Constructive Logic (15-317), Spring 2020
Assignment 4:

Instructor: André Platzer
TAs: Avery Cowan, Cameron Wong, Carter Williams, Klaas Pruiksma

Due: Tuesday, Feburary 11, 2020, 11:59 pm

1 Quantification

It is important to note that quantification extends as far to the right as syntactically possible. For example, the proposition $\exists x : \tau. A(x) \supset \forall x : \tau. A(x)$ should be interpreted as $\exists x : \tau.(A(x) \supset \forall x : \tau. A(x))$ and not as $(\exists x : \tau. A(x)) \supset (\forall x : \tau. A(x))$. Tutch implements the same convention.

1.1 Distributivity properties

If you're a logic nerd, you may know that universal quantification distributes over conjuction, that is,

$$(\forall x : \tau. A(x) \wedge B(x)) \equiv (\forall x : \tau. A(x)) \wedge \forall x : \tau. B(x) \textit{ true}.$$

In this section, we will explore various other distributivity properties.

Task 1 (8 points)**.** Dually, existential quantification distributes over disjunction, that is,

$$(\exists x : \tau. A(x) \vee B(x)) \equiv (\exists x : \tau. A(x)) \vee \exists x : \tau. B(x) \textit{ true}.$$

In this task, you will show this equivalence by giving a natural deduction proof of each of the following directions:

a. $(\exists x : \tau. A(x) \vee B(x)) \supset (\exists x : \tau. A(x)) \vee \exists x : \tau. B(x)$ *true*

b. $(\exists x : \tau. A(x)) \vee (\exists x : \tau. B(x)) \supset \exists x : \tau. A(x) \vee B(x)$ *true*

1.2 Constructive and classical quantification

Task 2 (8 points)**.** For each of the following judgments, give a constructive natural deduction proof and the corresponding proof term if it is constructively valid. If it is not constructively valid, state this and briefly explain why. *N.B. The following judgments are all classically valid.*

a. $(\neg\forall x : \tau. \neg A(x)) \supset \exists x : \tau. A(x)$ *true*

b. $(\exists x : \tau. A(x)) \supset \neg\forall x : \tau. \neg A(x)$ *true*

1.3 Tutch, Quantified

Tutch uses the concrete syntax `?x:t.A(x)` and `!x:t.A(x)` for $\exists x : \tau.A(x)$ and $\forall x : \tau.A(x)$, respectively. We encourage you to review the scoping rules for quantifiers described at the beginning of Section 1 of this assignment before starting this portion of the assignment. Please see the Tutch manual for more information on how to use quantifiers in Tutch.

Task 3 (8 points)**.** Prove each of the following propositions using Tutch. Place the proof for part a (and only the proof for part a) in `hw4_8a.tut`, ..., and the proof for part c (and only the proof for part c) in `hw4_8c.tut`.

a.
```
proof apply : (!x:t.A(x) => B(x))
              => (!x:t.A(x)) => (!x:t.B(x));
```

b.
```
proof instance : (!x:t.A(x)) & (?y:t.B(y)) => ?z:t.A(z);
```

c.
```
proof frobenius : (R & ?x:t.Q(x)) <=> ?x:t.(R & Q(x));
```

2 Constructive Arithmetic

Recall that our discussion of quantified logic glossed over the judgment $t : \tau$ assigning terms to types and its interaction with the rest of the logic. Here, we'll formally define the system of Heyting arithmetic.

2.1 Natural Numbers

First, here are the rules for the judgment $n : \mathsf{nat}$.

$$\frac{}{0 : \mathsf{nat}}\ \mathsf{nat}I_0 \qquad \frac{x : \mathsf{nat}}{\mathsf{s}\ x : \mathsf{nat}}\ \mathsf{nat}I_S$$

$$\frac{x : \mathsf{nat} \quad C(0)\ \mathit{true} \quad \begin{array}{c}\overline{y : \mathsf{nat}} \quad \overline{C(y)\ \mathit{true}}^{\,u} \\ \vdots \\ C(\mathsf{s}\ y)\ \mathit{true}\end{array}}{C(x)\ \mathit{true}}\ \mathsf{nat}E^{y,u}$$

2.2 Equality

Now, we'll add a new atomic proposition, equality, and define the truth judgment at equality.

$$\frac{}{0 = 0\ \mathit{true}} = I_{00} \qquad \frac{x = y\ \mathit{true}}{\mathsf{s}\ x = \mathsf{s}\ y\ \mathit{true}} = I_{SS}$$

$$\frac{0 = \mathsf{s}\ x\ \mathit{true}}{C\ \mathit{true}} = E_{0S} \qquad \frac{\mathsf{s}\ x = 0\ \mathit{true}}{C\ \mathit{true}} = E_{S0} \qquad \frac{\mathsf{s}\ x = \mathsf{s}\ y\ \mathit{true}}{x = y\ \mathit{true}} = E_{SS}$$

2.3 Primitive Recursion

Lastly, we need the ability to define primitive recursive functions like addition and multiplication *within* the theory of the natural numbers so that we may reason about them in the natural deduction system with the induction principle introduced two subsections ago.

$$\frac{}{R(0; t_0; x, r.t_S) \Rightarrow_R t_0} \Rightarrow_R I_0 \qquad \frac{}{R(\mathsf{s}\; n; t_0; x, r.t_S) \Rightarrow_R [R(n; t_0; x, r.t_S)/r][n/x]t_S} \Rightarrow_R I_S$$

$$\frac{A(x)\ \textit{true} \quad x \Rightarrow_R y}{A(y)\ \textit{true}} \Rightarrow_R E_1 \qquad \frac{A(y)\ \textit{true} \quad x \Rightarrow_R y}{A(x)\ \textit{true}} \Rightarrow_R E_2$$

If you are confused by the rules for reducing primitive recursive forms, you can think of the sentence $R(n; t_0; x, r.t_S)$ as a recursive function as follows:

- If $n = 0$, then return t_0.
- Otherwise, let $x = n - 1$ and $r = R(n - 1; t_0; x, r.t_S)$ in t_S.

Make sure to understand this definition and how it relates to the given rules before attempting the problems.

Task 4 (6 points)**.** Let `double`(n) be defined by $R(n; 0; x, r.\mathsf{s}\ (\mathsf{s}\ r))$. Using the reduction rules above (involving the judgment $t \Rightarrow_R t'$), reduce the term `double`$(\mathsf{s}\ (\mathsf{s}\ 0))$ to a normal form, or until neither of the rules apply. Although it is not a rule, you should also step "underneath" the successors – that is, you may assume that, if $x \Rightarrow_R y$, then $\mathsf{s}\ x \Rightarrow_R \mathsf{s}\ y$.

Task 5 (10 points)**.** Let $m + n$ be defined by $R(m; n; x, r.\mathsf{s}\ r)$. Show that 0 is the right additive identity i.e. give a derivation of $\forall m : \mathsf{nat}.\ R(m; 0; x, r.\mathsf{s}\ r) = m$ *true*.