Logical Preliminaries

Johannes C. Flieger
Scheme UK

March 2003

Abstract

Survey of intuitionistic and classical propositional logic; introduction to the computational interpretation of intuitionistic logic in the form of the Curry-Howard Isomorphism.

1 Introduction

These are the notes for the first of a series of brief, informal talks on resource logics, functional programming, and sundry related matters, held under the auspices of Scheme UK. The aim of these talks is to impart enough working knowledge of basic concepts, terminology, and notation to approach the literature with a modicum of confidence, and to impress relatives at family gatherings.

This first session covers certain elementary logical notions, and is intended to be a guerilla introduction to logic for those relatively new to it, as well as a refresher for those who have blocked out the traumatic memory of their first encounter with the subject. For those familiar with the material, it will serve to fix terminology and notation for future sessions.

2 Propositional Logic

Definition 1 (Deductive system) *A deductive system $\mathfrak{S}$ is a pair*

$$\mathfrak{S} = \langle \mathfrak{L}, \mathfrak{R} \rangle$$

such that

- $\mathfrak{L}$ *is a formal language consisting of countably many formulae, and*
- $\mathfrak{R}$ *is a set of rules of inference.*

2.1 Language

We shall base our language on a symbolic repertoire comprising three kinds of symbols:

Propositional Parameters: $p, q, r, \ldots, p_1, q_1, r_1, \ldots$

Propositional Constants: $\bot$

Propositional Connectives: $\rightarrow, \wedge, \vee$

We employ lowercase Greek letters $\alpha, \beta, \gamma, \ldots, \alpha_1, \beta_1, \gamma_1, \ldots$ as metavariables over formulae, and Greek capitals $\Gamma, \Delta, \Sigma, \ldots, \Gamma_1, \Gamma_2, \Sigma_1, \ldots$ to denote (finite) *sequences* of formulae. The notation Γ, α denotes the result of appending one more occurrence of α to the sequence Γ, and Γ, Δ denotes the result of appending the sequence Δ to the sequence Γ.

We construct propositional formulae inductively as follows:

Definition 2 (Formulae of Propositional Logic)

1. *All propositional parameters and propositional constants are formulae.*
2. *If α is a formula and β is a formula, then $(\alpha \Box \beta)$ is a formula, for $\Box \in \{\wedge, \vee, \rightarrow\}$.*
3. *Nothing else is a formula.*

Other connectives and the constant $\top$ are defined as abbreviations:

$$\begin{aligned} \neg\alpha &\stackrel{\text{def}}{=} (\alpha \rightarrow \bot) \\ \top &\stackrel{\text{def}}{=} \neg\bot \\ (\alpha \leftrightarrow \beta) &\stackrel{\text{def}}{=} ((\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)) \end{aligned}$$

2.2 Inference Rules

Definition 3 (Sequent) *A* sequent *has the form $\Gamma \vdash \varphi$, and is read as 'The truth of φ follows from the truth of Γ', or 'From Γ we can conclude that φ', or simply 'Γ entails φ'.*

A *derivation* is a rooted tree whose nodes are labelled by judgements. A *rule* consists of zero or more judgements written above a line, and a single judgement below it; if all the judgements above the line are derivable, then the judgement below is also derivable.

2.2.1 Axioms

Definition 4 (Id)

$$\frac{}{\alpha \vdash \alpha}\ Id$$

2.2.2 Rules for Logical Constants and Connectives

Definition 5 (∧I)

$$\frac{\Gamma \vdash \alpha \quad \Delta \vdash \beta}{\Gamma, \Delta \vdash (\alpha \wedge \beta)}\ \wedge I$$

Definition 6 (∧E)

1.

$$\frac{\Gamma \vdash (\alpha \wedge \beta)}{\Gamma \vdash \alpha}\ \wedge E$$

2.

$$\frac{\Gamma \vdash (\alpha \wedge \beta)}{\Gamma \vdash \beta}\ \wedge E$$

Definition 7 (∨I)

1.

