Constructive Logic (15-317), Fall 2015
Recitation 4: Quantifiers and Arithmetic (9/23/2015)
Vincent Huang (`vincenth@andrew`), Anna Gommerstadt (`hgommers@cs`), Michael Coblenz (`mcoblenz@cs`)

0 Sidenote

A bunch of people asked if this is a valid reduction step:

$$\langle \mathrm{fst}\, u, \mathrm{snd}\, u\rangle \Longrightarrow_R u$$

This is actually a thing that some people call an eta-reduction (kind of like the opposite of eta-expansions). We don't have a rule for this in our system, but it's a thing.

1 Quantifier proofs

Task 1. $(\forall x : \tau.\ \neg A(x)) \supset (\neg\exists x : \tau.\ A(x))\ \mathit{true}$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\exists x:\tau.\ A(x)\ \mathit{true}}\,v \qquad \dfrac{\dfrac{}{A(t)\ \mathit{true}}\,w \qquad \dfrac{\dfrac{}{\forall x:\tau.\ \neg A(x)\ \mathit{true}}\,v \qquad \dfrac{}{t:\tau}}{\neg A(t)\ \mathit{true}}\,\forall E}{\bot\ \mathit{true}}\,\supset E}{\bot\ \mathit{true}}\,\exists E^{t,w}}{\neg\exists x:\tau.\ A(x)\ \mathit{true}}\,\supset I^v}{\forall x:\tau.\ \neg A(x) \supset \neg\exists x:\tau.\ A(x)\ \mathit{true}}\,\supset I^u$$

Task 2. $(\exists x : \tau.\ P(x) \wedge Q(x)) \supset (\exists x : \tau.\ P(x) \wedge \exists x : \tau.\ Q(x))\ \mathit{true}$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\exists x:\tau.\ P(x)\wedge Q(x)\ \mathit{true}}\,u \qquad \dfrac{\dfrac{}{a:\tau} \qquad \dfrac{\dfrac{}{P(a)\wedge Q(a)\ \mathit{true}}\,v}{P(a)\ \mathit{true}}\,\wedge E_L}{\exists x:\tau.\ P(x)\ \mathit{true}}\,\exists I}{\exists x:\tau.\ P(x)\ \mathit{true}}\,\exists E^{a,v} \qquad \dfrac{\dfrac{}{\exists x:\tau.\ P(x)\wedge Q(x)\ \mathit{true}}\,u \qquad \dfrac{\dfrac{}{b:\tau} \qquad \dfrac{\dfrac{}{P(a)\wedge Q(a)\ \mathit{true}}\,w}{Q(a)\ \mathit{true}}\,\wedge E_R}{\exists x:\tau.\ Q(x)\ \mathit{true}}\,\exists I}{\exists x:\tau.\ Q(x)\ \mathit{true}}\,\exists E^{b,w}}{\exists x:\tau.\ P(x)\wedge\exists x:\tau.\ Q(x)\ \mathit{true}}\,\wedge I}{(\exists x:\tau.\ P(x)\wedge Q(x))\supset(\exists x:\tau.\ P(x)\wedge\exists x:\tau.\ Q(x))\ \mathit{true}}\,\supset I^u$$

Task 3. $(P \wedge \exists x : \tau.\ Q(x)) \supset (\exists x : \tau.\ P \wedge Q(x))\ \mathit{true}$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{P\wedge\exists x:\tau.\ Q(x)\ \mathit{true}}\,u}{\exists x:\tau.\ Q(x)\ \mathit{true}}\,\wedge E_R \qquad \dfrac{\dfrac{}{a:\tau} \qquad \dfrac{\dfrac{\dfrac{}{P\wedge\exists x:\tau.\ Q(x)\ \mathit{true}}\,u}{P\ \mathit{true}}\,\wedge E_L \qquad \dfrac{}{Q(a)\ \mathit{true}}\,v}{P\wedge Q(a)\ \mathit{true}}\,\wedge I}{\exists x:\tau.\ P\wedge Q(x)\ \mathit{true}}\,\exists I}{\exists x:\tau.\ P\wedge Q(x)\ \mathit{true}}\,\exists E^{a,v}}{(P\wedge\exists x:\tau.\ Q(x))\supset(\exists x:\tau.\ P\wedge Q(x))\ \mathit{true}}\,\supset I^u$$

