Justifications, type theory and modal calculi:

a proposal

Konstantinos Pouliasis

August 4, 2016

Abstract

This work consists of my dissertation proposal and it is stuctured in two parts: The first part is, essentially, my second examination paper with some revisions and additions. It introduces basic elemements of my research topic which rests in the intersection of Justification Logic, Constructive Modality and Type Theory. I will present the relevant systems syntactically and I will pause on the basic metatheoretic proof techniques which will be useful in the rest of the text. In the second part I will delineate the current state of my research in the area. I will elaborate on a modal type system that enhances simple type theory with elements of justification logic. I will present its accompanying calculus obtained a la Curry-Howard and I will argue for its computational relevance. More specifically, I will show that the obtained calculus characterizes certain computational phenomena that abound in modern programming language semantics. I will omit full metatheoretic results here and I will merely hint to proof methods that can be adopted from the first part. Finally, I will propose certain directions for future work. Such developments together with the omitted metatheoretic results are expected to be the study of my dissertation thesis. This is a paper accompanying my dissertation proposal and a partial requirement for my Phd candidacy under the supervision of Distinguished Professor Sergei Artemov at the Department of Computer Science of the Graduate Center, CUNY.

Contents

Chapter 1

Intuitionistic Logic

1.1 Intuitionism

In this Chapter, I will be presenting foundational work in the intersection of *Intuitionistic Logic* and *Type Theory*. The presentation is scaffolding following Prof. Robert Harper's lecture videos in *Homotopy Type Theory* [18] and the accompanying notes by students of the class [21]. I will often diverge to standard textbooks in the field [9], [17], [32] to present further important results.

1.1.1 A bird's eye view

In a nutshell, *Intuitionistic mathematics* is a program in foundations of mathematics that extends *Brouwer's program* [12]. Brouwer, in an almost Kantian fashion, viewed mathematical reasoning as a human faculty and mathematics

as a language of the "creative subject' aiming to communicate mathematical concepts. The concept of *algorithm* as a step–by–step constructive process is brought in the foreground in Brouwer's program. As a result, intuitionistic theories adhere to computational interpretations. In the following I will be using the terms intutionistic and *constructive* interchangeably.

For the purposes of this paper, the main diverging point of Brouwer's program, later explicated by Heyting [22] and Kolmogorov [24] [8], lies in the treatment of proofs. In contrast to classical approaches to foundations that treat proof objects as external to theories, the constructive approach treats proofs as the fundamental forms of construction and hence, as first class citizens. As a result, the constructive view of logic draws heavily from proof theory and Gentzen's developments [16]. For the reader interested also in the philosophical implications of constructive foundations and *antirealism*, Dummet's treatment is a classic in the field [13].

It has to be emphasized that proofs in the intuitionistic approach are treated as stand–alone and are not bound to formal systems (i.e the notion of proof *precedes* that of a formal system). It is necessary, hence, to draw a distinction between the notion of *proof as construction* and the typical notion of *proof in a formal system* [20, 19].

A formal proof is a proof given in a fixed formal system, such as the axiomatic theory of sets, and arises from the application of the inductively defined rules in that system. Formal proofs can, thus, be viewed as gödelizations of textual derivation in some fixed system.

Although every formal proof (in a specific system) is also a proof (assuming soundness of the system) the converse is not true. This conforms with Gödel's Incompleteness Theorem, which precisely states that there exist true propositions (with a proof in *some* formal system), but for which there cannot be given a formal proof in the formal system that is at stake. This *openness* of the nature of proofs is necessary for a foundational treatment of proofs that respects Gödelian phenomena. This is often coined as "Axiomatic Freedom" of intuitionistic foundations.

Following the same line of thought, and adopting the doctrine of *proof relevance* for obtaining true judgments, leads to another main difference of the constructive approach and the classical one i.e the (default) absence of the *law of excluded middle*. Current developments in constructive foundations like *Homotopy Type Theory* and in general systems that rely on *Martin-Löf Type Theory* [26] do not necessarily rule out *LEM* but they might permit its usage locally, if needed, in a proof.

1.2   IPL

Intuitionistic Propositional Logic (IPL) can be viewed as "the logic of *proof relevance*" conforming with the intuitionistic view described in 1.1. To judge a fact as *true* one may provide a *proof* appropriate of the fact. *Proofs* can be synthesized to obtain proofs for more complex facts (*introduction rules*) and consumed to provide proofs relevant for other facts (*elimination rules*). The

importance of the interplay between introduction and elimination rules was developed by Gentzen. A discussion on the meaning of the logical connectives that is prevalent in *MLTT* can be found in [25] Following the presentation style by Martin-Löf we split the notions of *judgment* and *proposition*. We have two main kinds of judgments:

- *Judgments* that are logical arguments about the truth(or, equivalently, proof) of a *proposition*. They might, optionally, involve assumptions on the truth (or, equivalently, proof) of other propositions. We might call these *logical judgments*.

- Judgments on *propositionality* or typeability. *Propositions* are the *subjects* of *logical judgments*. If something is judged to be a proposition then it belongs to the universe of discourse and can be mentioned in *logical judgments*.

In addition, since a *logical judgment* might involve a set Γ of assumptions (or a *context*), it is convenient to add a third kind of judgment of the form Γ ctx Thus, in IPL, we get the judgments $\phi \in$ Prop, ϕ true and Γ ctx:

$$\phi \in \textsf{Prop} \quad \phi \text{ is a (well-formed) proposition}$$

$$\phi \ \textsf{true} \quad \text{Proposition } \phi \text{ is true}$$
$$\text{i.e., has a proof.}$$

$$\Gamma \ \textsf{ctx} \quad \Gamma \text{ is a (well-formed) context of assumptions}$$

The natural deduction system of IPL is given below:

Prop Formation

$$\frac{}{P_i \in \mathsf{Prop}}\ \text{Atom} \qquad \frac{}{\top \in \mathsf{Prop}}\ \text{Top} \qquad \frac{}{\bot \in \mathsf{Prop}}\ \text{Bottom}$$

$$\frac{\phi_1 \in \mathsf{Prop} \qquad \phi_2 \in \mathsf{Prop}}{\phi_1 \supset \phi_2 \in \mathsf{Prop}}\ \text{Arr} \qquad \frac{\phi_1 \in \mathsf{Prop} \qquad \phi_2 \in \mathsf{Prop}}{\phi_1 \wedge \phi_2 \in \mathsf{Prop}}\ \text{Conj}$$

$$\frac{\phi_1 \in \mathsf{Prop} \qquad \phi_2 \in \mathsf{Prop}}{\phi_1 \vee \phi_2 \in \mathsf{Prop}}\ \text{Disj}$$

Context Formation

$$\frac{}{\mathsf{nil}\ \mathsf{ctx}}\ \text{Nil} \qquad \frac{\Gamma\ \mathsf{ctx} \qquad \phi \in \mathsf{Prop}}{\Gamma, \phi\ \mathsf{true}\ \mathsf{ctx}}\ \Gamma\text{-Add}$$

Context Reflection

$$\frac{\Gamma\ \mathsf{ctx} \qquad \phi\ \mathsf{true} \in \Gamma}{\Gamma \vdash \phi\ \mathsf{true}}\ \Gamma\text{-Refl}$$

Top Introduction − Bottom Elimination

$$\frac{}{\Gamma \vdash \top \text{ true}} \top \text{I}$$

$$\frac{\Gamma \vdash \bot \text{ true}}{\Gamma \vdash \phi \text{ true}} \bot \text{E}$$

Implication Introduction and Elimination

$$\frac{\Gamma, \phi_1 \text{ true} \vdash \phi_2 \text{ true}}{\Gamma \vdash \phi_1 \supset \phi_2 \text{ true}} \supset \text{I}$$

$$\frac{\Gamma \vdash \phi_1 \supset \phi_2 \text{ true} \qquad \Gamma \vdash \phi_1 \text{ true}}{\Gamma \vdash \phi_2 \text{ true}} \supset \text{E}$$

Conjunction Introduction and Elimination

$$\frac{\Gamma \vdash \phi_1 \text{ true} \qquad \Gamma \vdash \phi_2 \text{ true}}{\Gamma \vdash \phi_1 \wedge \phi_2 \text{ true}} \wedge \text{I}$$

$$\frac{\Gamma \vdash \phi_1 \wedge \phi_2 \text{ true}}{\Gamma \vdash \phi_1 \text{ true}} \wedge \text{E}_{\text{L}}$$

$$\frac{\Gamma \vdash \phi_1 \wedge \phi_2 \text{ true}}{\Gamma \vdash \phi_2 \text{ true}} \wedge \text{E}_{\text{R}}$$

Disjunction Introduction and Elimination

$$\frac{\Gamma \vdash \phi_1 \text{ true}}{\Gamma \vdash \phi_1 \vee \phi_2 \text{ true}} \vee \text{I}_{\text{L}}$$

$$\frac{\Gamma \vdash \phi_2 \text{ true}}{\Gamma \vdash \phi_1 \vee \phi_2 \text{ true}} \vee \text{I}_{\text{R}}$$

$$\frac{\Gamma \vdash \phi_1 \vee \phi_2 \text{ true} \qquad \Gamma, \phi_1 \text{ true} \vdash \phi \text{ true} \qquad \Gamma, \phi_2 \text{ true} \vdash \phi \text{ true}}{\Gamma \vdash \phi \text{ true}} \vee E$$

1.2.1 Basic Properties of Intuitionistic Entailment

Reflexivity

$$\frac{}{\Gamma, \phi \text{ true} \vdash \phi \text{ true}}$$

Transitivity

$$\frac{\Gamma \vdash \psi \text{ true} \qquad \Gamma, \psi \text{ true} \vdash \phi \text{ true}}{\Gamma, \phi \text{ true} \vdash \phi \text{ true}}$$

Contraction

$$\frac{\Gamma, \phi \text{ true}, \phi \text{ true} \vdash \psi \text{ true}}{\Gamma, \phi \text{ true} \vdash \psi \text{ true}}$$

> Exchange
>
> $$\frac{\Gamma \vdash \phi \text{ true}}{\pi(\Gamma) \vdash \phi \text{ true}}$$

1.3   Order Theoretic Semantics: *Hayting Algebras*

IPL viewed order theoretically gives rise to a *Hayting Algebra(HA)*. To define *HA* we need the notion of a *lattice*. For our purposes we define it as follows[1]:

> **Definition:** A *lattice* is a *pre–order* with finite meets and joins.

