15-399 Supplementary Notes: Mid-Course Review

Robert Harper

February 23, 2004

Judgments and Evidence

A *judgment* is an assertion of knowledge about some subject matter. Examples include “it is snowing” or “it is true that every prime other than 2 is odd”. A judgment is *correct*, or *evident*, if there is evidence for it. What counts as evidence depends on the form of judgment. Evidence for “it is snowing” might be some form of direct observation; evidence for “it is true that every prime number other than 2 is odd” is a proof.

An *analytic* judgment is self-evident; it requires no further information to assess its correctness. A *synthetic* judgment requires external evidence to establish its correctness. The judgment “every prime other than 2 is odd expresses a proposition” is analytic; we need only look at the purported proposition to see that it does in fact express one. The judgments “it is snowing” and “it is true that every prime other than 2 is odd” are synthetic; evidence for the former consists of sense data, evidence for the latter is a proof.

A *categorical* judgment is unconditional. Logic is concerned with the categorical judgments $P\,\mathit{prop}$ stating that P expresses a proposition, $P\,\mathit{true}$ stating that P is a true proposition, and $M : P$ stating that M is a proof of proposition P.

A *hypothetical* judgment is an assertion made under the assumption of evidence for the truth of some specified set of propositions. A hypothetical judgment has the form $J_1, \ldots, J_n \vdash J$, where the *antecedents* are $J_1, \ldots, J_n$, and the *consequent* is J. Evidence for a hypothetical judgment consists of evidence for J built up from the presumed evidence for each J_i $(1 \leq i \leq n)$.

Classical and Constructive Semantics

Classical logic is the logic of *reference*, or *denotation*. The meaning of a proposition is a truth value, either **true** or **false**. Consequently, the *law of the excluded middle*, $P \vee \neg P$, is true for any proposition P. Since negation swaps truth values, the *law of double negation elimination*, $\neg\neg P \supset P$, is true for every proposition P.

Constructive logic is the logic of *sense*, or *knowledge*. The meaning of a proposition P is a problem to be solved; the solution consists of a proof. The judgment $P\ \mathit{true}$ asserts that we have evidence for P, namely a proof of it. Since there are unsolved problems, we do not always have evidence for either $P\ \mathit{true}$ or $\neg P\ \mathit{true}$. Consequently the law of the excluded middle does not hold in general, nor does the law of double negation elimination. A proposition P for which $P \vee \neg P\ \mathit{true}$ is said to be *decidable*. One for which double negation elimination is valid is said to be *stable*. Every decidable proposition is stable; if all propositions are stable, then all propositions are decidable.

The categorical judgment $P\ \mathit{true}$ is synthetic, because evidence for it consists of a proof that is not itself part of the judgment. If we include this evidence in the judgment, writing $M : P$ to assert that M is a proof of P, then the judgment is analytic. The judgment $P\ \mathit{true}$ means that $M : P$ for some proof term M — the term M is the external evidence required for the truth of P. The hypothetical judgment $P_1\ \mathit{true}, \ldots, P_n\ \mathit{true} \vdash P\ \mathit{true}$ is synthetic because it asserts the existence of a proof of P from the assumptions that we know a proof of each P_i. On the other hand the hypothetical judgment $u_1{:}P_1, \ldots, u_n{:}P_n \vdash M : P$ is analytic, because it contains the evidence, M, for P, which may involve the presumed evidence u_i for each P_i.

The hypothetical judgment in constructive logic obeys the following rules:

- Identity: $\Gamma, u{:}P \vdash u : P$.
- Substitution: If $\Gamma \vdash M : P$ and $\Gamma, u{:}P \vdash N : Q$, then $\Gamma \vdash [M/u]N : Q$.
- Weakening: If $\Gamma \vdash M : P$, then $\Gamma, u{:}Q \vdash M : P$.
- Contraction: If $\Gamma, u{:}P, v{:}P \vdash M : Q$, then $\Gamma \vdash u{:}P \vdash [u/v]M : Q$.

Propositional Logic

The rules of logic codify legitimate forms of evidence for the truth of propositions. These rules are organized according to the logical connective used to form the proposition. For each connective there are three classes of rules that determine its meaning. The *formation* rules determine the conditions for forming a proposition from that connective. The *introduction* rules for a connective define the primary, or canonical, forms of evidence for judging a proposition formed with that connective to be true. The *elimination* rules for a connective invert the corresponding introduction rules for that connective by extracting the canonical evidence from the truth of a proposition formed with that connective.