$$\frac{\Gamma \vdash \alpha}{\Gamma \vdash (\alpha \vee \beta)}\ \vee I$$

2.

$$\frac{\Gamma \vdash \alpha}{\Gamma \vdash (\beta \vee \alpha)}\ \vee I$$

Definition 8 (∨E)

$$\frac{\Gamma \vdash (\alpha \vee \beta) \quad \Delta \vdash (\alpha \to \theta) \quad \Sigma \vdash (\beta \to \theta)}{\Gamma, \Delta, \Sigma \vdash \theta}\ \vee E$$

Definition 9 (→I)

$$\frac{\Gamma, \alpha \vdash \beta}{\Gamma \vdash \alpha \to \beta}\ \to I$$

Definition 10 (→E)

$$\frac{\Gamma \vdash (\alpha \to \beta) \quad \Delta \vdash \alpha}{\Gamma, \Delta \vdash \beta} \to E$$

Definition 11 (EFSQ)

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash \alpha} \; EFSQ$$

2.2.3 Structural Rules

Definition 12 (Exchange)

$$\frac{\Gamma, \alpha \vdash \beta}{\alpha, \Gamma \vdash \beta} \; Exchange$$

Definition 13 (Weakening)

$$\frac{\Gamma \vdash \beta}{\Gamma, \alpha \vdash \beta} \; Weakening$$

Definition 14 (Contraction)

$$\frac{\Gamma, \alpha, \alpha \vdash \beta}{\Gamma, \alpha \vdash \beta} \; Contraction$$

- *Exchange* expresses that the order of the premises is irrelevant.
- *Weakening* expresses that any premise may be discarded.
- *Contraction* expresses that any premise can be duplicated.

2.2.4 Alternative Rules for Conjunction and Disjunction Elimination

There are alternative formulations of several of the rules of inference given above. For example, an alternative rule of conjunction elimination can be derived from the version of ∧**E** given above, using the structural rules of Weakening and Exchange:

Definition 15 (∧E)

$$\frac{\Gamma \vdash (\alpha \wedge \beta) \quad \Delta, \alpha, \beta \vdash \theta}{\Gamma, \Delta \vdash \theta} \wedge E$$

An alternative rule of disjunction elimination can be derived from the version of $\vee$**E** given above:

Definition 16 ($\vee$E)

$$\frac{\Gamma \vdash (\alpha \vee \beta) \quad \Delta, \alpha \vdash \theta \quad \Sigma, \beta \vdash \theta}{\Gamma, \Delta, \Sigma \vdash \theta} \; {\scriptstyle \vee E}$$

3 From Intuitionistic to Classical Logic

The addition of any of the following (interderivable) rules to our intuitionistic system yields a system of classical propositional logic:

Definition 17 (Law of Excluded Middle)

$$\frac{}{\vdash \alpha \vee \neg\alpha} \; {\scriptstyle EM}$$

Definition 18 (Dilemma)

$$\frac{\Gamma, \alpha \vdash \theta \quad \Delta, \neg\alpha \vdash \theta}{\Gamma, \Delta \vdash \theta} \; {\scriptstyle D}$$

Definition 19 (Reductio ad Absurdum—RAA)

$$\frac{\Gamma, \neg\alpha \vdash \bot}{\Gamma \vdash \alpha} \; {\scriptstyle RAA}$$

Definition 20 (Double Negation)

$$\frac{\Gamma \vdash \neg\neg\alpha}{\Gamma \vdash \alpha} \; {\scriptstyle DN}$$

4 Proofs and Programs

Intuitionistic logic may be viewed as that part of classical logic that admits of an *effective* interpretation, and which exhibits a correspondence between proofs and programs.

We can obtain a more exact analysis of proofs by treating them as first-class entities. To that end, we introduce *proof objects* into our derivations. We denote proof objects by *terms* which encode the structure of the proof. In this system, a statement is a pair $x : \varphi$ of a term x and a (propositional) formula φ, and is read as 'x is a proof of proposition φ'.