In addition, we define *bounded lattice* as follows:

> **Definition:** A *bounded lattice* $(L, \leq)$ is a lattice that additionally has a greatest element 1 and a least element 0, which satisfy
>
> $0 \leq x \leq 1$ for every x in L

Finally, we can define *HA*:

> **Definition:** A *HA* is a bounded lattice $(L, \leq, 0, 1)$ s.t. for every $a, b \in L$ there exists an x (we name it $a \to b$) with the properties:
>
> 1. $a \wedge x \leq b$

---

[1]One can take a lattice being a partial order. The same results hold with slight modifications.

2. x is the greatest such element

Axiomatization of HAs

We can axiomatize the meet (i.e. greatest lower bound)($\wedge$) of ϕ, ψ for any lower bound χ.

$$\overline{\phi \wedge \psi \leq \phi} \qquad\qquad \overline{\phi \wedge \psi \leq \psi}$$

$$\frac{\chi \leq \phi \quad \chi \leq \psi}{\chi \leq \phi \wedge \psi}$$

We can axiomatize the join ($\vee$)(i.e. the least upper bound) of ϕ, ψ for any upper bound χ as follows .

$$\overline{\phi \leq \phi \vee \psi} \qquad\qquad \overline{\psi \leq \phi \vee \psi}$$

$$\frac{\phi \leq \chi \quad \psi \leq \chi}{\phi \vee \psi \leq \chi}$$

We can axiomatize the existence of a greatest element as follows:

$$\overline{\chi \leq 1}$$

which says that 1 is the greatest element.

We can axiomatize the existence of a least element as follows:

$$\overline{0 \leq \chi}$$

which says that 0 is the least element.

Finally, to axiomatize *HAs* we require the existence of exponentials for every ϕ, ψ as follows:

$$\overline{\phi \wedge (\phi \supset \psi) \leq \psi} \qquad \frac{\phi \wedge \chi \leq \psi}{\chi \leq \phi \supset \psi}$$

Soundness and Completeness

Theorem. $\Gamma \vdash_{IPL} \phi$ true iff for any *Heyting Algebra* H we have $\Gamma^+ \leq \phi^*$ where $*$ is defined as the lifting of any map of Props to elements of H and $(+)$ is defined inductively on the length of Γ as follows

$$nil^+ = \top$$

$$(\Gamma, \phi)^+ = \Gamma^+ \wedge \phi*$$

Chapter 2

Lambda Calculus With Types

2.1 From intuitionistic provability to proof trees

IPL can be viewed as a declarative axiomatization of proof constructs. Take the introduction rule for conjunction as an example:

$$\frac{\Gamma \vdash \phi_1 \text{ true} \qquad \Gamma \vdash \phi_2 \text{ true}}{\Gamma \vdash \phi_1 \wedge \phi_2 \text{ true}} \wedge \text{I}$$

The rule says, "given the existence a proof of ϕ and a proof of ψ from assumptions Γ, there exists a proof of $\phi \wedge \psi$ from assumptions Γ at hand ".

We used the description "declarative" because in this format *IPL* sequents $\Gamma \vdash$ true do not describe how such existentials are realized. It is in essence a

logic of "proof relevant truth" but it does not involve the proofs themselves as first class objects.

An alternative presentation is to explicate proof constructs by directly providing a system of "proof trees". Such, an approach was actually championed in Gentzen's natural deduction systems and is the necessary move to obtain proof calculi. Once we have proof explicit proof objects (either as trees, or as we will, see as terms) the system is enriched with equality principles involving such objects. Such rules give computational value ("proof dynamics") to the constructs and are the driver idea in the "Curry–Howard Isomorphism" and its extensions.

Here we provide such a formulation in proof trees of judgments together with the equality rules on trees, essentially following Gentzen. Proof trees of judgments have the following shape:

$$J_1, \ldots, J_i$$
$$\vdots$$

$$J$$

We focus one judgments J of the form $A\ \mathsf{true}$. Here are the the rules for constructing proof trees with labeled assumptions[1]. First, the deductions

---

[1]Essentially the constructs are directed acyclic graphs since assumptions with the same label are "bind" and substitutable together but we will be cavalier with such a details

using reflection on hypothesis are valid:

$$x_1 : A_1 \text{ true}, \ldots, x_i : A_i \text{ true}$$
$$\vdots$$
$$A_{j \in 1 \ldots i} \text{ true}$$

$$x_1 : A_1 \text{ true}, \ldots, x_i : A_i \text{ true}$$
$$\vdots$$
$$\top \text{ true}$$

$$\frac{\begin{matrix} \mathcal{D} & \mathcal{E} \\ A \text{ true} & B \text{ true} \end{matrix}}{A \wedge B \text{ true}}$$

$$\frac{\begin{matrix} \mathcal{D} \\ A \wedge B \text{ true} \end{matrix}}{A \text{ true}} \qquad \frac{\begin{matrix} \mathcal{D} \\ A \wedge B \text{ true} \end{matrix}}{B \text{ true}}$$

$$\frac{\begin{matrix} \mathcal{D} \\ A \text{ true} \end{matrix}}{A \vee B \text{ true}} \qquad \frac{\begin{matrix} \mathcal{D} \\ B \text{ true} \end{matrix}}{A \vee B \text{ true}}$$
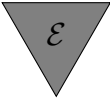
$$\cfrac{\begin{array}{ccc} & A \text{ true} & B \text{ true} \\ \mathcal{D} & \mathcal{E} & \mathcal{F} \\ A \vee B \text{ true} & C \text{ true} & C \text{ true} \end{array}}{C \text{ true}}$$

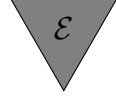$$\cfrac{\begin{array}{c} \mathcal{D} \\ \bot \text{ true} \end{array}}{C \text{ true}}$$

2.1.1 Properties of Intuitionistic Entailment Redux

Proof trees by their nature satisfy the properties of entailment in 1.2.1. We will not bother with reflection and contraction. The first is trivial and the second can be shown by simple induction on the structure of trees with the proof highlighting that reflection on hypothesis is order-irrelevant. Transitivity is established by *compositionality* of proof trees and reflects the essence of hypothetical reasoning: proof trees of the appropriate proposition can be "plugged in" for assumptions to create new valid trees.

Theorem. If
$$\begin{array}{c} x : A \\ \mathcal{D} \\ B \text{ true} \end{array}$$
and
$$\begin{array}{c} \mathcal{E} \\ A \text{ true} \end{array}$$
are valid proof trees their compo-

$$
\begin{array}{c}
\mathcal{E} \\
\overline{}
\end{array}
$$

sition denoted as $\quad A\ \mathsf{true}\quad$, defined by substituting all occurrences of

$$
\begin{array}{c}
\mathcal{D} \\
\\
B\ \mathsf{true}
\end{array}
$$

$x : A$ for E in $\mathcal{D}$, is a valid proof tree for $B\ \mathsf{true}$.

2.1.2   Equating Proof Trees

Having proof objects as first class citizens, permits for developing logics, essentially, as theories of (typed) equality among such objects. This idea was stemmed from Gentzen's work on natural deduction and cut elimination and it is what gives to proofs computational content. Here are the proposed equalities for the proof relevant *IPL* introduced initially by Gentzen as the driver of the proof cut elimination. We will be revisiting these very same equalities and reframe them as equalities among proof terms in the next section. Nevertheless, they can be expressed in proof tree form. We show indicatively the equalities regarding the $\supset$ connective proofs reserving the rest for the more concise notation.

$$
\cfrac{\cfrac{\begin{array}{c} x:A \\ \mathcal{D} \\ B \text{ true} \end{array}}{A \supset B \text{ true}} \quad \cfrac{\mathcal{E}}{A \text{ true}}}{B \text{ true}} \quad = \quad \begin{array}{c} \mathcal{E} \\ A \text{ true} \\ \mathcal{D} \\ B \text{ true} \end{array}
$$

$$
\cfrac{\cfrac{\begin{array}{c} \mathcal{D} \\ A \supset B \text{ true} \end{array} \quad \overline{x:A}}{B \text{ true}}}{A \supset B} \quad = \quad \begin{array}{c} \mathcal{D} \\ A \supset B \text{ true} \end{array}
$$

2.2 Linear representation of trees with proof terms: λ calculus

Proof terms provide an alternative linear representation for proof trees. The simply typed lambda calculus and its equational system can, thus, be viewed as a calculus for proof trees and proof reductions for intuitionistic logic. What's more, following the doctrine of proof relevance and of characterizing connectives by their proof reductions, i.e. working in the realm of Curry – Howard Isomorphism, we hit two birds with one stone: we both develop

proof relevant logics and we get typed programming languages that reflect their computational content. The "simplest" language obtained within this program is the simply typed lambda calculus, but we will see that the same doctrine extends to different logics with different judgmental constructs.