The rules of propositional logic govern the connectives $\top$, $\bot$, $\wedge$, $\supset$, $\vee$, $\neg$.

Truth

$$\frac{}{\top\ \mathit{prop}} \qquad \frac{}{\Gamma \vdash \langle\,\rangle : \top} \qquad \textit{(no elim rule)}$$

Conjunction

$$\frac{P\ \mathit{prop} \quad Q\ \mathit{prop}}{P \wedge Q\ \mathit{prop}} \qquad \frac{\Gamma \vdash M : P \quad \Gamma \vdash N : Q}{\Gamma \vdash \langle M, N\rangle : P \wedge Q}$$

$$\frac{\Gamma \vdash M : P \wedge Q}{\Gamma \vdash \mathbf{fst}(M) : P} \qquad \frac{\Gamma \vdash N : P \wedge Q}{\Gamma \vdash \mathbf{snd}(M) : Q}$$

Implication

$$\frac{P\ \mathit{prop} \quad Q\ \mathit{prop}}{P \supset Q\ \mathit{prop}} \qquad \frac{\Gamma, u{:}P \vdash M : Q}{\Gamma \vdash \lambda u{:}P.M : P \supset Q}$$

$$\frac{\Gamma \vdash M : P \supset Q \quad \Gamma \vdash N : P}{\Gamma \vdash M\,N : Q}$$

Falsity

$$\frac{}{\bot\ \mathit{true}} \qquad \textit{(no introduction rule)} \qquad \frac{\Gamma \vdash M : \bot}{\Gamma \vdash \mathbf{abort}_P(M) : P}$$

Disjunction

$$\frac{P\ \mathit{prop} \quad Q\ \mathit{prop}}{P \vee Q\ \mathit{prop}} \qquad \frac{\Gamma \vdash M : P}{\Gamma \vdash \mathbf{inl}_{P,Q}(M) : P \vee Q} \qquad \frac{\Gamma \vdash N : Q}{\Gamma \vdash \mathbf{inr}_{P,Q}(N) : P \vee Q}$$

$$\frac{\Gamma \vdash M : P \vee Q \quad \Gamma, u{:}P \vdash M_1 : R \quad \Gamma, v{:}Q \vdash M_2 : R}{\Gamma \vdash \mathbf{case}\, M\ \ \mathbf{of}\ \ \mathbf{inl}(u : P) \Rightarrow M_1 \mid \mathbf{inr}(v : Q) \Rightarrow M_2 : R}$$

Simplification rules state that the elimination constructs are inverse to the introduction constructs. These may be stated as axioms of *definitional equality*, written $M \equiv N$.[1]

Definitional equality is the least congruence closed under these axioms:

$$\frac{}{\mathbf{fst}(\langle M, N\rangle) \equiv M} \qquad \frac{}{\mathbf{snd}(\langle M, N\rangle) \equiv N}$$

$$\frac{}{(\lambda u{:}P.M)\,N \equiv [N/u]M}$$

$$\frac{}{\mathbf{case}\,\mathbf{inl}(M)\ \ \mathbf{of}\ \ \mathbf{inl}(u{:}P) \Rightarrow M_1 \mid \mathbf{inr}(v{:}Q) \Rightarrow M_2 \equiv [M/u]M_1}$$

$$\frac{}{\mathbf{case}\,\mathbf{inr}(M)\ \ \mathbf{of}\ \ \mathbf{inl}(u{:}P) \Rightarrow M_1 \mid \mathbf{inr}(v{:}Q) \Rightarrow M_2 \equiv [M/v]M_2}$$

To say that definitional equality is a congruence is to say that in addition to these rules we have rules stating that it is an equivalence relation (reflexive, symmetric, and transitive) and that we may "replace equals by equals" anywhere within a term.

[1]The Pfenning notes uses $M \Leftrightarrow N$ for this.

Data Types

Logic is embedded in the larger framework of *type theory*. Type theory is concerned with the categorical judgments τ *type* stating that τ is a type, and $t \in \tau$ stating that t is a term of type τ. Data types include basic types such as the natural numbers, and complex types such as Cartesian product types, function types, and disjoint union types. Types are defined in much the same manner as are propositions. This is no accident: it will emerge that propositions are just certain forms of type.