4.1 The Language of Terms

It turns out that a language based on the (simply-typed) λ-calculus is particularly well-suited to representing the structure of proof objects. We assume the availability of countably many variables $x_1, x_2, x_3, \ldots$ which may be used to refer to proofs of any proposition φ; the proposition φ can be thought of as the *type* of its proofs. In order to keep track of the type of the proof referred to by the variable, we will sometimes explicitly write the type as a superscript: $x_1^\varphi, x_2^\varphi, x_3^\varphi, \ldots$

Let *u,v,...,y,z* range over variables and *a,b,c,...,s,t,* range over terms. Then the language of terms is defined by

$$t ::= x \mid \lambda x.f \mid (f\ g) \mid (f, g) \mid (\mathrm{inl}\ f) \mid (\mathrm{inr}\ f) \mid (\mathrm{cases}\ f\ g\ h) \mid (\mathrm{abort}\ f)$$

Premises are written as

$$x_1 : \varphi_1, \ldots, x_n : \varphi_n$$

where $x_1, \ldots, x_n$ are variables, $\varphi_1, \ldots, \varphi_n$ are propositions, and $n \geq 0$. We shall require that the variables $x_1, \ldots, x_n$ in a sequence be *distinct*; note that this extends also to concatenations of premises, so that in a concatenation Γ, Δ, we shall require that Γ and Δ contain distinct variables.

4.2 Inference Rules Revisited

4.2.1 Axioms

Definition 21 (Id)

$$\frac{}{x : \alpha \vdash x : \alpha}\ Id$$

4.2.2 Rules for Logical Constants and Connectives

Definition 22 ($\wedge$I)

$$\frac{\Gamma \vdash s : \alpha \qquad \Delta \vdash t : \beta}{\Gamma, \Delta \vdash (s, t) : (\alpha \wedge \beta)}\ \wedge I$$

Definition 23 ($\wedge$E)

1.

$$\frac{\Gamma \vdash t : (\alpha \wedge \beta)}{\Gamma \vdash (\mathrm{fst}\ t) : \alpha}\ \wedge E$$

2.

$$\frac{\Gamma \vdash t : (\alpha \wedge \beta)}{\Gamma \vdash (\text{snd}\ \ t) : \beta}\ {\scriptstyle \wedge E}$$

Definition 24 (∨I)

1.

$$\frac{\Gamma \vdash t : \alpha}{\Gamma \vdash (\text{inl}\ \ t) : (\alpha \vee \beta)}\ {\scriptstyle \vee I}$$

2.

$$\frac{\Gamma \vdash t : \alpha}{\Gamma \vdash (\text{inr}\ \ t) : (\beta \vee \alpha)}\ {\scriptstyle \vee I}$$

Definition 25 (∨E)

$$\frac{\Gamma \vdash t : (\alpha \vee \beta) \quad \Delta \vdash f : (\alpha \rightarrow \theta) \quad \Sigma \vdash g : (\beta \rightarrow \theta)}{\Gamma, \Delta, \Sigma \vdash (\text{cases}\ \ t\ \ f\ \ g) : \theta}\ {\scriptstyle \vee E}$$

Definition 26 (→I)

$$\frac{\Gamma, x : \alpha \vdash f : \beta}{\Gamma \vdash \lambda x^{\alpha}.f : (\alpha \rightarrow \beta)}\ {\scriptstyle \rightarrow I}$$

Definition 27 (→E)

$$\frac{\Gamma \vdash f : (\alpha \rightarrow \beta) \quad \Delta \vdash g : \alpha}{\Gamma, \Delta \vdash (f\ \ g) : \beta}\ {\scriptstyle \rightarrow E}$$

Definition 28 (EFSQ)

$$\frac{\Gamma \vdash t : \bot}{\Gamma \vdash (\text{abort}^{\alpha}\ \ t) : \alpha}\ {\scriptstyle EFSQ}$$