Simply typed lambda calculus

Type Formation

$$\frac{}{P_i \in \mathsf{Type}} \text{ ATOM} \qquad \frac{}{\top \in \mathsf{Type}} \text{ TOP} \qquad \frac{}{\bot \in \mathsf{Type}} \text{ BOTTOM}$$

$$\frac{\phi_1 \in \mathsf{Type} \qquad \phi_2 \in \mathsf{Type}}{\phi_1 \to \phi_2 \in \mathsf{Type}} \text{ ARR} \qquad \frac{\phi_1 \in \mathsf{Type} \qquad \phi_2 \in \mathsf{Type}}{\phi_1 \times \phi_2 \in \mathsf{Type}} \text{ PROD}$$

$$\frac{\phi_1 \in \mathsf{Type} \qquad \phi_2 \in \mathsf{Type}}{\phi_1 + \phi_2 \in \mathsf{Type}} \text{ UNION}$$

Context Formation

$$\frac{}{\mathsf{nil}\ \mathsf{ctx}} \text{ NIL} \qquad \frac{\Gamma\ \mathsf{ctx} \qquad \phi \in \mathsf{Type} \qquad x \text{ fresh in } \Gamma}{\Gamma,\ x : \phi\ \mathsf{ctx}} \text{ } \Gamma\text{-ADD}$$

Context Reflection

$$\frac{\Gamma \text{ ctx} \qquad x : \phi \in \Gamma}{\Gamma \vdash x : \phi} \ \Gamma\text{-Refl}$$

Top Introduction – Bottom Elimination

$$\frac{}{\Gamma \vdash \langle \rangle : \top} \ \top\text{I} \qquad\qquad \frac{\Gamma \vdash M : \bot}{\Gamma \vdash abort[\phi](M) : \phi} \ \bot\text{E}$$

Function Construction and Application

$$\frac{\Gamma, x : \phi_1 \vdash M : \phi_2}{\Gamma \vdash \lambda x.M : \phi_1 \to \phi_2} \ \lambda-\text{Abs} \qquad\qquad \frac{\Gamma \vdash M : \phi_1 \to \phi_2 \qquad \Gamma \vdash M' : \phi_1}{\Gamma \vdash (MM') : \phi_2} \ \text{App}$$

Tuple Construction and Projections

$$\frac{\Gamma \vdash M : \phi_1 \qquad \Gamma \vdash M' : \phi_2}{\Gamma \vdash \langle M, M' \rangle : \phi_1 \times \phi_2} \ \text{Tup}$$

$$\frac{\Gamma \vdash M : \phi_1 \times \phi_2}{\Gamma \vdash \text{fst}(M) : \phi_1} \ \text{LPrj} \qquad\qquad \frac{\Gamma \vdash M : \phi_1 \times \phi_2}{\Gamma \vdash \text{snd}(M) : \phi_2} \ \text{RPrj}$$

Union Construction and Elimination

$$\frac{\Gamma \vdash M : \phi_1}{\Gamma \vdash inj_l[\phi_2](M) : \phi_1 + \phi_2} \text{ InjL} \qquad \frac{\Gamma \vdash M : \phi_2}{\Gamma \vdash inj_r[\phi_1](M) : \phi_1 + \phi_2} \text{ InjR}$$

$$\frac{\Gamma \vdash M : \phi_1 + \phi_2 \qquad \Gamma, x : \phi_1 \vdash N : \phi \qquad \Gamma, y : \phi_2 \vdash O : \phi}{\Gamma \vdash \text{case } M \text{ of } inj_l(x) \longmapsto N \mid inj_r(y) \longmapsto O : \phi} \vee\text{E}$$

2.3 Operational (a.k.a "term") Semantics

Given the formulation of the λ-calculus we can think of *formulae-as-types* and *proof terms-as-programs*. This enriches logic with a computational aspect(*proof dynamics*) that is absent from other formulations. Dynamics stems from Gentzen's insight to give an equational system for proofs equipped with $\beta\eta$ rules for proof-tree conversion. These insights, give rise to λ-calculus dynamics if we devise an execution strategy (an *operational semantics*) for program reduction that respects these rules. The correspondence between computations and Gentzen equational principles for proof terms is enlightened by specific metatheoretic results.

2.3.1 Definitional Equality

We want to think about when two proofs $M : A$ and $M' : A$ are the same. In the following we elaborate Gentzen's principles introducing an equivalence relation called *definitional equality* that respects these principles, denoted $M \equiv M' : A$. We want definitional equality $\equiv$ to be the least congruence closed under the β rules. We will define what this means:

A *congruence* is an equivalence relation (i.e. reflexive transitive and antisymmetric) that respects our operators as formulated below.

$$\frac{\quad\quad\quad\quad\quad}{\Gamma \vdash M \equiv M : \phi} \; R \qquad\qquad \frac{\Gamma \vdash M \equiv M' : \phi}{\Gamma \vdash M' \equiv M : \phi} \; S$$

$$\frac{\Gamma \vdash M \equiv M' : \phi \qquad \Gamma \vdash M' \equiv M'' : \phi}{\Gamma \vdash M \equiv M'' : \phi} \; T$$

For the equivalence relation to respect operators we means that if $M \equiv M' : A$, then that their image under any operator should be equivalent. In other words, we should be able to replace M with M' everywhere. For example:

Congruence over fst

$$\frac{\Gamma \vdash M \equiv M' : A \wedge B}{\Gamma \vdash \mathrm{fst}(M) \equiv \mathrm{fst}(M') : A}$$

Inversion Principle

Gentzen's Inversion Principle captures the idea that "elim is post-inverse to intro," which is the informal notion that the elimination rules should cancel the introduction rules.

The β rules are as follows:

$$\frac{\Gamma \vdash M : \phi_1 \qquad \Gamma \vdash N : \phi_2}{\Gamma \vdash \mathrm{fst}(\langle M, N \rangle) \equiv M : \phi_1} \ \beta\wedge_1$$

$$\frac{\Gamma \vdash M : \phi_1 \qquad \Gamma \vdash N : \phi_2}{\Gamma \vdash \mathrm{snd}(\langle M, N \rangle) \equiv N : \phi_2} \ \beta\wedge_2 \qquad \frac{\Gamma, x : A \vdash M : B \qquad \Gamma \vdash N : A}{\Gamma \vdash (\lambda x.M)(N) \equiv [N/x]M : B} \ \beta \supset$$

$$\frac{\Gamma, x : \phi_1 \vdash N : \psi \qquad \Gamma, y : \phi_2 \vdash O : \psi \qquad \Gamma \vdash P : \phi_1}{\Gamma \vdash (\mathrm{case}\ inj_l(P)\ \mathrm{of}\ inj_l(x) \longmapsto N|\ inj_r(y) \longmapsto O) \equiv [P/x]N : \psi} \ \beta\vee_1$$

$$\frac{\Gamma, x : \phi_1 \vdash N : \psi \qquad \Gamma, y : \phi_2 \vdash O : \psi \qquad \Gamma \vdash Q : \phi_2}{\Gamma \vdash (\mathrm{case}\ inr_r(Q)\ \mathrm{of}\ inj_l(x) \longmapsto N|\ inj_r(y) \longmapsto O) \equiv [Q/y]O : \psi} \ \beta\vee_2$$

Unicity Principle

Gentzen's Unicity Principles on the other hand capture the idea that "intro is post-inverse to elim". There should be only one way – modulo definitional equivalence – to prove something. The "β" rules give rise to computational interpretation. The "η" rules impose properties that the computational model

should satisfy (e.g. Church-Rosser property).

The η rules are given below:

$$\frac{\Gamma \vdash M : \top}{\Gamma \vdash M \equiv \langle\,\rangle : \top} \, \eta\top \qquad\qquad \frac{\Gamma \vdash M \equiv \langle \mathrm{fst}(M), \mathrm{snd}(M)\rangle : A \wedge B}{\Gamma \vdash M : A \wedge B} \, \eta\wedge$$

$$\frac{\Gamma \vdash M : \phi \supset \psi}{\Gamma \vdash M \equiv \lambda x.Mx : \phi \supset \psi} \, \eta\supset$$

$$\frac{\Gamma, z : \phi_1 + \phi_2 \vdash M : \psi \qquad \Gamma \vdash N : \phi_1 + \phi_2}{\Gamma \vdash M[N/z] \equiv \mathrm{case}\ N\ \mathrm{of}\quad |inj_l(x) \mapsto M[inj_l(x)/z] \atop inj_r(y) \mapsto M[inj_r(y)/z]) : \psi} \, \eta\vee$$

2.3.2   Propositions as Types

There is a correspondence between propositions and types:

Propositions	Types
$\top$	1
$A \wedge B$	$A \times B$
$A \supset B$	function $A \rightarrow B$ or B^A
$\bot$	0
$A \vee B$	$A + B$

2.4   Categories for proof relevant *IPL*

In a Heyting Algebra, we have a preorder (or, partial order in the "textbook" definition) $\phi \leq \psi$ when ϕ implies ψ. *HAs* are insufficient, however, for the treatment of proof objects (there can be at most one instance of $\phi \leq \psi$ for specific ϕ,ψ). We can keep track of proofs, so if M is a proof from Γ to ψ, we want to think of it as a map $M : \Gamma+ \to \psi+$. The category theoretic analog of a Heyting Algebra is a Bi–Cartesian Closed Category (*BiCCC*). That is a category with all finite products, co–products and exponentials. The axiomatization of a category (in general), finite (and nullary) products and co-products and exponentials is given in this section.

2.4.1   Definitions and Axioms of a Category

A category has *objects* $\phi, \psi, \ldots$ and *arrows* $f, g, h \ldots$ Each arrow goes from an object to an object. To say that g goes from ϕ to ψ we write $g : \phi \to \psi$, or say that ϕ is the domain of g, and ψ the *co–domain*. We write $Dom(g) = \phi$ and $Cod(g) = \psi$. We say that two arrows f and g are *composable* with $Dom(f) = Cod(g)$. If f and g are composable, they have a *composite*, an arrow called $f \circ g$. There is an *identity* for every object ϕ.

$$\frac{}{\text{id} : \phi \to \phi} \; \text{ID}_{ex} \qquad \frac{f : \phi \to \psi \qquad g : \psi \to \chi}{g \circ f : \phi \to \chi} \; \textsc{Comp}$$

$$\frac{f : \phi \to \psi}{id_\psi \circ f = f : \phi \to \psi} \; \text{ID}_l \qquad \frac{f : \phi \to \psi}{f \circ id_\phi = f : \phi \to \psi} \; \text{IDR}$$

$$\frac{f : \phi \to \psi \qquad g : \psi \to \chi \qquad h : \chi \to \omega}{h \circ (g \circ f) = (h \circ g) \circ f : \phi \to \omega} \; \text{IDR}$$

2.4.2 Terminal, Co-Terminal objects, Products and Co-Products

Now we can think about objects in the category that correspond to propositions given in the correspondence.

