

## Comp 411 Notes - Wed, Feb 9, 2005

### The Curry-Howard Isomorphism

Scribe: Eric Cheng  
(ericc@rice.edu)

## 1 Overview

In this class, we discussed the formulation of constructive logic and linearization of a proof tree. We also saw some of the correspondence between the typing rules of the lambda calculus and constructive logics, known as the *Curry-Howard correspondence* (Curry and Feys, 1958; Howard, 1980).

## 2 Types and Constructive Logic

Recall the definition of types:

$$t ::= b \mid t \rightarrow t$$

The whole thing is called an atomic proposition. Consider the BNF of constructive logic:

$$P ::= a \mid P \Rightarrow P \mid P \wedge P \mid P \vee P \mid x \mid \forall x.P \mid \exists x.P$$

What is similar between them?

Let's look at the sequent calculus formulation:

$$\frac{}{H \vdash T} (1) \quad \frac{P \in H}{H \vdash P} (2) \quad \frac{H, P_1 \vdash P_2}{H \vdash P_1 \Rightarrow P_2} (3)$$

$$\frac{H \vdash P_1 \Rightarrow P_2 \quad H \vdash P_1}{H \vdash P_2} (4) (\textit{modus ponens}) \quad \frac{H \vdash P_1 \quad H \vdash P_2}{H \vdash P_1 \wedge P_2} (5)$$

Each  $H$  denotes a sequence of propositions.

In the types derivation for the lambda calculus, the terms capture the form of the tree. By just looking at the lambda term on the bottom, we can tell exactly what rules would be used to derive the proof. Is it the same case for our formulation above? Consider the second rule (2) and the modus ponens rule (4) above. By solely looking at the judgements, one cannot tell which rule was used during the derivation. We want a way to linearize a tree, while still capturing the individual rules used throughout the proof.

We rewrite each rule by labeling each proposition as follows:

$$\frac{}{H \vdash 1 : T} \quad (1) \quad \frac{\ell : P \in H}{H \vdash \ell : P} \quad (2) \quad \frac{H, \ell : P_1 \vdash e : P_2}{H \vdash (\ell, e) : P_1 \Rightarrow P_2} \quad (3)$$

$$\frac{H \vdash e_1 : P_1 \Rightarrow P_2 \quad H \vdash e_2 : P_1}{H \vdash e_1 @ e_2 : P_2} \quad (4) \quad \frac{H \vdash e_1 : P_1 \quad H \vdash e_2 : P_2}{H \vdash (e_1, e_2) : P_1 \wedge P_2} \quad (5)$$

This does exactly what we want! Now we can just show the judgement to people and they would be able to know how we got the result.

Note that these rules look very similar to the typing rules for the lambda calculus. Specifically, (1) is similar to the type axioms, (2) looks like the variable rule, (3) looks like the abstraction rule, and (4) looks like the application rule. We will discuss (5) next time. This is called the Curry-Howard isomorphism.

**Examples of tree linearization:**

1.

$$\frac{}{H, T \vdash T} \quad \text{could be rewritten as} \quad \frac{}{H, x : T \vdash 1 : T}$$

$$\frac{T \in H, T}{H, T \vdash T} \quad \text{could be rewritten as} \quad \frac{x : T \in H, x : T}{H, x : T \vdash x : T}$$

2.

$$G \equiv \begin{array}{ll} RED \Rightarrow HOT, & (x) \\ RED, & (y) \\ HOT \Rightarrow DANGEROUS & (z) \\ RED \Rightarrow DANGEROUS & (a) \end{array}$$

Prove that  $\frac{}{H \vdash DANGEROUS}$

There is more than one way to reach the judgement. As we will see, with the labeling technique, we could easily tell which rules were used during the derivation.

$$(4) \quad \frac{(2) \frac{y : R \in G}{G \vdash y : R} \quad (2) \frac{a : R \Rightarrow D \in G}{G \vdash a : R \Rightarrow D}}{G \vdash a @ y : D}$$

$$(4) \quad \frac{(2) \frac{R \in G}{G \vdash y : R} \quad (2) \frac{R \Rightarrow H \in G}{G \vdash x : R \Rightarrow H} \quad (2) \frac{H \Rightarrow D \in G}{G \vdash z : H \in D}}{G \vdash z @ (x @ y) : D}$$

**Exercise:** Prove that

$$\overline{G \vdash RED \Rightarrow DANGEROUS}$$

using both (x) and (z), but not (y) or (a).