In the presence of data types we have a new form of judgment, called the *general judgment*. It has the same form as the hypothetical judgment, except that the antecedent can involve assumptions of the form $x \in \tau$, where x is a term variable and τ ia a type. A *hypothetico-general judgment* is one that intermixes the two forms of assumption: $u{:}P$ and $x \in \tau$. The hypothetico-general judgment obeys the same sort of structural rules as does the hypothetical judgment in logic: identity, substitution, weakening, and contraction.

The rules of type theory govern the "logical" type constructors $\mathbf{1}$, $\times$, $\rightarrow$, $\mathbf{0}$, $+$, and "data" type **nat**. The logical type constructors correspond to the propositional connectives, whereas the data type constructors do not. This distinction, however, is only suggestive; there is no fundamental difference between the two.

Unit

$$\frac{}{\mathbf{1}\ \mathit{type}} \qquad \frac{}{\Gamma \vdash \langle\,\rangle \in \mathbf{1}} \qquad \textit{(no elim rule)}$$

Product

$$\frac{\tau_1\ \mathit{type} \quad \tau_2\ \mathit{type}}{\tau_1 \times \tau_2\ \mathit{type}} \qquad \frac{\Gamma \vdash t_1 \in \tau_1 \quad \Gamma \vdash t_2 \in \tau_2}{\Gamma \vdash \langle t_1, t_2 \rangle \in \tau_1 \times \tau_2}$$

$$\frac{\Gamma \vdash t \in \tau_1 \times \tau_2}{\Gamma \vdash \mathbf{fst}(t) \in \tau_1} \qquad \frac{\Gamma \vdash t \in \tau_1 \times \tau_2}{\Gamma \vdash \mathbf{snd}(t) \in \tau_2}$$

Function

$$\frac{\tau_1\ \mathit{type} \quad \tau_2\ \mathit{type}}{\tau_1 \rightarrow \tau_2\ \mathit{type}} \qquad \frac{\Gamma, x \in \tau_1 \vdash t \in \tau_2}{\Gamma \vdash \lambda x \in \tau_1 . t \in \tau_1 \rightarrow \tau_2}$$

$$\frac{\Gamma \vdash t_1 \in \tau_2 \rightarrow \tau \quad \Gamma \vdash t_2 \in \tau_2}{\Gamma \vdash t_1\, t_2 \in \tau}$$

Void

$$\frac{}{\mathbf{0}\ \mathit{type}} \qquad \textit{(no intro)} \qquad \frac{\Gamma \vdash t \in \mathbf{0}}{\Gamma \vdash \mathbf{abort}_\tau(t) \in \tau}$$

Sum

$$\frac{\tau_1 \ \mathit{type} \quad \tau_2 \ \mathit{type}}{\tau_1 + \tau_2 \ \mathit{type}}$$

$$\frac{\Gamma \vdash t \in \tau_1}{\Gamma \vdash \mathbf{inl}_{\tau_1,\tau_2}(t) \in \tau_1 + \tau_2} \qquad \frac{\Gamma \vdash t \in \tau_2}{\Gamma \vdash \mathbf{inr}_{\tau_1,\tau_2}(t) \in \tau_1 + \tau_2}$$

$$\frac{\Gamma \vdash t \in \tau_1 + \tau_2 \quad \Gamma, x \in \tau_1 \vdash t_1 \in \tau \quad \Gamma, y \in \tau_2 \vdash t_2 \in \tau}{\Gamma \vdash \mathbf{case}\, t \ \mathbf{of} \ \mathbf{inl}(x \in \tau_1) \Rightarrow t_1 \mid \mathbf{inr}(y \in \tau_2) \Rightarrow t_2 \in \tau}$$

Natural Numbers

$$\frac{}{\mathbf{nat} \ \mathit{type}}$$

$$\frac{}{\Gamma \vdash \mathbf{0} \in \mathbf{nat}} \qquad \frac{\Gamma \vdash t \in \mathbf{nat}}{\Gamma \vdash \mathbf{s}(t) \in \mathbf{nat}}$$

$$\frac{\Gamma \vdash t \in \mathbf{nat} \quad \Gamma \vdash t_1 \in \tau \quad \Gamma, x \in \mathbf{nat}, f(x) \in \tau \vdash t_2 \in \tau}{\Gamma \vdash \mathbf{rec}\, t \ \mathbf{of} \ f(\mathbf{0}) \Rightarrow t_1 \mid f(\mathbf{s}(x \in \mathbf{nat})) \Rightarrow t_2 \in \tau}$$

