RECITATION 2

1. Proofs Are Programs

As discussed previously in lecture, there is a tight correspondence between the structure of a derivation for a constructive proof and a term in some particular programming language. This leads to the slogans "proofs are programs" and "propositions are types". The (Curry-Howard-Lambek) correspondence can be fleshed out for the logic we're studying (intuitionistic propositional logic)[1] by the following table

Propositions	Types
$A \wedge B$	$A * B$
$A \vee B$	$A + B$
$A \supset B$	$A \to B$
$\top$	1 (unit)
$\bot$	0 (void)

Based on this we can produce a version of our rules from the previous recitation that annotate each proposition step in the derivation with the program that it constructs. Those rules are

$$\frac{M : A \qquad N : B}{\langle M, N\rangle : A \wedge B} \qquad \frac{M : A}{\mathsf{inl}(M) : A \vee B} \qquad \frac{M : B}{\mathsf{inr}(M) : A \vee B} \qquad \frac{M : A \wedge B}{\mathsf{fst}(M) : A} \qquad \frac{M : A \wedge B}{\mathsf{snd}(M) : B}$$

$$\frac{M : A \vee B \qquad \begin{array}{c}\overline{u : A}^{\,u} \\ \vdots \\ \hline N : C\end{array} \qquad \begin{array}{c}\overline{v : B}^{\,v} \\ \vdots \\ \hline R : C\end{array}}{\mathsf{case}\ M\ \mathsf{of}\ \mathsf{inl}(c) \Rightarrow N \mid \mathsf{inl}(c) \Rightarrow R : C}\,u,v \qquad \frac{\begin{array}{c}\overline{u : A}^{\,u} \\ \vdots \\ \hline M : B\end{array}}{\mathsf{fn}\ u \Rightarrow M : A \supset B}\,u \qquad \frac{M : A \supset B \qquad N : A}{M(N) :}$$

2. Translation

We now turn to the question of translating proofs to programs and back again. In these notes, we present both for the sake of accessibility.

(1) $(A \supset B \supset C) \supset (B \supset A \supset C)$

Proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{A \supset B \supset C\ \text{true}}^{\,f} \qquad \overline{A\ \text{true}}^{\,a}}{B \supset C\ \text{true}} \qquad \overline{B\ \text{true}}^{\,b}}{C\ \text{true}}\supset E}{A \supset C\ \text{true}}\supset I^a}{B \supset A \supset C\ \text{true}}\supset I^b}{(A \supset B \supset C) \supset (B \supset A \supset C)\ \text{true}}\supset I^f$$

Program: `fn f => fn b => fn a => (f a) b`

Date: January 21, 2020.

[1] Of course, what makes this correspondence so remarkable is that it extends far beyond this one logic. It is quite robust and extends to almost any well-behaved logic. It also maps between logic and functional programming and lattices which are just closed cartesian categories

(2) $((A \supset B) \vee (A \supset C)) \supset A \supset (B \vee C)$

Proof:

$$
\dfrac{\dfrac{\dfrac{\dfrac{}{(A \supset B)\ \text{true} \vee (A \supset C)\ \text{true}}\,fg \qquad \dfrac{\dfrac{\dfrac{}{A \supset B\ \text{true}}\,f \quad \dfrac{}{A\ \text{true}}\,a}{B\ \text{true}}\supset E}{B \vee C\ \text{true}}\vee I_1 \qquad \dfrac{\dfrac{\dfrac{}{A \supset C\ \text{true}}\,g \quad \dfrac{}{A\ \text{true}}\,a}{C\ \text{true}}\supset E}{B \vee C\ \text{true}}\vee I_2}{B \vee C\ \text{true}}\vee E^{f,g}}{A \supset (B \vee C)\ \text{true}}\supset I^a}{((A \supset B) \vee (A \supset C)) \supset A \supset (B \vee C)\ \text{true}}\supset I^{fg}
$$

Program:

```
fn x => case snd x of inl b => inl (fst x, b) | inr c => inr (fst x, c)
```

3. Inventing proof terms

Let's consider a new connective $\curlywedge$. We'll give the intro and elim rules and try to come up with constructors, destructors and reduction rules that make sense.

$$
\dfrac{A\ \text{true} \qquad \begin{matrix}\dfrac{}{B\ \text{true}}\,u \\ \vdots \\ \hline \bot\ \text{true}\end{matrix}}{A \curlywedge B\ \text{true}}
$$

$$
\dfrac{\begin{matrix}\dfrac{}{B\ \text{true}}\,u \\ \vdots \\ \hline \bot\ \text{true}\end{matrix} \qquad A\ \text{true}}{A \curlywedge B\ \text{true}}
\qquad
\dfrac{A \curlywedge B\ \text{true} \qquad \begin{matrix}\dfrac{}{A\ \text{true}}\,u \quad \dfrac{}{\neg B\ \text{true}}\,v \\ \vdots \qquad\quad \vdots \\ \hline C\ \text{true}\end{matrix} \qquad \begin{matrix}\dfrac{}{\neg A\ \text{true}}\,u \quad \dfrac{}{B\ \text{true}}\,v \\ \vdots \qquad\quad \vdots \\ \hline C\ \text{true}\end{matrix}}{C\ \text{true}}
$$