Terminal Object 1 is the terminal object, also called the final object, which corresponds to $\top$. For any object Γ there is a unique map $\Gamma \to 1$.

$$\frac{}{\langle\,\rangle : \phi \to 1} \; \textsc{Existence} \qquad \frac{M : \Gamma \to 1}{M = \langle\,\rangle : \Gamma \to 1} \; \textsc{Unicity}(\eta)$$

Product   For any objects ϕ and ψ there is an object $\chi = \phi \times \psi$ equipped with arrows fst : $\phi \times \psi \to \phi$ and snd : $\phi \times \psi \to \psi$ that is the *product* of ϕ and ψ, which corresponds to the join $\phi \wedge \psi$. For any other object Γ with arrows $M : \Gamma \to \phi$ and $\Gamma \to \psi$ there exists *unique* arrow, $\langle M, N \rangle$ s.t. fst $\circ \langle M, N \rangle = M(\beta \times_1)$ and snd $\circ \langle M, N \rangle = N(\beta \times_2)$.

$$\frac{M : \Gamma \to \phi \qquad N : \Gamma \to \psi}{\langle M, N \rangle : \Gamma \to \phi \times \psi} \; \text{Exist}_1$$

$$\frac{M : \Gamma \to \phi \qquad N : \Gamma \to \psi}{\text{fst} \circ \langle M, N \rangle : \Gamma \to \phi} \; \text{Exist}_2(\beta_1)$$

$$\frac{M : \Gamma \to \phi \qquad N : \Gamma \to \psi}{\text{snd} \circ \langle M, N \rangle : \Gamma \to \phi} \; \text{Exist}_3(\beta_2)$$

$$\frac{P : \Gamma \to \phi \times \psi \qquad \text{fst} \circ P = M : \Gamma \to \phi \qquad \text{snd} \circ P = N : \Gamma \to \psi}{P = \langle M, N \rangle : \Gamma \to \phi \times \psi} \; \text{Un}(\eta)$$

Diagrammatically:
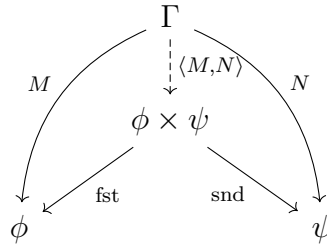
Exponentials Given objects A and B, an exponential B^A (which corresponds to $A \supset B$) is an object with the following universal property:

$$
\begin{array}{ccc}
C & & C \times A \\
\big\downarrow \lambda(h) & & \big\downarrow \lambda(h) \times \mathrm{id}_A \quad\searrow^{h} \\
B^A & & B^A \times A \xrightarrow{\ \mathrm{ap}\ } B
\end{array}
$$

such that the diagram commutes.

This means that there exists a map $\mathrm{ap} : B^A \times A \to B$ (application map) that corresponds to implication elimination.

The universal property is that for all objects C that have a map $h : C \times A \to B$, there exists a unique map $\lambda(h) : C \to B^A$ such that

$$
\mathrm{ap} \circ (\lambda(h) \times \mathrm{id}_A) = h : C \times A \to B
$$

This means that the diagram commutes. Another way to express the induced map is $\lambda(h) \times \mathrm{id}_A = \langle \lambda(h) \circ \mathrm{fst}, \mathrm{snd} \rangle$.

The map $\lambda(h) : C \to B^A$ is unique, meaning that

$$
\frac{\mathrm{ap} \circ (g \times \mathrm{id_A}) = h : C \times A \to B}{g = \lambda(h) : C \to B^A}
$$

Co–Products For any objects ϕ and ψ there is an object $\chi = \phi + \psi$ equipped with arrows $\mathrm{inl} : \phi \to \phi + \psi$ and $\mathrm{inr} : \psi \to \phi + \psi$ that is the

co-product of ϕ and ψ, which corresponds to the meet $\phi \wedge \psi$. For any other object ω with arrows $M : \omega \to \phi \vee \psi$ and $N : \omega \to \phi \vee \psi$ there exists *unique* arrow, M, N s.t. $\{M, N\} \circ \text{inl} = M$ and $\{M, N\} \circ \text{inr} = N$.

$$\frac{O : \Gamma \to \phi}{\text{inl} \circ O : \Gamma \to \phi + \psi} \text{EXIST}_1 \qquad \frac{P : \Gamma \to \psi}{\text{inr} \circ P : \Gamma \to \phi + \psi} \text{EXIST}_2$$

$$\frac{O : \Gamma \to \phi \qquad M : \phi \to \omega \qquad N : \psi \to \omega}{\{M, N\} \circ \text{inl} \circ O = M \circ O : \Gamma \to \omega} \text{EXIST}_3(\beta_1)$$

$$\frac{P : \Gamma \to \psi \qquad M : \phi \to \omega \qquad N : \psi \to \omega}{\{M, N\} \circ \text{inr} \circ P = N \circ P : \Gamma \to \omega} \text{EXIST}_3(\beta_2)$$

$$\frac{M : \Gamma \to \phi \qquad N : \Gamma \to \psi}{\text{snd} \circ \langle M, N \rangle : \Gamma \to \phi} \text{EXIST}_3(\beta_2)$$

$$\frac{O : \Gamma \to \phi \quad P : \Gamma \to \psi \quad U : \phi + \psi \to \omega \quad M : \phi \to \omega \quad N : \psi \to \omega \quad U \circ \text{inl} \circ O = M \quad U \circ \text{inr} \circ N = M}{U = \{M, N\}} \text{UN}(\eta)$$

Diagrammatically:

$$
\begin{array}{ccc}
\phi & & \psi \\
 & \phi + \psi & \\
 & C &
\end{array}
$$

inl inr

(β) (β)

M M $((\eta))$ $\{M,N\}$ N
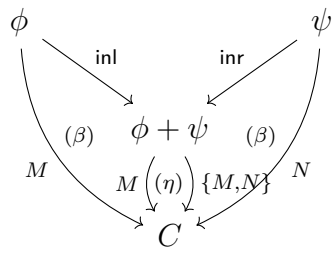
Chapter 3

Justification Logic

In the second part of this paper I will give an overview of *Justification Logic* (JL) highlighting the parts that are closely related to constructivity to remain coherent with 1. I will emphasize LP, the very first logic of justification, and its deep relation with IPL. My scaffolding will be based upon [6], [4] that reflect this relation. Beforehand, I will allow for a more general discussion on JL following [2] and other relevant papers.

3.1   A bird's eye view

According to [2]

> Justification logics are epistemic logics which allow knowledge
> and belief modalities to be "unfolded" into justification terms.

More specifically, in JL the modality in question is witnessed by a reason

and propositions of the kind become $t : \phi$ that reads "ϕ is justified by reason t". Witnesses in JL have structure and operations. Different choices of operators result in logics that explicate different modalities (K,T,$S4$,$S5$). For our purposes, and in addition to type theoretic approaches to logic, JL reveals a computational content for *validity* in classical terms. As we will see following [1], JL and especially its $S4$ counterpart *The Logic of Proofs* (LP), can provide a unified classical *semantics* for type theoretic formulations of intuitionistic logic. In addition, following [7] and [33], JL mechanics can be viewed type theoretically to provide for modal typed systems that enrich computational type theories with "semantical" notions such as explicit reflection and modular binding.

3.2   Minimal Justification Logic J_0

To permit for an account of reasons, the logic is enriched with an extra sort for j for justifications. The sort of propositions is then enriched with propositions of the kind $j : \phi$ with ϕ being a proposition. Here is the abstract syntax:

$$j :=s_i|\ C_i|j_1 * j_2|j_2 + j_2$$

$$\phi :=P_i|\ \bot|\ \phi_1 \wedge \phi_2|\ \phi_1 \vee \phi_2|\ \phi_2 \supset \phi_2|\ \neg\phi|\ j : \phi$$

Constants C_i are symbols that can be assigned to logic axioms that are

assumed to be necessary. Weaker justifications logics exist without any assignment of constants (empty *constant specifications*) or with partial constant specifications. Nevertheless, in order for the *rule of necessitation* to be admissible each axiom instance of the underlying propositional logic has to be assigned a constant. We will be coming back to this topic in later sections. Symbols s_i stand for variables.

A Hilbert–style axiomatization of J_0 is given below. Its components are Hilbert's axioms for propositional logic together with two basic rules for justification: *applicativity* and *concatenation*. Concatenation internalizes weakening of proofs.