Task 4. $(P \vee \forall x : \tau.\ Q(x)) \supset (\forall x : \tau.\ P \vee Q(x))\ \mathit{true}$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{P\vee\forall x:\tau.\ Q(x)\ \mathit{true}}\,u \qquad \dfrac{\dfrac{}{P\ \mathit{true}}\,v}{P\vee Q(a)\ \mathit{true}}\,\vee I_L \qquad \dfrac{\dfrac{\dfrac{}{\forall x:\tau.\ Q(x)\ \mathit{true}}\,w \qquad \dfrac{}{a:\tau}}{Q(a)\ \mathit{true}}}{P\vee Q(a)\ \mathit{true}}\,\vee I_R}{P\vee Q(a)\ \mathit{true}}\,\vee E^{v,w}}{\forall x:\tau.\ P\vee Q(x)\ \mathit{true}}\,\forall I^a}{P\vee\forall x:\tau.\ Q(x)\supset\forall x:\tau.\ P\vee Q(x)\ \mathit{true}}\,\supset I^u$$

2 Natural numbers

2.1 Recap of rules

$$\frac{}{0 : \mathsf{nat}}\ \mathsf{nat}I_0 \qquad\qquad \frac{n : \mathsf{nat}}{\mathsf{S}\,n : \mathsf{nat}}\ \mathsf{nat}I_s \qquad\qquad \frac{n : \mathsf{nat} \qquad C(0)\ \mathit{true} \qquad \begin{array}{c}\overline{x : \mathsf{nat}} \quad \overline{C(x)\ \mathit{true}}^{\,u} \\ \vdots \\ C(\mathsf{S}\,x)\ \mathit{true}\end{array}}{C(n)\ \mathit{true}}\ \mathsf{nat}E^{x,u}$$

Recall how the nat$E$ rule is essentially induction.

Task 5. Prove

$$\forall n : \mathsf{nat}.\ C(0) \supset (\forall x : \mathsf{nat}.\ C(x) \supset C(\mathsf{S}\,x)) \supset C(n)$$

2.2 Sidenote on local reduction

In lecture, we saw a perfectly normal-looking local reduction involving the natI_0 rule. Then we followed that up with a really strange-looking local reduction involving natI_s

$$\frac{\dfrac{\begin{array}{c}\mathcal{D} \\ n' : \mathsf{nat}\end{array}}{\mathsf{s}\,n' : \mathsf{nat}}\ \mathsf{nat}I_s \qquad \begin{array}{c}\mathcal{E} \\ C(0)\ \mathit{true}\end{array} \qquad \begin{array}{c}\overline{x : \mathsf{nat}} \quad \overline{C(x)\ \mathit{true}}^{\,u} \\ \mathcal{F} \\ C(\mathsf{s}\,x)\ \mathit{true}\end{array}}{C(\mathsf{s}\,n')\ \mathit{true}}\ \mathsf{nat}E^{x,u}$$

$$\Longrightarrow_R \qquad \begin{array}{c} \begin{array}{c}\mathcal{D} \\ n' : \mathsf{nat}\end{array} \quad \dfrac{\begin{array}{c}\mathcal{D} \\ n' : \mathit{nat}\end{array} \quad \begin{array}{c}\mathcal{E} \\ C(0)\ \mathit{true}\end{array} \quad \begin{array}{c}\overline{x : \mathsf{nat}} \quad \overline{C(x)\ \mathit{true}}^{\,u} \\ \mathcal{F} \\ C(s\,x)\ \mathit{true}\end{array}}{C(n')\ \mathit{true}}\ \mathsf{nat}E^{x,u} \\ [n'/x]\mathcal{F}' \\ C(s\,n')\ \mathit{true}\end{array}$$

where everything looks bigger and messier and more convoluted. How is this even a reduction???

