ \text{mintree}\ left)\ \ (fold\_mintree\ right)$

(by definition of *tfold* and *fold_mintree*)

$= \text{min}\ (mintree\ left)\ (mintree\ right)$

(by inductive hypothesis)

$= mintree\ Branch\ (left, right)$

(by definition of *mintree*)

So, by induction we have shown that

$\forall t \in\ 'a\ tree,$ p(t):=$mintree\ t = fold\_mintree\ t$ holds