Propositional Axioms

P1. $\vdash \phi \supset (\psi \supset \phi)$

P2. $\vdash (\phi \supset (\psi \supset \chi)) \supset ((\phi \supset \psi) \supset (\phi \supset \chi))$

P3. $\vdash \phi \supset \psi \supset \phi \wedge \psi$

P4. $\vdash \phi \supset \psi \supset \psi \wedge \phi$

P5. $\vdash \phi \supset \phi \vee \psi$

P6. $\vdash \psi \supset \phi \vee \psi$

P7. $\vdash (\phi \supset \psi) \supset (\neg\psi \supset \neg\phi)$

Justification Axioms

$$\text{Times.} \vdash j : (\phi \supset \psi) \supset (j' : \phi \supset j * j' : \psi)$$

$$\text{PlusL.} \vdash j : \phi \supset (j + j' : \phi)$$

$$\text{PlusR.} \vdash j : \phi \supset (j' + j : \phi)$$

The rule of the system is *Modus Ponens.*

Modus Ponens

$$\frac{\phi \supset \psi \qquad \phi}{\psi} \ \text{MP}$$

For the rule of necessitation to be admissible, we need necessitation of axioms to be admissible. For that reason a constant specification is required. We focus here on axiomatically appropriate constant specification CS because of its relation to combinatorial calculi. An axiomatization of axiomatically appropriate CS given below. Elements of CS are pairs (C, ϕ) of constants and propositions:

Axiomatic CS

$$\frac{}{\vdash (\mathsf{C_1}[\phi,\psi],\ \phi \to (\psi \to \phi)) \in \mathsf{CS}}\ \mathsf{C_1}$$

$$\frac{}{\vdash (\mathsf{C_2}[\phi,\psi,\chi],\ (\phi \supset (\psi \supset \chi)) \supset ((\phi \supset \psi) \supset (\phi \supset \chi)))) \in \mathsf{CS}}\ \mathsf{C_2}$$

$$\frac{}{\vdash (\mathsf{C_3}[\phi,\psi],\ \phi \supset \psi \supset \phi \wedge \psi) \in \mathsf{CS}}\ \mathsf{C_3}$$

$$\frac{}{\vdash (\mathsf{C_4}[\phi,\psi],\ \phi \supset \psi \supset \psi \wedge \phi) \in \mathsf{CS}}\ \mathsf{C_4}$$

$$\frac{}{\vdash (\mathsf{C_5}[\phi,\psi],\ \phi \supset \phi \vee \psi) \in \mathsf{CS}}\ \mathsf{C_5} \qquad \frac{}{\vdash (\mathsf{C_6}[\phi,\psi],\ \psi \supset \phi \vee \psi) \in \mathsf{CS}}\ \mathsf{C_6}$$

$$\frac{}{\vdash (\mathsf{C_7}[\phi,\psi],\ (\phi \supset \psi) \supset (\neg\psi \supset \neg\phi)) \in \mathsf{CS}}\ \mathsf{C_7}$$

$$\frac{}{\vdash (\mathsf{C_8}[\phi,\psi,j,j'],\ j:(\phi \supset \psi) \supset (j':\phi \supset j*j':\psi)) \in \mathsf{CS}}\ \mathsf{C_8}$$

$$\frac{\vdash (\mathsf{C},\phi) \in \mathsf{CS}}{\vdash (\mathsf{C!},\ \mathsf{C}:\phi) \in \mathsf{CS}}\ C!$$

Finally we require reflection on CS:

Specification Reflection

$$\frac{\vdash (\mathsf{C},\phi) \in \mathsf{CS}}{\vdash \mathsf{C}:\phi}\ \mathrm{CSR}$$

The system can be given a Natural Deduction formulation a la IPL since the following theorem holds:

> **Deduction Theorem** For any set of propositional assumptions Γ,
> $\Gamma, \phi \vdash \psi$ implies $\Gamma \vdash \phi \supset \psi$

3.3 Epistemic motivation

JL as an epistemic logic departs from previous traditions of logic of knowledge based on universality judgments. From [2]

> The modal approach to the logic of knowledge is, in a sense, built around the universal quantifier: X is known in a situation if X is true in all situations indistinguishable from that one. Justifications, on the other hand, bring an existential quantifier into the picture: X is known in a situation if there exists a justification for X in that situation

This fresh approach on epistemic tradition has been utilized to solve many problems in formal epistemology (see [3]). We give here a solution to the famous 'Red barn problem' that is also a pedagogical example on how deduction in the system works.

The red barn problem can be stated as follows:

> Suppose I am driving through a neighborhood in which, unbeknownst to me, papier-mâché barns are scattered, and I see that

the object in front of me is a barn. Because I have barn-before-me percepts, I believe that the object in front of me is a barn. Our intuitions suggest that I fail to know barn. But now suppose that the neighborhood has no fake red barns, and I also notice that the object in front of me is red, so I know a red barn is there. This juxtaposition, being a red barn, which I know, entails there being a barn, which I do not, "is an embarrassment"

The red barn example can be represented in a system of modal logic where $\Box \phi$ represents knowledge of ϕ that, in contrast to the the justified approach, is forgetful with respect to reasons. The formalization and the accompanying problem go as follows:

1. $\Box B$, 'I believe that the object in front of me is a red barn'.

2. $\Box(B \wedge R)$, 'I believe that the object in front of me is a red barn'.

   At the metalevel, 2 is actually knowledge, whereas by the problem description, 1 is not knowledge.

3. $\Box(B \wedge R \supset B)$, a knowledge assertion of a logical axiom.

   Within this formalization, it appears that epistemic closure in its modal form (2) is violated:line 2, $\Box(B \wedge R)$, and line 3, $(B \wedge R \supset B)$ are cases of knowledge whereas $\Box B$ (line 1) is not knowledge. The modal language here does not seem to help resolving this issue.

Of course, one can resolve this by introducing a second modality(e.g. for 'I believe that'). But then similar problems can occur (e.g. by adding a third

modality read as 'it should be'). Indexing of modalities with reasons solves this problem in its generality: by permitting the applicative closure only on reasons of the same sort one can overcome this defect.

1. $u : B$, 'u is a reason to believe that the object in front of me is a barn';

2. $v : (B \wedge R)$, 'v is a reason to believe that the object in front of me is a red barn';

3. $a : (B \wedge R \supset B)$, because of logical awareness.

   On the metalevel, the problem description states that 2 and 3 are cases of knowledge, and not merely belief, whereas 1 is belief which is not knowledge. Here is how the formal reasoning goes:

4. $a : (B \wedge R \supset B) \supset (v : (B \wedge R) \supset a * v : B)$, by Times

5. $v : (B \wedge R) \supset a * v : B$, from 3 and 4, by propositional logic; $a * v : B$, from 2 and 5, by propositional logic.

3.4   Proof theoretic view

In 1 we gave an analytic account of the *Brower-Heyting-Kolmogorov* (BHK) principles of constructive proofs. In the paper "Eine Interpretation des intuitionistischen Aussagenkalküls ", Göedel gave a classical provability interpretation of BHK using the modal system S4.

The standard axiomatization of S4 is given below:

The system S4

$$P1 - P7$$

K. $\vdash \Box(\phi \supset \psi) \supset (\Box\phi \supset \Box\psi)$

T. $\vdash \Box\phi \supset \phi$

4. $\vdash \Box\phi \supset \Box\Box\phi$

Modus Ponens

$$\frac{\phi \supset \psi \qquad \phi}{\psi} \text{ MP}$$

Göedel's result can be summarized in the following theorem:

Gödel-Tarski Translation of Intuitionistic Logic

$$\Gamma \vdash_{\mathsf{IPL}} \phi \rightarrow \Gamma \vdash_{\mathsf{S4}} \mathrm{tr}(\phi)$$

where $\mathrm{tr}(\phi)$ is obtained by ϕ by $\Box$-ing its subformulas.

After this result the state of the project of a classical interpretation of BHK semantics was as follows: IPC $\hookrightarrow$ S4 $\hookrightarrow$? $\hookrightarrow$ CLASSICAL PROOFS. Filling the missing part was the motivation behind LP, the first Justification Logic.

3.5   The Logic of Proofs

An axiomatization of LP with axiomatically appropriate constant specification
as defined in 3.2 can be given as follows:

The system LP

$$P1 - P7$$

$$\text{Times.} \vdash j : (\phi \supset \psi) \supset (j' : \phi \supset j * j' : \psi)$$

$$\text{PlusL.} \vdash j : \phi \supset (j + j' : \phi)$$

$$\text{PlusR.} \vdash j : \phi \supset (j' + j : \phi)$$

$$\text{T.} \vdash j : \phi \supset \phi$$

$$4. \vdash j : \phi \supset (j! : j : \phi)$$

3.6   Metatheoretic Results

The *Deduction Theorem* holds for LP

Deduction Theorem Any deduction of the kind $\Gamma, \phi \vdash \psi$ implies $\Gamma \vdash$
$\phi \supset \psi$.

Also, the lifting property can be obtained:

Lifting Lemma

Any deduction of the kind $\vec{j} : \Gamma, \Delta \vdash \phi$ implies $\vec{j} : \Gamma, \vec{s} : \Delta \vdash$
$j'(\vec{j}, \vec{s}) : \phi$ where $\vec{j}$ is a vector metavariables to be substituted for arbitrary

polynomials and $\vec{s}$ is a vector of (object) variables.

In addition, LP is the forgetful projection of $S4$. More specifically, consider and formula of LP ϕ and the transformation $F_\Box(\phi)$ that replaces all subformulae of ϕ of the kind $j : \phi'$ with $\Box \phi'$. The following theorem holds:

Forgetful Projection Property

$\Gamma \vdash_{\mathsf{LP}} \phi$ implies $\Gamma \vdash_{\mathsf{S4}} F_\Box(\phi)$

The inverse also holds as the realization theorem says. Before introducing the realization procedure we give a motivating example.

Example: Realization of $\vdash_{\mathsf{S4}} \Box\phi \vee \Box\psi \supset \Box(\phi \vee \psi)$

1. $\phi \supset \phi \vee \psi$, $\psi \supset \phi \vee \psi$ Prop. Axioms;

2. $C : (\phi \supset \phi \vee \psi)$, $C' : (\psi \supset \phi \vee \psi)$ From CS rules.

3. $s : \phi \supset C * s : \phi \vee \psi$, From 1,2 and Times and MP

4. $t : \psi \supset C' * t : \phi \vee \psi$, Similarly

5. $C*s : \phi\vee\psi \supset (C*s+C'*t) : \phi\vee\psi$ and $C'*t : \phi\vee\psi \supset (C*s+C'*t) : \phi\vee\psi$, From Rplus, Lplus

6. $s : \phi \supset (C * s + C' * t) : \phi \vee \psi$, From 3,5 by Propositional Logic.

7. $t : \psi \supset (C * s + C' * t) : \phi \vee \psi$, From 4,5 by Propositional Logic.

8. $s : \phi \vee t : \psi \supset (C * s + C' * t) : \phi \vee \psi$, From 6,7 and Propositional Logic.

3.6.1 Realization

The realization gives an algorithmic process of transforming deductions in S4 to LP. By an LP-realization of a modal formula ϕ we mean an assignment of proof polynomials to all occurrences of the modality in ϕ. Let ϕ^r be the image of ϕ under a realization r.