The simplification rules for the logical type constructors are just the same as those for proof terms. The simplification rules for primitive recursion are as follows:

$$\frac{}{\mathbf{rec}\,\mathbf{0} \ \mathbf{of} \ f(\mathbf{0}) \Rightarrow t_1 \mid f(\mathbf{s}(x \in \mathbf{nat})) \Rightarrow t_2 \in \tau \equiv t_1}$$

$$\frac{}{\mathbf{rec}\,\mathbf{s}(t) \ \mathbf{of} \ f(\mathbf{0}) \Rightarrow t_1 \mid f(\mathbf{s}(x \in \mathbf{nat})) \Rightarrow t_2 \in \tau \equiv [t/x, (*)/f(x)]t_2}$$

where $(*)$ stands for the entire primitive recursion term on the left-hand side of these two equations.

We write $t \equiv u$ to mean that t and u are definitionally equivalent, and write $\Gamma \vdash t \equiv u \in \tau$ to mean that $\Gamma \vdash t \in \tau$, $\Gamma \vdash u \in \tau$, and $t \equiv u$.

The Curry-Howard Isomorphism

There is an evident similarity between propositions and types, and between proof terms for a proposition and the elements of the corresponding type. This correspondence may be made explicit by defining a type, $\mathbf{pfs}(P)$, of the proof terms for the proposition P:

$$\begin{array}{rcl}
\mathbf{pfs}(\top) & \equiv & \mathbf{1} \\
\mathbf{pfs}(\bot) & \equiv & \mathbf{0} \\
\mathbf{pfs}(P \wedge Q) & \equiv & \mathbf{pfs}(P) \times \mathbf{pfs}(Q) \\
\mathbf{pfs}(P \vee Q) & \equiv & \mathbf{pfs}(P) + \mathbf{pfs}(Q) \\
\mathbf{pfs}(P \supset Q) & \equiv & \mathbf{pfs}(P) \to \mathbf{pfs}(Q)
\end{array}$$

With these definitions, it is easy to see that the judgment, $M : P$, stating that M is a proof of P, may be regarded as an abbreviation for the judgment $M \in \mathbf{pfs}(P)$. The following rules are derivable, given the definitional equalities governing the types $\mathbf{pfs}(P)$:

$$\frac{\Gamma \vdash P\,\mathit{prop}}{\Gamma \vdash \mathbf{pfs}(P)\,\mathit{type}} \qquad \frac{\Gamma \vdash P \equiv Q\,\mathit{prop}}{\Gamma \vdash \mathbf{pfs}(P) \equiv \mathbf{pfs}(Q)}$$

The correspondence between propositions and types, and between proofs of propositions and elements of their associated types, is called the *Curry-Howard Isomorphism*, or the *propositions-as-types principle.* Over time we will do away with the separation between propositions and types, and simply identify each proposition with the type of its proofs. In this way logic is reduced to type theory — or, put more strikingly, mathematics is reduced to programming!

Predicate Logic

A *predicate* over a type τ expresses a proposition about the elements τ. (A *binary relation* is simply a predicate over a product type.) The formation rule for a predicate p over τ has the form

$$\frac{\Gamma \vdash t \in \tau}{\Gamma \vdash p(t)\,\mathit{prop}}$$

For example, the equality predicate over the type $\mathbf{nat} \times \mathbf{nat}$ has the formation rule

$$\frac{\Gamma \vdash t \in \mathbf{nat} \quad \Gamma \vdash u \in \mathbf{nat}}{\Gamma \vdash t =_N u\,\mathit{prop}}\,.$$

Since predicates give rise to propositions that contain terms, we must generalize the judgment $P\,\mathbf{prop}$ to include hypotheses governing the variables that may occur within it. For example,

$$x \in \mathbf{nat}, y \in \mathbf{nat} \vdash x =_N y\,\mathit{prop}$$

is a correct hypothetical judgment stating that $x =_N y$ expresses a proposition for any $x \in \mathbf{nat}$ and any $y \in \mathbf{nat}$.