The important part is that initially, we were using nat$E$ on some $\mathsf{S}\,n'$, but after the reduction, we're using nat$E$ on n'. We've "decomposed" $\mathsf{S}\,n'$ into its components (ok, single component), and it should be intuitively obvious that n' is smaller than $\mathsf{S}\,n'$, since it contains one less constructor. Finally, since our introduction rules for natural numbers guarantee us a well-founded order for them, we know that this process terminates after a finite amount of time.

Having said that, we now observe that we can't really do a local expansion with the stuff we have now. Perfect segue into *primitive recursion*!

2.3 Primitive recursion

We allow primitive recursion on natural numbers by a proof term assignment, where we define a *recursor*, which we call R.

$$\frac{n : \mathsf{nat} \qquad t_0 : \tau \qquad \begin{array}{c}\overline{x : \mathsf{nat}} \quad \overline{r : \tau} \\ \vdots \\ t_s : \tau\end{array}}{R(n, t_0, x.\,r.\,t_s) : \tau}\ \mathsf{nat}E^{x,r} \qquad\qquad \frac{n : \mathsf{nat} \qquad M_0 : C(0)\ \mathit{true} \qquad \begin{array}{c}\overline{x : \mathsf{nat}} \quad \overline{u : C(x)\ \mathit{true}} \\ \vdots \\ M_s : C(\mathsf{S}\,x)\ \mathit{true}\end{array}}{R(n, M_0, x.\,u.\,M_s) : C(n)\ \mathit{true}}\ \mathsf{nat}E^{x,u}$$

Ok, lots of notation; what does it *mean*?

- t_0 is what we get back when $n = 0$, i.e. the base case
- t_s is what we get back when $n = \mathsf{S}\,n'$, i.e. the inductive/recursive case

- $x = n'$, i.e. x is bound to the predecessor of n
- $r = R(n', t_0, x.\, r.\, t_s)$, i.e. r is bound to the result of calling the recursor on the predecessor of n

Similarly, on the side with proof terms (M's instead of t's).

2.3.1 Local reduction

There are 2 cases for the local reduction with proof terms, of course.

$$\begin{aligned} R(0, M_0, x.\, u.\, M_s) &\Longrightarrow_R M_0 \\ R(\mathsf{S}\, n', M_0, x.\, u.\, M_s) &\Longrightarrow_R [R(n', M_0, x.\, u.\, M_s)/u][n'/x]M_s \end{aligned}$$

2.3.2 Local expansion

Using the recursor as defined on the datatype, we can write a local expansion!

$$\begin{array}{c} \mathcal{D} \\ n : \mathrm{nat} \end{array} \quad \Longrightarrow_E \quad \dfrac{\begin{array}{c} \mathcal{D} \\ n : \mathrm{nat} \end{array} \quad \dfrac{}{0 : \mathrm{nat}}\,\mathrm{nat}I_0 \quad \dfrac{\overline{x : \mathrm{nat}}}{\mathsf{S}\, x : \mathrm{nat}}\,\mathrm{nat}I_s}{R(n, 0, x.\, r.\, \mathsf{S}\, x) : \mathrm{nat}}\,\mathrm{nat}E^{x,r}$$

2.4 Adding and multiplying

Task 6. Define plus on natural numbers.

How do we think about addition in terms of primitive recursion? Well,

$$\begin{aligned} m + 0 &= m \\ m + \mathsf{S}\, n &= \mathsf{S}(m + n) \end{aligned}$$

$$\mathrm{plus} = \lambda m : \mathrm{nat}.\ \lambda n : \mathrm{nat}.\ R(n, m, n'.\ r.\ \mathsf{S}\ r)$$

Task 7. Define mult on natural numbers, using plus.

$$\begin{aligned} m \times 0 &= 0 \\ m \times (\mathsf{S}\, n) &= (m \times n) + m \end{aligned}$$

$$\mathrm{mult} = \lambda m : \mathrm{nat}.\ \lambda n : \mathrm{nat}.\ R(n, 0, x.\ r.\ \ \mathrm{plus}(r, m))$$

Task 8. Define the Ackermann function!