Induction

Tue Oct 25, 2011

Recall the issue we faced last time:

What is the realizing computational evidence for induction: $(A(0)\ \&\ \forall x.\ A(x) \Rightarrow A(s(x))) \Rightarrow \forall x.A(x)$?

How do we use $A(0)$ and the function witnessing $\forall x.\ (A(x) \Rightarrow A(s(x)))$ to produce the function for $\forall x. A(x)$?

We need $\lambda(h.\ spread(h; b, f.\ \underline{?}))$ and ? must provide a realizer for $\forall x. A(x)$.

We saw that

$A(0)$	$A(1)$	$A(2)$	...
by	by	by	
a_0	$f(0)(a_0)$	$f(1)(f(0)(a_0))$	
	a_1	a_2	

But ? can't be .... It must be a computation expression, a program. Something like let rec $ind(x) = f(x-1)(ind(x-1))$ might work. In this case we need $x>0$ to allow $f(x-1)$. What about the 0 case? Then we use the base, b. Thus

$$ind(x) = \underline{\text{if}}\ x=0\ \underline{\text{then}}\ b\ \underline{\text{else}}\ f(x-1, ind(x-1))$$

In the Nuprl style proof rules we use

$ind(x; b; u,i.\ f(u,i))$ with these computation rules.

$ind(0; b; -) \downarrow b \qquad ind(s(x); b; u,i.f(u,i)) \downarrow f(x, ind(x; b; u,i.f(u,i)))$

CS 5860

Tue Oct 25, 2011

Induction continued

We use the induction form for all computation on numbers.

For example, to decide whether a number is zero, we compute with ind(x; *; u,i. —) where — is any computation form that "aborts" like ap(0;0).

We can use ind to prove statements such as

1. $\forall x.(Z(x) \vee \sim Z(x))$

2. $\forall x,y.(Eq(x,y) \vee \sim Eq(x,y))$

3. $\forall x (\sim Z(x) \Rightarrow \exists y. Suc(y,x))$

We can use induction to prove

4. $\forall x,y. \exists z. Add(x,y,z)$

5. $\forall x,y. \exists z. Mult(x,y,z)$

We will prove 4 below. You should try 1, 2, 3, 5 as exercises. Also try 6 below.

6. Define $x<y$ iff $\exists z.(x+z = y \;\&\; z \neq 0)$. Show:

   (a) $x < s(x)$ (b) $(x<y \;\&\; y<z) \Rightarrow x<z$ (c) $\sim(x<x)$

CS 5860 Using Induction

Tue Oct 25, 2011

Recall 1. $\forall y. Add(0,y,y)$ Kleene Ax 18 $add(0,y) = y$

2. $\forall x,y,z. (Add(x,y,z) \Rightarrow Add(s(x),y,s(z)))$

Kleene Ax 19 $add(s(x),y) = s(add(x,y))$

We show Theorem $\forall x,y. \exists z. Add(x,y,z)$ by induction (Kleene Ax 13)

$\vdash \forall x \forall y. \exists z. Add(x,y,z)$ by $\lambda(x. \_\_)$

$x:D \vdash \forall y \exists z. Add(x,y,z)$ by $\lambda(y. \_\_)$

$x:D, y:D \vdash \exists z. Add(x,y,z)$ by $ind(x; \_\_; u,i. \_\_)$

$\vdash \exists z. Add(0,y,z)$ by $\langle y, \_\_ \rangle$

$\vdash D$ by y

$\vdash Add(0,y,y)$ by $ap(ax18; y)$

$x:D, y:D, u:D, i: \exists z. Add(u,y,z) \vdash \exists z. Add(s(u),y,z)$ by $spread(i; z_0, a. \_\_)$

$z_0:D, a: Add(u,y,z_0) \vdash \exists z. Add(s(u),y,z)$ by $\langle s(z_0), \_\_ \rangle$

$\vdash D$ $s(z_0)$

$\vdash Add(s(u),y, s(z_0))$

$ax19(): Add(\overset{u}{x},y,z_0) \Rightarrow Add(s(u),y,s(z_0))$ by $ap(ax19(s(u),y,s(z_0)))$

$\lambda(x. \lambda(y. ind(\langle y, ap(ax18,y) \rangle; \ldots$

$u,i. spread(i; z_0, a. \langle s(z_0), ap(ax19(s(u),y,s(z_0))) \rangle))))$

CS 5860 Induction continued

Tue Oct 25, 2011

The computation form $ind(x; b; u,i.f)$ is recursive. You might know that recursion is more expensive than iteration. Can we use computational forms such as

$$\underline{\text{for}}\ i = 0 \text{ to } n\ \underline{\text{do}}\quad x := f(i,x)\ \underline{\text{od}}$$

as forms of induction? We'd be computing

$$f(0,x_0),\quad f(1, f(0,x_0)),\quad f(2, f(1, f(0,x_0))), \ldots$$

If we set $x = b$, this would be a version of induction. Here is another version.

$x := b\ ;\ i := 0$

$\underline{\text{while}}\ i < x\ \underline{\text{do}}$

$\quad x := f(i,x)$

$\underline{\text{od}}$

It would be interesting to see if we can treat

$\underline{\text{while}}\ b(x)\ \underline{\text{do}}$

$\quad x := f(x)$

$\underline{\text{od}}$

as a realizer in pure first-order logic. We will consider this further on Thursday.

CS 5860

Induction as Iteration

Tue Oct 25, 2011

We can have a more efficient form of induction if we used a "tail recursive" or iterative form. For example, we can define an iterative add that uses only a constant amount of space vs a linear amount.

$add(x, y) = it\text{-}add(x, y, y)$ where

$it\text{-}add(x, y, z) =$

<u>if</u> $x = 0$ <u>then</u> z

<u>else</u> $it\text{-}add(x-1, y, z+1)$

What is the iterative form of induction?