The polarity of $\Box$s in a formula is relevant in realizations. We define positive and negative occurrences of modality in a formula and a sequent.

$\Box$ Polarities

1. The indicated occurrence of $\Box$ in $\Box\phi$ is of positive polarity;

2. any occurrence of $\Box$ in the subformula ϕ of $\psi \supset \phi, \psi \wedge \phi, \phi \wedge \psi, \psi \vee \phi, \phi \vee \psi, \Box\phi, \Gamma \Rightarrow \Delta, \phi$ – we will be defining $\Rightarrow$ momentarily – has the same polarity as the same occurrence of $\Box$ in ϕ.

3. any occurrence of $\Box$ in the subformula ϕ of $\neg\phi, \phi \supset \psi, \Gamma, \phi \Rightarrow \Delta$, has polarity opposite to the polarity of the very same occurrence of $\Box$ in ϕ.

Next we give a a cut-free sequent formulation of S4 (reference) with sequents $\Gamma \vdash \Delta$, where Γ and Δ are finite multisets of modal formulas. The left hand multisets are to be read conjunctively and the right hand ones disjunctively. The rules are the rules given below together with the typical structural ones.

$$\frac{}{\Gamma, \phi \vdash \phi, \Delta} \text{ Refl} \qquad \frac{\Gamma \vdash \phi, \Delta}{\Gamma, \neg\phi \vdash \Delta} \neg\text{L} \qquad \frac{\phi, \Gamma \vdash \Delta}{\Gamma \vdash \neg\phi, \Delta} \neg\text{R}$$

$$\frac{\Gamma, \phi, \psi \vdash \Delta}{\Gamma, \phi \wedge \psi \vdash \Delta} \wedge\text{L} \qquad \frac{\Gamma \vdash \phi, \Delta \qquad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \phi \wedge \psi, \Delta} \wedge\text{L}$$

$$\frac{\Gamma, \phi \vdash \Delta \qquad \Gamma, \psi \vdash \Delta}{\Gamma, \phi \vee \psi \vdash \Delta} \vee\text{L} \qquad \frac{\Gamma \vdash \phi, \psi, \Delta}{\Gamma \vdash \phi \vee \psi \vdash \Delta} \vee\text{R}$$

$$\frac{\Gamma \vdash \phi, \Delta \qquad \Gamma\psi \vdash \Delta}{\Gamma, \phi \supset \psi \vdash \Delta} \supset\text{L} \qquad \frac{\Gamma \vdash \phi, \Delta \qquad \Gamma\psi \vdash \Delta}{\Gamma, \phi \supset \psi \vdash \Delta} \supset\text{R}$$

$$\frac{\phi, \Gamma \vdash \Delta}{\Box\phi, \Gamma \vdash \Delta} \Box\text{L} \qquad \frac{\Box\Gamma \vdash \phi, \Delta}{\Box\Gamma \vdash \Box\phi, \Delta} \Box\text{R}$$

Relevant in the realization proof is the sequent formulation of LP, the system LPG which enjoys the cut-elimination property resulting in the system LPG$^-$. The rules relevant to justifications are given below.

$$\frac{\Gamma, \phi \vdash \phi, \Delta}{\Gamma, t : \phi \vdash \phi, \Delta} \text{ :L} \qquad \frac{\Gamma \vdash t : phi, \Delta}{\Gamma \vdash !t : t : \phi, \Delta} \text{ !R} \qquad \frac{\Gamma \vdash t : \phi, \Delta}{\Gamma \vdash (t + s) : \phi, \Delta} \text{ +L}$$

$$\frac{\Gamma \vdash t : \phi, \Delta}{\Gamma \vdash (s + t) : \phi, \Delta} \text{ +R} \qquad \frac{\Gamma \vdash s : \phi \supset \psi, \Delta \qquad \Gamma \vdash t : \phi, \Delta}{\Gamma \vdash s * t : \psi, \Delta} \text{ *R}$$

$$\frac{\Gamma \vdash \phi, \Delta}{\Gamma \vdash c : \phi, \Delta} \text{ } c\text{R}$$

Utilizing the previous systems the realization theorem shows:

Realization Theorem If $\Gamma \vdash_{\mathsf{S4}} \phi$ then there is a *normal* realization s.t. $\Gamma \vdash_{\mathsf{LP}} \phi^r$. By normal we mean a realization for which all occurrences of $\square$ are realized by proof variables and the corresponding constant specification is injective.

3.6.2 Kripke - Fitting Semantics

In this section I will be discussing Kripke – Fitting Semantics[15] for Justification Logic $\mathsf{J_0} + \mathsf{CS}$ very briefly.

A possible world justification logic model for the system $\mathsf{J_0} + \mathsf{CS}$ is a structure $M = \langle G, R, E, V \rangle$. $\langle G, R \rangle$ is a standard K frame, where G is a set of possible worlds and R is a binary relation on it. V is a mapping from propositional variables to subsets of G, specifying atomic truth at possible

worlds. E is an evidence function that maps pairs of justification terms and formulas to sets of worlds.

Given such a model, we define the $\models$ relation as follows:

$\forall \Gamma \in G$

$\quad M, \Gamma \models P$ iff $\Gamma \in V(P)$ for P a propositional letter

- It is not the case that $M, \Gamma \models \bot$
- $M, \Gamma \models \phi \supset \psi$ iff it is not the case that $M, \Gamma \models \phi$ or $M, \Gamma \models Y$
- $M, \Gamma \models (j : \phi)$ if and only if $\Gamma \in E(j, \phi)$ and, $\forall \Delta \in G$ with $\Gamma R \Delta$, we have that $M, \Delta \models \phi$.

The following conditions on evidence functions are assumed:

$E(j, \phi \supset \psi) \cap E(j', \phi) \subseteq E(j * j', \psi)$

$E(j, \phi) \cup E(j', \phi) \subseteq E(j + j', \phi)$

Finally, the Constant Specification CS should be taken into account. Recall that constants are intended to represent reasons for basic assumptions that are accepted outright. A model $M = \langle G, R, E, V \rangle$ meets Constant Specification CS provided: if $(C, \phi) \in CS$ then $E(c, \phi) = G$.

Typical, soundness and completeness results can be shown for such models. They can also be extended for all other justification logics.

Chapter 4

Modal Type Theory

In this chapter I will give a short, example-driven introduction to typed modality and its applications. The discussion will remain informal when it comes to metatheory and will be focused on usage of modal typed calculi in applications. Apropos, I will discuss Operational Semantic as the essence of the computational approach to logic.

4.1   A bird's eye view

Significant in the development of modal logic within a type-theoretic framework is the work of Moggi [27]. In his seminal work he mentioned the need of shifting from simply typed calculi to systems that can capture notions of computation such exceptions, partial functions, binding constructs etc. Moggi's initiative stems from a categorical point of view.

In the realm of deductive systems with explicit witnesses, Artemov's work on Operational Modal Logic[5], and the "Gang of four" [11],[**?**] revived the interest in constructive modality and its computational view. Of course earlier work in the intersection of modal and constructive logic exists (cf. [34],[14]) but its relation with programming languages is not yet explicit. It's worth mentioning that the work by DePaiva initiates from a categorical view too. That is Categorical Semantics for Linear Logic.

Since it is of great importance in my work, I will be presenting here the *judgmental* approach followed by Pfenning's "Judgmental Reconstruction of Modal Logic" [31] which is a foundational approach that captures previous work on $\Box$ Calculi for $S4$. Although the system presented is a judgmental reconstruction of the system $S4$ the approach can be used to host other modalities.

4.2 Judgmental Reconstruction of Modal Logic

The $\Box$ fragment of Pfenning's Judgmental reconstruction, consists of the judgments of IPL as developed in the first Chapter together with judgments of validity. The definition of validity is given in a proof theoretic manner. In a nutshell, judgments of validity internalize judgments of proof without assumptions. "Evidence for validity of ϕ, is simply unconditional evidence of truth of ϕ".

1. If $\mathsf{nil} \vdash \phi$ true then ϕ Valid

2. If ϕ Valid then $\Gamma \vdash \phi$ true

The logical judgments are now extended in the form:

$$\phi_1' \text{ Valid}, \phi_2' \text{ Valid}, \ldots, \phi_n' \text{ Valid}; \phi_1 \text{ true} \phi_2 \text{ true}, \ldots \phi_m \text{ true} \vdash \phi \text{ true}$$

In the rules, we restrict ourselves to proving judgments of the form ϕ true (rather than ϕ Valid), which is possible since the latter is directly defined in terms of the former. The meaning of hypothetical judgments yields the general substitution principle.

$\Delta \vdash \phi$ Valid and Δ, ϕ Valid $\vdash J$ then $\Delta \vdash J$

This principle can be rewritten utilizing the definition of validity:

Substitution Principle For Validity $\Delta; \mathsf{nil} \vdash \phi \text{true}$ and $\Delta, \phi\mathsf{Valid}; \Gamma \vdash J$ then $\Delta; \Gamma \vdash J$

Additionally, we add the following hypothesis reflection rule for valid contexts:

Validity Context Reflection

$$\frac{\Delta \text{ ctx} \qquad \Gamma \text{ ctx} \qquad \phi \text{ Valid} \in \Delta}{\Delta; \Gamma \vdash \phi \text{ true}} \; \Delta\text{-Refl}$$

In the typability rules we add the following:

$$\frac{\phi \text{ Prop}}{\Box \phi \text{ Prop}} \; \Box\text{F}$$

The $\Box$ introduction rule just allows the internalization of the validity of ϕ as truth of $\Box\phi$, according to the definition of validity.

