Constructive Logic — Carnegie Mellon University
Fall 2000 — Frank Pfenning

Assignment 7

Out: Friday Oct 27 — Due: Thursday Nov 2

For all of the following problems you are allowed to use any lemma we have proven in lecture or assignments so far.

1. Verification of *nth* (40 Points)

Consider the function $nth : \mathbf{nat\,list} \to \mathbf{nat} \to \mathbf{nat} \to \mathbf{nat}$ for lists over natural numbers (cf. assignment 4):

$$\begin{array}{lllll} nth & \mathbf{nil} & n & a & = & a \\ nth & (x :: l) & 0 & a & = & x \\ nth & (x :: l) & (\mathbf{s}\,n) & a & = & nth\ l\ n\ a \end{array}$$

We want to verify that *nth* works correctly. Our idea of verification is the following: Given the sequence of natural numbers $l = 0 :: s\,0 :: s\,(s\,0) :: \ldots$ we show that $nth\ x\ l\ a = x$ for some $a \in \mathbf{nat}$. Since we cannot represent an infinite list of all natural number, we will actually prove this proposition for the sequence $0 :: s\,0 :: s\,(s\,0) :: \ldots :: x$. To create these sequences, we use the function *upto* (familiar from the midterm), but this time we define it without accumulator argument to avoid complications in our proofs.

1. (10 Points) Implement $upto : \mathbf{nat} \to \mathbf{nat\,list}$ according to this idea:

   $$\begin{array}{llll} upto & 0 & = & \mathbf{nil} \\ upto & (\mathbf{s}\,n) & = & 0 :: f(upto\ n) \end{array}$$

   where f is a placeholder for an expression that adds 1 to each member of $upto\ n$. Express f in terms of a function $map : (\mathbf{nat} \to \mathbf{nat}) \to \mathbf{nat\,list} \to \mathbf{nat\,list}$ which, given a function g and a list l of natural numbers, returns a list l' resulting from applying g to each member of l.

2. (30 Points) Prove

   $$\forall x \in \mathbf{nat}.\ \exists a \in \mathbf{nat}.\ nth\ (upto\ (\mathbf{s}\ x))\ x\ a =_N x$$

   informally following the examples given in the lecture. You might have to prove lemmata about the interaction of the three functions defined here first.

3. (15 Points extra credit) Implement your proof in `tutch`.

2. Binary natural numbers (30 Points)

Prove the representation theorem for $add : \mathbf{bin} \to \mathbf{bin} \to \mathbf{bin}$ informally (see lecture notes p. 86–92):

$$\forall b \in \mathbf{bin}.\ \forall c \in \mathbf{bin}.\ tonat\ (add\ b\ c) =_N plus\ (tonat b)\ (tonat c)$$

You can use any valid lemma on natural numbers without proof. However, state the lemmata you use and where you use them.

3. Enconding of integers (30 Points)

One possible encoding of integers by already introduced concepts is to represent an integer a as the difference $n - m$ of two natural numbers. This gives rise to the following notational definitions:

$$\begin{aligned} int &= \mathbf{nat} \times \mathbf{nat} \\ a =_I b &= plus\ (\mathbf{fst}\ a)\ (\mathbf{snd}\ b) =_N plus\ (\mathbf{snd}\ a)\ (\mathbf{fst}\ b) \\ nonneg\ a &= (\mathbf{snd}\ a) < \mathbf{s}\,(\mathbf{fst}\ a) \end{aligned}$$

We show that these integers are a sound extension of natural numbers.

1. (10 Points) Implement an embedding $toInt : \mathbf{nat} \to int$ from natural numbers into integers and an embedding $toNat : int \to \mathbf{nat}$ from integers into natural numbers. The application of $toNat$ to a negative number is unspecified, it can return any natural number. Prove informally

$$\forall n \in \mathbf{nat}.\ toNat\ (toInt\ n) =_N n$$

2. (20 Points) Prove informally, that every non-negative integer is represented by a natural number.

$$\forall a \in int.\ nonneg\ a \supset toInt\ (toNat\ a) =_I a$$

3. (15 points extra credit) Implement your proofs in tutch.

Good luck!