Let's come up with constructors that make sense for $\curlywedge$

$$
\dfrac{M : A \qquad \begin{matrix}\dfrac{}{u : B}\,u \\ \vdots \\ \hline N : \bot\end{matrix}}{\mathsf{inl}(M, u.N) : A \curlywedge B}
\qquad\qquad
\dfrac{\begin{matrix}\dfrac{}{u : A}\,u \\ \vdots \\ \hline M : \bot\end{matrix} \qquad N : B}{\mathsf{inr}(u.M, N) : A \curlywedge B}
$$

And the destructor...

$$
\dfrac{E : A \curlywedge B \qquad \begin{matrix}\dfrac{}{u : A}\,u \quad \dfrac{}{v : \neg B}\,v \\ \vdots \qquad\quad \vdots \\ \hline M : C\end{matrix} \qquad \begin{matrix}\dfrac{}{u : \neg A}\,u \quad \dfrac{}{v : B}\,v \\ \vdots \qquad\quad \vdots \\ \hline N : C\end{matrix}}{\mathsf{case}\, E\, \mathsf{of}\, \mathsf{inl}(u, v) \Rightarrow M \mid \mathsf{inr}(u, v) \Rightarrow N : C}
$$

Now we still need to define a reduction rule for $\curlywedge$. Reduction rules are applied when the destructor is applied to a constructor.

$$
\mathsf{case}\, \mathsf{inl}(N', u'.M')\, \mathsf{of}\, \mathsf{inl}(u, v) \Rightarrow M \mid \mathsf{inr}(u, v) \Rightarrow N \Longrightarrow^r [N'/u, \mathsf{fn}\, u' \Rightarrow M'/v]M
$$

$$
\mathsf{case}\, \mathsf{inr}(u'.N', M')\, \mathsf{of}\, \mathsf{inl}(u, v) \Rightarrow M \mid \mathsf{inr}(u, v) \Rightarrow N \Longrightarrow^r [\mathsf{fn}\, u' \Rightarrow N'/u, M'/v]N
$$

4. Reductions

Let's try reducing a term until we can no longer appy reduction rules.

$$(\mathsf{fn}\, a \Rightarrow \mathsf{fn}\, b \Rightarrow (\mathsf{fn}\, f \Rightarrow \mathsf{fn}\, p \Rightarrow \langle \mathsf{fst}(f)\mathsf{fst}(p), \mathsf{snd}(f)\mathsf{snd}(p)\rangle)\langle \mathsf{fn}\, u \Rightarrow a, \mathsf{fn}\, u \Rightarrow b\rangle\langle b, a\rangle)$$

$$(\mathsf{fn}\, a \Rightarrow \mathsf{fn}\, b \Rightarrow (\mathsf{fn}\, p \Rightarrow \langle \mathsf{fst}(\langle \mathsf{fn}\, u \Rightarrow a, \mathsf{fn}\, u \Rightarrow b\rangle)\mathsf{fst}(p), \mathsf{snd}(\langle \mathsf{fn}\, u \Rightarrow a, \mathsf{fn}\, u \Rightarrow b\rangle)\mathsf{snd}(p)\rangle)\langle b, a\rangle)$$

Notice at this point we have a few options on how to proceed. It's actually the case that there is a term that we will reach no matter which order we apply reduction rules. It's generally know as the Church Rosser theorem that if a term finishes reducing in two ways, then they arrive at the same place. With our system we'll always reach a "normal" form after a finite number of reductions, so we can apply rules in what ever order we wish.

$$(\mathsf{fn}\, a \Rightarrow \mathsf{fn}\, b \Rightarrow (\mathsf{fn}\, p \Rightarrow \langle (\mathsf{fn}\, u \Rightarrow a)\mathsf{fst}(p), \mathsf{snd}(\langle \mathsf{fn}\, u \Rightarrow a, \mathsf{fn}\, u \Rightarrow b\rangle)\mathsf{snd}(p)\rangle)\langle b, a\rangle)$$

$$(\mathsf{fn}\, a \Rightarrow \mathsf{fn}\, b \Rightarrow (\mathsf{fn}\, p \Rightarrow \langle (\mathsf{fn}\, u \Rightarrow a)\mathsf{fst}(p), (\mathsf{fn}\, u \Rightarrow b)\mathsf{snd}(p)\rangle)\langle b, a\rangle)$$

$$(\mathsf{fn}\, a \Rightarrow \mathsf{fn}\, b \Rightarrow (\mathsf{fn}\, p \Rightarrow \langle a, (\mathsf{fn}\, u \Rightarrow b)\mathsf{snd}(p)\rangle)\langle b, a\rangle)$$

$$(\mathsf{fn}\, a \Rightarrow \mathsf{fn}\, b \Rightarrow (\mathsf{fn}\, p \Rightarrow \langle a, b\rangle)\langle b, a\rangle)$$
$$(\mathsf{fn}\, a \Rightarrow \mathsf{fn}\, b \Rightarrow \langle a, b\rangle)$$