Necessity Introduction

$$\frac{\Delta; \text{ nil} \vdash \phi \text{ true}}{\Delta; \Gamma \vdash \Box\phi \text{ true}} \; \Box\text{I}$$

The elimination rule is harder. A simplified version like the one below is unsound since the hypotheses in Γ are unjustified.:

$$\frac{\Delta; \Gamma \vdash \Box\phi \text{ true}}{\Delta; \vdash \Box\phi \text{ true}} \; \Box\text{E-Unsound}$$

Another approach would be the rule below. Which is locally sound but not complete Gentzen's inversion principle is not satisfied. After eliminating

the $\square$ we cannot re-introduce it.

$$\frac{\Delta; \vdash \square\phi \text{ true}}{\Delta; \Gamma \vdash \phi \text{ true}} \ \square\text{E-Incomplete}$$

The proposed rule that satisfies local reduction and local expansion is :

$$\frac{\Delta; \Gamma \vdash \square\phi \text{ true} \qquad \Delta, \phi \text{ Valid}; \Gamma \vdash \psi \text{ true}}{\Delta; \Gamma \vdash \psi \text{ true}} \ \square\text{E}$$

The negative fragment of the system is, thus, as follows:

Prop Formation

$$\frac{}{P_i \in \text{Prop}} \ \text{Atom} \qquad \frac{}{\top \in \text{Prop}} \ \text{Top} \qquad \frac{\phi_1 \in \text{Prop} \qquad \phi_2 \in \text{Prop}}{\phi_1 \supset \phi_2 \in \text{Prop}} \ \text{Arr}$$

Context Γ Formation

$$\frac{}{\text{nil} \ \text{ctx}} \ \text{Nil} \qquad \frac{\Gamma \ \text{ctx} \qquad \phi \in \text{Prop}}{\Gamma, \phi \text{ true} \ \text{ctx}} \ \Gamma\text{-Add}$$

Context Δ Formation

$$\frac{}{\mathsf{nil}\ \mathsf{ctx}}\ \mathrm{N_{IL}} \qquad\qquad \frac{\Delta\ \mathsf{ctx} \qquad \phi \in \mathsf{Prop}}{\Delta, \phi\ \mathsf{Valid}\ \mathsf{ctx}}\ \Delta\text{-}\mathrm{A_{DD}}$$

Compound $\Gamma; \Delta$ Context

$$\frac{\Delta\ \mathsf{ctx} \qquad \Gamma\ \mathsf{ctx}}{\Delta; \Gamma \vdash \mathsf{ctx}}\ \Gamma; \Delta\text{-}\mathrm{F}$$

Context Γ Reflection

$$\frac{\Delta; \Gamma\ \mathsf{ctx} \qquad \phi\ \mathsf{true} \in \Gamma}{\Delta; \Gamma \vdash \phi\ \mathsf{true}}\ \Gamma\text{-}\mathrm{R_{EFL}}$$

Context Δ Reflection

$$\frac{\Delta; \Gamma\ \mathsf{ctx} \qquad \phi\ \mathsf{Valid} \in \Delta}{\Delta; \Gamma \vdash \phi\ \mathsf{true}}\ \Delta\text{-}\mathrm{R_{EFL}}$$

Top Introduction

$$\frac{}{\Delta; \Gamma \vdash \top\ \mathsf{true}}\ \top\mathrm{I}$$

Implication Introduction and Elimination

$$\frac{\Delta; \Gamma, \phi_1 \text{ true} \vdash \phi_2 \text{ true}}{\Delta; \Gamma \vdash \phi_1 \supset \phi_2 \text{ true}} \supset\!\text{I} \qquad \frac{\Delta; \Gamma \vdash \phi_1 \supset \phi_2 \text{ true} \qquad \Delta; \Gamma \vdash \phi_1 \text{ true}}{\Delta; \Gamma \vdash \phi_2 \text{ true}} \supset\!\text{E}$$

Necessity Introduction and Elimination

$$\frac{\Delta; \text{ nil} \vdash \phi \text{ true}}{\Delta; \Gamma \vdash \Box\phi \text{ true}} \Box\text{I} \qquad \frac{\Delta; \Gamma \vdash \Box\phi \text{ true} \qquad \Delta, \phi \text{ Valid}; \Gamma \vdash \psi \text{ true}}{\Delta; \Gamma \vdash \psi \text{ true}} \Box\text{E}$$

4.2.1 Properties of Entailment For the Judgmental Reconstruction

The guiding transitivity principles, weakening, contraction, and exchange can be proved for the system:

Transitivity The guiding substitution principle can be expressed as a property of this formal system and also be proven by induction over the structure of derivations

- If $\Delta; \Gamma, \phi \text{ true}, \Gamma' \vdash \psi \text{ true}$ and $\Delta; \Gamma \vdash \phi \text{ true}$ then $\Delta; \Gamma, \Gamma' \vdash \psi \text{ true}$

- If $\Delta, \phi \text{ Valid}, \Delta'; \Gamma \vdash \psi \text{ true}$ and $\Delta; \text{ nil} \vdash \phi \text{ true}$ then $\Delta\Delta'; \Gamma \vdash \psi \text{ true}$

Weakening, Contraction and Exchange properties can be also shown to hold.

4.2.2 Adding proof terms

The system can be assigned proof terms as follows:

Context Γ Formation

$$\frac{}{\text{nil ctx}} \; \text{N\textsc{il}} \qquad \frac{\Gamma \; \text{ctx} \qquad \phi \in \text{Prop} \qquad x \notin \Gamma}{\Gamma, x : \phi \; \text{ctx}} \; \Gamma\text{-A\textsc{dd}}$$

Context Δ Formation

$$\frac{}{\text{nil ctx}} \; \text{N\textsc{il}} \qquad \frac{\Delta \; \text{ctx} \qquad \phi \in \text{Prop} \qquad s \notin \Delta}{\Delta, s :: \phi \; \text{ctx}} \; \Delta\text{-A\textsc{dd}}$$

Compound $\Gamma; \Delta$ Context

$$\frac{\Delta \; \text{ctx} \qquad \Gamma \; \text{ctx}}{\Delta; \Gamma \vdash \text{ctx}} \; \Gamma; \Delta\text{-F}$$

Context Γ Reflection

$$\frac{}{\Delta; \Gamma, x : \phi, \Gamma' \vdash x : \phi} \; \Gamma\text{-R\textsc{efl}}$$

Context Δ Reflection

$$\frac{}{\Delta, s :: \phi, \Delta'; \Gamma \vdash s : \phi} \; \Delta\text{-Refl}$$

Top Introduction

$$\frac{}{\Delta; \Gamma \vdash \langle \rangle : \top} \; \top\text{I}$$

Implication Introduction and Elimination

$$\frac{\Delta; \Gamma, x : \phi_1 \vdash M : \phi_2}{\Delta; \Gamma \vdash \lambda x : \phi_1.M : \phi_1 \supset \phi_2} \; \supset\text{I}$$

$$\frac{\Delta; \Gamma \vdash M : \phi_1 \supset \phi_2 \qquad \Delta; \Gamma \vdash N : \phi_1}{\Delta; \Gamma \vdash (MN) : \phi_2} \; \supset\text{E}$$

Necessity Introduction and Elimination

$$\frac{\Delta; \mathsf{nil} \vdash M : \phi}{\Delta; \Gamma \vdash \mathrm{box}(M) : \Box\phi \; \mathsf{true}} \; \Box\text{I}$$

$$\frac{\Delta; \Gamma \vdash M : \Box\phi \; \mathsf{true} \qquad \Delta, s :: \phi; \Gamma \vdash N : \psi}{\Delta; \Gamma \vdash \mathrm{let} \, \mathrm{box}(s) = M \, \mathrm{in} \, N : \psi} \; \Box\text{E}$$

4.3   Computational Interpretation

One of the possible ways to read $\Box\phi$ is as representing *source* code of type ϕ. This makes the *Box* calculus given a framework for typing programs with explicit *staged computation*. Explicit staging exists in many languages. One of its most characteristic implementations is the *quote* constructs in Lisp [10]. We introduce the concept and the application of the calculus with a motivating example following [28].

Consider the exponential function exp : $nat \rightarrow nat \rightarrow nat$ and the two definitions

```
exp(0) = lm x -> 1
exp(s(n)) = lm x -> x* exp n x
```

```
exp'(0) = lm x -> 1
exp(s(n)) = let f = exp'n in lm x -> x* f x
```

The two functions although behaviorally equivalent have a completely different operational behavior. For the first function applied to a $s(s(0))$ will unfold to

```
lm x-> x* exp(s(0)) x
```

the second though recurs completely on its argument unfolding to:

```
lm x-> x* (lm x-> x *( lm x->1) x) x
```

which after reduction under λ can be reduced to:

```
lm x-> x* x*1
```

We can see that the second version does a lot more computation than the first. However, if the resulting function is applied many times, to many different bases, then the second can be more efficient. We want to extend our language with types that discriminate between the two cases.

We will try to explore this and show the how $\square$ types can be useful to discriminate between the two. Prior to this we have to speak about about operational semantics.

4.3.1   Small Step Semantics

Small steps semantics, is a transition system that describes how an abstract state machine would execute well typed programs expressed in a λ calculus. Small steps semantics gives local reductions and follows a deterministic evaluation principle. Other kind of operational semantics exist (e.g big step or non-deterministic. The Church-Rosser theorem for a calculus can give a proof that all evaluation strategies are equivalent for a calculus). We also need a notion of value. That is a term that accepts no more reductions under our strategy. We work here with call-by-value semantics. That is functions in a λ form are not further reduced and when the term is a function call the arguments are reduced to values before application.

Here is an example of transition system for small step semantics of the negative fragment of IPL:

$$\frac{}{\langle\rangle \; value} \qquad \frac{}{(\lambda x : \phi.M) \; value} \qquad \frac{M \; value}{(\lambda x : \phi.N)M \mapsto [M/x]N}$$

$$\frac{M \; value \qquad N \; value}{\text{fst}\langle M, N \rangle \mapsto M} \qquad \frac{N \; value \qquad N \; value}{\text{snd}\langle M, N \rangle \mapsto M}$$

$$\frac{M \mapsto M'}{\langle M, N \rangle \mapsto \langle M', M \rangle} \qquad \frac{M \; value \qquad N \mapsto N'}{\langle M, N \rangle \mapsto \langle M, N' \rangle} \qquad \frac{M \mapsto M'}{\text{fst}(M) \mapsto fst(M')}$$

$$\frac{M \mapsto M'}{\text{snd}(M) \mapsto snd(M')} \qquad \frac{M \mapsto M'}{(MN) \mapsto (M'N)} \qquad \frac{M value \qquad N \mapsto N'}{(MN) \mapsto (MN')}$$

Now we have *preservation* and *progress* property. Those are standard properties for any small step semantics transition system. They are formulated only on closed terms because, unlike the process of proof reduction, we only evaluate expressions that are closed.