The occurrence of terms within propositions induces a notion of definitional equality of propositions. If p is a predicate over τ, then we have

$$\frac{\Gamma \vdash t \equiv u \in \mathbf{nat}}{\Gamma \vdash p(t) \equiv p(u)\,\mathit{prop}}$$

Definitionally equivalent propositions are indistinguishable from one another. In particular, we have

$$\frac{\Gamma \vdash M : P \quad \Gamma \vdash P \equiv Q\,\mathit{prop}}{\Gamma \vdash M : Q}\,.$$

Suppressing proof terms, this means

$$\frac{\Gamma \vdash P\,\mathit{true} \quad \Gamma \vdash P \equiv Q}{\Gamma \vdash Q\,\mathit{true}}\ .$$

For example, since $2 + 2 \equiv 3 + 1 \in \mathbf{nat}$, we have $2 + 2 =_N 3 + 1 \equiv 4 =_N 4\,\mathit{prop}$ — they express the very same proposition.

The primitive recursion construct is the elimination rule for **nat** for constructing members of types from a natural number. The principle of *mathematical induction* is the elimination rule for **nat** for constructing proofs of a proposition for any natural number. It is stated as follows:

$$\frac{\begin{array}{cc} \Gamma \vdash t \in \mathbf{nat} & \Gamma, x \in \mathbf{nat} \vdash P\,\mathit{prop} \\ \Gamma \vdash M_1 : [0/x]P & \Gamma, x \in \mathbf{nat}, u(x) : P \vdash M_2 : [\mathbf{s}(x)/x]P \end{array}}{\Gamma \vdash \mathbf{ind}\,t\ \mathbf{of}\ u(\mathbf{0}) \Rightarrow M_1 \mid u(\mathbf{s}(x \in \mathbf{nat})) \Rightarrow M_2 : [t/x]P}$$

There is an evident similarity with the rule for primitive recursion, the main difference being that the proposition includes the natural number for which the proof term provides a proof.

The simplification rules are as follows:

$$\frac{}{\mathbf{ind}\,\mathbf{0}\ \mathbf{of}\ u(\mathbf{0}) \Rightarrow M_1 \mid u(\mathbf{s}(x \in \mathbf{nat})) \Rightarrow M_2 : P \equiv M_1}$$

$$\frac{}{\mathbf{ind}\,\mathbf{s}(t)\ \mathbf{of}\ u(\mathbf{0}) \Rightarrow M_1 \mid u(\mathbf{s}(x \in \mathbf{nat})) \Rightarrow M_2 : P \equiv [t/x, (*)/u(x)]M_2}$$

Here again there is an evident similarity with primitive recursion. Primitive recursion is the computational content of an inductive proof!

Quantification

The *quantifiers* are used to express generality and existence. The proposition $\forall x \in \tau.P$ means that $[t/x]P$ holds for every $t \in \tau$. The proposition $\exists x \in \tau.P$ means that $[t/x]P$ holds for some $t \in \tau$.

Universal

$$\frac{\Gamma, x \in \tau \vdash P\,\mathit{prop} \quad (x\ \mathit{fresh})}{\Gamma \vdash \forall x \in \tau.P\,\mathit{prop}} \qquad \frac{\Gamma, x \in \tau \vdash M : P \quad (x\ \mathit{fresh})}{\Gamma \vdash \lambda x \in \tau.M : \forall x \in \tau.P}$$

$$\frac{\Gamma \vdash M : \forall x \in \tau.P \quad \Gamma \vdash t \in \tau}{\Gamma \vdash M\,t \in [t/x]P}$$

The simplification rule is as follows:

$$(\lambda x \in \tau.M)\,t \equiv [t/x]M$$

Existential

$$\frac{\Gamma, x \in \tau \vdash P\ \mathit{prop} \quad (x\ \mathit{fresh})}{\Gamma \vdash \exists x \in \tau . P\ \mathit{prop}} \qquad \frac{\Gamma \vdash t \in \tau \quad \Gamma \vdash M : [t/x]P}{\Gamma \vdash (M, t) \in \exists x \in \tau . P}$$

$$\frac{\Gamma \vdash M : \exists x \in \tau . P \quad \Gamma, x \in \tau, u \in P \vdash N : Q}{\Gamma \vdash \mathbf{let}\, (x, u) = M \ \mathbf{in}\ N : Q}$$

The simplification rule is as follows:

$$\mathbf{let}\, (x, u) = (t, M) \ \mathbf{in}\ N \equiv [t/x, M/u]N.$$

Things To Come

Here is a summary of the main developments to come:

1. Introduction of *families of types*, or *dependent types*, that correspond to predicates, leading to a consolidation of propositions and types.

2. Introduction of a type of propositions, leading to *higher-order logic* and the elimination of connectives definable using higher-order quantifiers.