Constructive Logic (15-317), Fall 2016
Recitation 4: Quantifiers and Arithmetic (9/20/2016)
Evan Cavallo (ecavallo@cs.cmu.edu), Oliver Daids ojd@andrew, Giselle Reis greis@andrew

1 Quantifier proofs

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

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

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

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

Task 3. $(P \vee \forall x : \tau.\ Q(x)) \supset (\forall x : \tau.\ P \vee Q(x))\ true$

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

[Evan: In recitation I mentioned that the converse of this task, $(\forall x{:}\tau.P \vee Q(x)) \supset P \vee \forall x{:}\tau.Q(x)\ true$, is classically but not constructively provable. Constructively, the assumption $\forall x{:}\tau.P \vee Q(x))\ true$ we get is a program M which, for any $x{:}\tau$, spits out either a proof of $P\ true$ or a proof of $Q(x)\ true$. We are given this program, and we want to construct either a program that proves $P\ true$ or one that proves $\forall y{:}\tau.Q(y)\ true$ (I'm changing variable names for clarity here).

If we are thinking classically, then we know that either $M(x)$ produces a proof of $P\ true$ on some $x{:}\tau$ or it produces a proof of $Q(x)\ true$ for every $x{:}\tau$. In either case, we have what we need.

Constructively, however, we can't decide that one of these is true without concrete evidence. We have to either *find* an $x{:}\tau$ such that $M(x)$ gives a proof of $P\ true$, or check that it produces a $Q(x)$ for every $x{:}\tau$. If τ = nat, say, we can start enumerating natural numbers n and checking the result of $M(n)$, but we can't check all of them exhaustively.

As a concrete example, consider the following open conjecture in number theory:

> **Conjecture:** There are no odd perfect numbers. ("A perfect number is a positive integer that is equal to the sum of its proper positive divisors" – Wikipedia)

Let

$$\begin{aligned} P &= \text{"there exists an odd perfect number"} \\ Q(x) &= \text{"}x\text{ is not an odd perfect number"} \end{aligned}$$

Then we can define a terminating program that proves $\forall x{:}\tau.P \vee Q(x)$ *true*: given any x, it checks if x is odd and perfect, which is possible in finite time. If it is, then we have a proof of P *true*, and if not, we have a proof of $Q(x)$ *true*. However, there is no way to use this program to construct a new terminating program which proves $P \vee \forall x{:}\tau.Q(x)$ *true*, because this program would have to decide if there are *any* odd perfect numbers – if we could, we would solve the conjecture!]

2 Natural numbers

2.1 Recap of rules

$$\frac{}{\mathsf{z} : \mathrm{nat}}\ \mathrm{nat}I_{\mathsf{z}} \qquad \frac{n : \mathrm{nat}}{\mathsf{s}\, n : \mathrm{nat}}\ \mathrm{nat}I_{\mathsf{s}}$$

$$\frac{n : \mathrm{nat} \qquad C(\mathsf{z})\ \mathit{true} \qquad \begin{array}{c}\overline{x : \mathrm{nat}} \quad \overline{C(x)\ \mathit{true}}^{\,u} \\ \vdots \\ C(\mathsf{s}\, x)\ \mathit{true}\end{array}}{C(n)\ \mathit{true}}\ \mathrm{nat}E^{x,u}$$

2.2 Primitive recursion

$$\frac{n : \mathrm{nat} \qquad t_{\mathsf{z}} : \tau \qquad \begin{array}{c}\overline{x : \mathrm{nat}} \quad \overline{r : \tau} \\ \vdots \\ t_{\mathsf{s}} : \tau\end{array}}{R(n, t_{\mathsf{z}}, x.\, r.\, t_{\mathsf{s}}) : \tau}\ \mathrm{nat}E^{x,r}$$

2.2.1 Local reduction

$$R(\mathsf{z}, M_{\mathsf{z}}, x.\, u.\, M_{\mathsf{s}}) \Longrightarrow_R M_{\mathsf{z}}$$
$$R(\mathsf{s}\, n', M_{\mathsf{z}}, x.\, u.\, M_{\mathsf{s}}) \Longrightarrow_R [R(n', M_{\mathsf{z}}, x.\, u.\, M_{\mathsf{s}})/u][n'/x]M_{\mathsf{s}}$$

2.2.2 Local expansion

$$\begin{array}{c}\mathcal{D} \\ n : \mathrm{nat}\end{array} \qquad \Longrightarrow_E \qquad \frac{\begin{array}{c}\mathcal{D} \\ n : \mathrm{nat}\end{array} \qquad \frac{}{\mathsf{z} : \mathrm{nat}}\ \mathrm{nat}I_{\mathsf{z}} \qquad \frac{\overline{x : \mathrm{nat}}}{\mathsf{s}\, x : \mathrm{nat}}\ \mathrm{nat}I_{\mathsf{s}}}{R(n, \mathsf{z}, x.\, r.\ \mathsf{s}\, x) : \mathrm{nat}}\ \mathrm{nat}E^{x,r}$$

2.3 Arithmetic

Task 4. Define `double` on natural numbers.

Solution:

$$\mathtt{double}(n) \triangleq R(n, \mathsf{z}, x.r.\, \mathsf{s}(\mathsf{s}\, r))$$

Task 5. Evaluate `double` (s(s z)).

Solution:

$$\begin{array}{rcl}
\mathtt{double}(\mathsf{s}(\mathsf{s}\,\mathsf{z})) & = & R(\mathsf{s}(\mathsf{s}\,\mathsf{z}), \mathsf{z}, x.r.\, \mathsf{s}(\mathsf{s}\, r)) \\
& \Rightarrow_R & [\mathsf{s}\,\mathsf{z}/x][R(\mathsf{s}\,\mathsf{z}, \mathsf{z}, x.r.\, \mathsf{s}(\mathsf{s}\, r))/r]\, \mathsf{s}(\mathsf{s}\, r) \\
& = & \mathsf{s}(\mathsf{s}(R(\mathsf{s}\, z, \mathsf{z}, x.r.\, \mathsf{s}(\mathsf{s}\, r)))) \\
& \Rightarrow_R & \mathsf{s}(\mathsf{s}([\mathsf{z}/x][R(\mathsf{z}, \mathsf{z}, x.r.\, \mathsf{s}(\mathsf{s}\, r))/r]\, \mathsf{s}(\mathsf{s}\, r))) \\
& = & \mathsf{s}(\mathsf{s}(\mathsf{s}(\mathsf{s}(R(\mathsf{z}, \mathsf{z}, x.r.\, \mathsf{s}(\mathsf{s}\, r)))))) \\
& \Rightarrow_R & \mathsf{s}(\mathsf{s}(\mathsf{s}(\mathsf{s}\,\mathsf{z})))
\end{array}$$

Task 6. Define `plus` on natural numbers.

Solution:

$$\texttt{plus}(m, n) \triangleq R(m, n, x.r.\, \texttt{s}\, r)$$

Task 7. Define `minus` on natural numbers.

Solution: First we define a predecessor function:

$$\texttt{pred}(n) \triangleq R(n, \texttt{z}, x.r.x)$$

Then we can define

$$\texttt{minus}(m, n) \triangleq R(m, n, x.r.\texttt{pred}(r))$$