Preservation If $\text{nil} \vdash M : \phi$ and $M \mapsto M'$ then $\text{nil} \vdash M' : \phi$

Progress If $\text{nil} \vdash M : \phi$ then either $\exists M'.M \mapsto M'$ or $M \; value$

Finally we have the *weak normalization theorem* or *termination theorem.* That is pertinent to the specific choice of semantics. In the simply typed lambda calculus the reduction strategy does not change the normalization property. That is *strong normalization* can be shown. Moreover, from the

Church–Rosser theorem and the normalization property one can deduce the existence and unicity of canonical forms. For other systems this might not be the case.

> **Termination** If $\mathsf{nil} \vdash M : \phi$ then $\exists V.V$ *value and* $M \mapsto^* V$. Where $\mapsto^*$ is the reflexive, transitive closure of $\mapsto$

4.3.2 Operational Semantics for Source Expressions

We extend the computational interpretation sketched above to encompass the necessity modality The interpretation goes as follows:

$x : \phi$	x stands for value of type ϕ
$s :: \phi$	s stands for a source expression ϕ
$[\![M/s]\!]N$	substitute the source expression M for s in N
$\mathrm{box}\, M$	quote the closed term M as a source expression
$\mathrm{let}\, \mathrm{box}(s) = M \,\mathrm{in}\, N$	evaluate M up to the (quoted expression of the) form $\mathrm{box}(M')$ and then evaluate $[\![M/s]\!]N$

We add the following values, a congruence rule and a reduction rule:

$$\frac{}{\operatorname{box}(M)\ \mathit{value}} \qquad \frac{M \mapsto M'}{\operatorname{let box}(s) = M \operatorname{in} N \mapsto \operatorname{let box}(u) = M' \operatorname{in} N}$$

$$\frac{}{\operatorname{let box}(s) = \operatorname{box}(M) \operatorname{in} N \mapsto [\![M/s]\!]N}$$

The importance of the typing is explained by Pfenning:

> The crucial restriction of the typing rules ensures that in an expression $\operatorname{box}(M)$, the term M does not refer to any free variables x that stand for values. It can, however, mention variables s that stand for source expressions. So when we substitute $[\![N/s]\!]\operatorname{box}(M)$ then we are building a larger source expression from two smaller ones, N and M. Conversely, when we substitute a value $[V/x]\operatorname{box} M = \operatorname{box}(M)$ the source expression is not affected.

Returning to our example, the system is able to discriminate between the two examples. The first version of exp still has type $nat \to (nat \to nat)$ whereas the second can be rewritten in the new syntax and has type $nat \to \Box(nat \to nat)$. It is crucial that the first version fails to be written as a source code generator since if we try to re-implement the definition as:

```
exp(s(n))= box(lm x-> x* exp n x)
```

the expression is ill-typed due to the reference to the value variable n. The second version can be written in our extension of the language as a code generator and it is well typed:

```
exp'(0)= box(lm x-> 1): Box(nat -> nat)
exp(s(n))= let box(f)= exp'(n) in box(lm x-> x* f  x): Box(nat -> nat)
```

The computational reading sheds new light to modal theorems as programming combinators. The canonical inhabitant of Axiom 4 of modal logic (seen in the Curry–Howard fashion) is the type of the polymorphic metaprogram that quotes a quoted source code expression:

$$\frac{D}{quote = \lambda x : \Box\phi.\, \text{let box}(s) = x \text{ in box box}(x) : \Box\phi \supset \Box\Box\phi}$$

The K axiom corresponds to the combinator that applies applicative source expressions and results to a larger source expression:

$$\lambda x : \Box(\phi \supset \psi).\lambda y : \Box\phi.\, \text{let box}(s) = x \text{ in let box}(t) = y \text{ in box}(st) : \Box(\phi \supset \psi) \supset \Box\phi \supset \Box\psi$$

Finally the factivity axiom corresponds to unquoting a quoted source code expression:

$$\frac{D}{unquote = \lambda x : \Box\phi.\, \text{let box}(s) = x \text{ in } s : \Box\phi \supset \phi}$$

These operations resemble monadic combinators in languages like Haskell or Scala (cf. [35],[36]). The connection of modality and monadic computation is thoroughly explored in [23]. This discussion, pushes towards a judgmental reconstruction of the possibility modality which we will not be discussing here.

Bibliography

[1] S Artemov. Unified semantics for modality and lambda terms via proof polynomials. *Logic, Language and Computation*, 97.

[2] Sergei Artemov and Melvin Fitting. Justification logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Fall 2012 edition, 2012.

[3] Sergei Artemov and Roman Kuznets. Logical omniscience as infeasibility. *Annals of Pure and Applied Logic*, 165(1):6–25, 2014.

[4] Sergei N. Artemov. Operational modal logic. Technical Report MSI 95–29, Cornell University, December 1995.

[5] Sergei N Artemov. Operational modal logic. 1995.

[6] Sergei N. Artemov. Explicit provability and constructive semantics. *Bulletin of Symbolic Logic*, 7(1):1–36, March 2001.

[7] Sergei N. Artëmov and Eduardo Bonelli. The intensional lambda calculus. In *LFCS*, pages 12–25, 2007.

[8] Sergei Nikolaevich Artemov. Kolmogorov and gödel's approach to intuitionistic logic: current developments. *Russian Mathematical Surveys*, 59(2):203, 2004.

[9] H. P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics.* Sole Distributors for the U.S.A. And Canada, Elsevier Science Pub. Co., 1984.

[10] Alan Bawden et al. Quasiquotation in lisp. In *PEPM*, pages 4–12. Citeseer, 1999.

[11] Nick Benton, Gavin Bierman, Valeria de Paiva, and Martin Hyland. *Term Assignment for Intuitionistic Linear Logic: Preliminary Report.* 1992.

[12] Luitzen Egbertus Jan Brouwer and A Heyting. *Collected Works: Vol.: 1.: Philosophy and Foundations of Mathematics.* North-Holland Publishing Company, American Elsevier Publishing Company, Incorporated, 1975.

[13] Michael AE Dummett. *Elements of intuitionism*, volume 39. Oxford University Press, 2000.

[14] Melvin Fitting. *Proof methods for modal and intuitionistic logics*, volume 169. Springer, 1983.

[15] Melvin Fitting. The logic of proofs, semantically. *Annals of Pure and Applied Logic*, 132(1):1–25, 2005.

[16] Gerhard Gentzen. The collected papers of gerhard gentzen. 1970.

[17] Jean-Yves Girard, Paul Taylor, and Yves Lafont. *Proofs and types*, volume 7. Cambridge University Press Cambridge, 1989.

[18] Robert Harper. Homotopy type theory seminar. `http://www.cs.cmu.edu/~rwh/courses/hott/notes/notes_week1.pdf`.

[19] Robert Harper. Extensionality, intensionality, and Brouwer's dictum. `http://existentialtype.wordpress.com/2012/08/11/extensionality-intensionality-and-brouwers-dictum/`, August 2012.

[20] Robert Harper. Constructive mathematics is not metamathematics. `http://existentialtype.wordpress.com/2013/07/10/constructive-mathematics-is-not-meta-mathematics/`, July 2013.

[21] S.Balzer H.DeYoung. Homotopy type theory seminar. `http://http://www.cs.cmu.edu/~rwh/courses/hott/`.

[22] Arend Heyting. *Intuitionism: an introduction*, volume 41. Elsevier, 1966.

[23] Satoshi Kobayashi. Monad as modality. *Theoretical Computer Science*, 175(1):29–74, 1997.

[24] Andrey Kolmogorov. O principe tertium non datur. mathematicheskij sbornik 32: 646–667. *English trans. in van Heijenoort [1967, 414-437]*, 1925.

[25] Per Martin-Löf. On the meanings of the logical constants and the justifications of the logical laws. *Nordic Journal of Philosophical Logic*, 1(1):11–60.

[26] Per Martin-Lof and Giovanni Sambin. *Intuitionistic type theory*, volume 17. Bibliopolis Naples, 1984.

[27] Eugenio Moggi. Notions of computation and monads. *Information and computation*, 93(1):55–92, 1991.

[28] Frank Pfenning. Computational interpretations of modalities. `http://www.cs.cmu.edu/~fp/courses/15816-s10/lectures/04-compmodal.pdf`, August 2009.

[29] Frank Pfenning. Lecture notes on harmony. `http://www.cs.cmu.edu/~fp/courses/15317-f09/lectures/03-harmony.pdf`, September 2009.

[30] Frank Pfenning. Lecture notes on natural deduction. `http://www.cs.cmu.edu/~fp/courses/15317-f09/lectures/02-natded.pdf`, August 2009.

[31] Frank Pfenning and Rowan Davies. A judgmental reconstruction of modal logic. *Mathematical structures in computer science*, 11(04):511–540, 2001.

[32] Benjamin C. Pierce. *Types and Programming Languages*. MIT Press, Cambridge, MA, USA, 2002.

[33] Konstantinos Pouliasis and Giuseppe Primiero. J-calc: A typed lambda calculus for intuitionistic justification logic. *Electr. Notes Theor. Comput. Sci.*, 300:71–87, 2014.

[34] Dag Prawitz. Natural deduction: A proof-theoretical study. *AMC*, 10:12.

[35] Philip Wadler. Comprehending monads. *Mathematical Structures in Computer Science*, 2(04):461–493, 1992.

[36] Philip Wadler. The essence of functional programming. In *Proceedings of the 19th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '92, pages 1–14, New York, NY, USA, 1992. ACM.