4.2.3 Structural Rules

Definition 29 (Exchange)

$$\frac{\Gamma, \alpha \vdash t : \beta}{\alpha, \Gamma \vdash t : \beta}\ {\scriptstyle \textit{Exchange}}$$

Definition 30 (Weakening)

$$\frac{\Gamma \vdash t : \beta}{\Gamma, x : \alpha \vdash t : \beta}\ {\scriptstyle \textit{Weakening}}$$

Definition 31 (Contraction)

$$\frac{\Gamma, x : \alpha, y : \alpha \vdash t : \beta}{\Gamma, u : \alpha \vdash t[x := u, y := u] : \beta}\ {\scriptstyle \textit{Contraction}}$$

4.3 Reduction Rules

Given a system of terms representing proof objects, we can provide a set of rules over terms which represent *proof reduction*:

Definition 32 (Reduction Rules)

projection operators

$$\begin{array}{rcl} (\text{fst } (x, y)) & \rightsquigarrow & x \\ (\text{snd } (x, y)) & \rightsquigarrow & y \end{array}$$

cases operator

$$\begin{array}{rcl} (\text{cases } (\text{inl } x) \; f \; g) & \rightsquigarrow & (f \; x) \\ (\text{cases } (\text{inr } x) \; f \; g) & \rightsquigarrow & (g \; x) \end{array}$$

β-reduction

$$((\lambda x.y) \; z) \quad \rightsquigarrow \quad y[x := z]$$

4.4 Curry-Howard Isomorphism

There a correspondence between the typed λ-calculus and constructive logic ('Curry-Howard Isomorphism'), under which propositions correspond to types and proofs correspond to members of those types (pairs, functions, etc.). The rules of constructive logic can be re-interpreted as rules of program construction in a typed functional programming language: under this interpretation, the linguistic formation rules define what the types of the programming language are, the introduction and elimination rules define which expressions belong to which types, and the proof reduction rules define how expressions can be evaluated. The pair $x : \varphi$, which earlier we read as 'x is a proof of proposition φ', is read under this interpretation as 'x is (an expression) of type φ'.

4.5 Expressiveness and Fixpoints

The typed term calculus is strictly less expressive than the *untyped* λ-calculus (note that every typed term has a normal form). In order to recover the ability to express every computable function, we add, for every type φ, a *general recursion* operator fix^{φ} of type $((\varphi \to \varphi) \to \varphi)$:

$$\text{fix}^{\varphi} : ((\varphi \to \varphi) \to \varphi)$$

together with the reduction rule

$$(\mathrm{fix}^{\varphi}\ \ f) \rightsquigarrow (f\ (\mathrm{fix}^{\varphi}\ \ f))$$

so that $(\mathrm{fix}^{\varphi}\ \ f)$ is a fixed point of the (functional) term f.

Note that a term containing this operator may not have a normal form, even if it is well-typed. In particular, we can derive the judgement $\vdash (\mathrm{fix}^{\varphi}\ \ \lambda x.x) : \varphi$ for every type φ, but this expression has no normal form:[1]

$$\begin{array}{rcl} (\mathrm{fix}^{\varphi}\ \ \lambda x.x) & \rightsquigarrow & (\lambda x.x\ (\mathrm{fix}^{\varphi}\ \ \lambda x.x)) \\ & \rightsquigarrow & (\mathrm{fix}^{\varphi}\ \ \lambda x.x) \\ & \rightsquigarrow & (\lambda x.x\ (\mathrm{fix}^{\varphi}\ \ \lambda x.x)) \\ & \rightsquigarrow & (\mathrm{fix}^{\varphi}\ \ \lambda x.x) \\ & \rightsquigarrow & \ldots \end{array}$$

5 Conclusion

In the next session, we introduce the view of premises as resources, and discuss the central motivations of relevance logic and linear logic.

[1]Note that this judgement is interpreted logically as an assertion that $(\mathrm{fix}^{\varphi}\ \ \lambda x.x)$ is a proof of φ, for *any* proposition φ. So, the absence of a normal form in this case should be viewed as a good thing.