Programming Paradigms 2022
Session 13: Reasoning about programs

Problems for solving and discussing

Hans Hüttel

6 December 2022

Problems that we will definitely talk about

1. ***(Everyone at the table – 10 minutes)***

   Prove by induction on lists that

   $$\text{length } (\text{reverse } xs) = \text{length } xs$$

   (One of the discussion problems from today is your friend.)

2. ***(Work in pairs – 30 minutes)***

   Here is our type declaration for binary trees with a-labelled trees and a declaration that this type as a functor.

```
data Tree a = Leaf a | Node (Tree a) (Tree a)

instance Functor Tree where
-- fmap :: (a -> b) -> Tree a -> Tree b
fmap g (Leaf x) = Leaf (g x)
fmap g (Node l r) = Node (fmap g l) (fmap g r)
```

   Prove by induction on trees that the two functor laws for this type hold (you need to look up these laws in the textbook!).

3. ***(Everyone at the table – 30 minutes)***

   Below is the usual definition of a function defining the Fibonacci numbers, now written in Haskell.

```
fib 0 = 1
fib 1 = 1
fib n = fib (n-1) + fib (n-2)
```

   Prove by induction that if $n > 1$ then $\text{fib}\, n \geq \phi^{n-2}$, where $\phi = \frac{1+\sqrt{5}}{2}$. It is useful to remember that $\phi^2 = 1 + \phi$ (you may want to check this).

   Next, prove by induction that one needs $\text{fib}\, n$ recursive calls to find $\text{fib}\, n$ if $n \geq 2$. What does this tell us about the Haskell implementation of the Fibonacci numbers? (Try finding fib 50.)

More problems to solve at your own pace

a) Find an implementation of the Fibonacci numbers in Haskell that allows you to compute fib 50. *Hint:* Compute the list of all Fibonacci numbers in a recursive manner and find entry number 50.