Logical Methods in Computer Science
Vol. 12(3:13)2016, pp. 1–31
www.lmcs-online.org

Submitted Apr. 5, 2016
Published Sep. 28, 2016

ON NATURAL DEDUCTION FOR HERBRAND CONSTRUCTIVE LOGICS I: CURRY-HOWARD CORRESPONDENCE FOR DUMMETT'S LOGIC LC

FEDERICO ASCHIERI

Institut für Diskrete Mathematik und Geometrie, Technische Universität Wien, Wiedner Hauptstraße 8-10/104, 1040, Vienna, Austria

Abstract. Dummett's logic LC is intuitionistic logic extended with Dummett's axiom: for every two statements the first implies the second or the second implies the first. We present a natural deduction and a Curry-Howard correspondence for first-order and second-order Dummett's logic. We add to the lambda calculus an operator which represents, from the viewpoint of programming, a mechanism for representing parallel computations and communication between them, and from the viewpoint of logic, Dummett's axiom. We prove that our typed calculus is normalizing and show that proof terms for existentially quantified formulas reduce to a list of individual terms forming an Herbrand disjunction.

1. Introduction

We call *Herbrand constructive* any intermediate logic – a logic stronger than intuitionistic but weaker than classical – which enjoys a strong form of Herbrand's theorem: for *every* provable formula $\exists\alpha\, A$, the logic proves as well an Herbrand disjunction

$$A[m_1/\alpha] \vee \ldots \vee A[m_k/\alpha]$$

Of course intuitionistic logic is trivially Herbrand constructive, but classical logic is not: A is arbitrary! In between, there are several interesting logics which do have the property. Yet for Herbrand constructive logics there are no known natural deduction formulations with associated Curry-Howard correspondences, except in trivial cases. We launch here a new series of papers to fill this void.

We begin with Dummett's first-order and second-order logic LC: intuitionistic logic extended with the so-called Dummett linearity axiom

$$(A \to B) \vee (B \to A)$$

LC was introduced by Dummett [16] as an example, in the propositional case, of a many-valued logic with a countable set of truth values. Its propositional fragment is also called

2012 ACM CCS: [**Theory of computation**]: Logic.

Key words and phrases: natural deduction, Dummett logic, Curry–Howard, normalization, Herbrand theorem.

This work was funded by the Austrian Science Fund FWF Lise Meitner grant M 1930–N35.

DOI:10.2168/LMCS-12(3:13)2016

Gödel-Dummett logic, because it is based on the truth definition given in Gödel's seminal paper on many-valued logics [20]. In this case, the logic can be formalized by Corsi's sequent calculus [14] or by the more elegant hypersequent calculus devised by Avron [8], [10]. Surprisingly, Avron's hypersequent calculus does not work for first-order LC: only recently Tiu [34] provided a more involved version of it, which indeed corresponds to LC at the first-order.

1.1. **Hyper Natural Deduction?** In all this story, natural deduction is the great absent. Since it is one of the most celebrated logical deduction systems, the question is: how is that possible?

The first issue is that LC is evidently a non-constructive system: for example, it proves the excluded middle for all negated formulas: $\neg A \vee \neg\neg A$; and Dummett's axiom poses even more problems. As it is well known, natural deduction was put aside by its own inventor, Gentzen, precisely for the reason that he was not able to prove a meaningful normalization theorem for classical natural deduction, whilst he *was* for the intuitionistic case [29]. It indeed took a surprisingly long time to discover suitable reduction rules for classical natural deduction systems with all connectives [21], [7]. Even this accomplishment, however, is still not enough: although Dummett's axiom is classically provable, the known classical natural deduction systems fail to provide a refined computational interpretation of LC. The trouble is that LC proofs are not closed under classical reductions, because during the normalization process instances of Dummett's axiom are replaced by reductio ad absurdum in $\lambda\mu$-calculus [28] and in [15], and by excluded middle in [7].

The second issue is that existential quantifiers are witnessed by multiple terms and so a parallel computational mechanism is desirable. No Curry-Howard correspondence offered a suitable one until very recently [7].

Sequent calculus solves these issues by means of *structural rules*. Classical logic is rendered by allowing more formulas on the righthand side of a sequent; Dummett's LC is rendered by allowing sequences of sequents and a communication mechanism between them. On the contrary, natural deduction usually solves the same issues by means of new *reduction rules*. When one wants to add some new axiom to intuitionistic natural deduction, it is enough to add it straight away or as a rule, and all the ingenuity of the construction lies in the proof transformations associated to the axiom.

Inspired by hypersequents, Baaz, Ciabattoni and Fermüller [9] did not follow the latter path and changed instead the very structure of natural deduction into an hyper version. The resulting logical calculus is an *hyper natural deduction* corresponding to Gödel-Dummett first-order logic (which is not to be confused with first-order LC and can be axiomatized by adding to LC the axiom scheme $\forall\alpha\,(A \vee B) \to \forall\alpha\, A \vee B$, where α does not occur in B). The Normal Form Theorem, however, is only obtained by translation into the hypersequent calculus, followed by cut-elimination and backward translation: no reduction rules for hyper deductions were provided. This last task was carried out by Beckmann and Preining [11], who formulated a *propositional* hyper natural deduction with a proof normalization procedure. Unfortunately, the structural rules are so complicated that the adjective "natural" does not fit any more. Another attempt along the "hyper line" has been made by Hirai [22], with the addition of an associated lambda calculus. One cannot speak of a Curry-Howard correspondence, however, because Subject reduction does not hold: there is no match between computational steps and proof reductions.

1.2. **Natural Deduction Again.** Although hyper natural deduction is a legitimate proof system in its own right, the "hyper approach" is not the one we follow. For two reasons.

The first reason is that we will show that natural deduction works perfectly as it is. There is no need to change its structure and, to render Dummett's axiom, it sufficient to add the inference rule

$$\dfrac{\begin{array}{cc}[A \to B] & [B \to A] \\ \vdots & \vdots \\ C & C\end{array}}{C}\ \mathsf{D}$$

which allows to conclude unconditionally C from two different deductions of C: one from the hypothesis $A \to B$ and one from the hypothesis $B \to A$. We shall define simple reduction rules for proofs ending with this inference and we shall show that they are all we need to extract witnesses for existentially quantified formulas.

The second reason is that natural deduction should stay *natural*. This is the very motivation that led to its discovery. Indeed, Gentzen starts his celebrated work [18] on natural deduction and sequent calculus complaining that the proof systems known at the time were far removed from the actual mathematical reasoning. And his main goal was to set up a formalism with the aim of "*reproducing as precisely as possible the real logical reasoning in mathematical proofs*". To avoid betraying natural deduction's philosophical motivations, there is no alternative but to add an inference rule that naturally mirrors the kind of reasoning corresponding to Dummett's axiom, which is our approach.

1.3. **Realizability.** One of the most attractive features of intuitionistic natural deduction is that, in a very precise sense, it does not need a truth-based semantics. Logical inferences are divided into two groups: introduction rules and elimination rules. And as Gentzen [18] himself famously suggested, introduction rules *define*, so to speak, the meaning of the logical constants they introduce; elimination rules, on the other hand, are nothing but *consequences* of these definitions. In other words, introduction rules are self-justifyng, because they fix themselves the meaning of their conclusions, whereas elimination rules are sound in virtue of the meaning fixed by the introductions. For example, the rule

$$\dfrac{\begin{array}{c}[A] \\ \vdots \\ B\end{array}}{A \to B}$$

says that the grounds for asserting $A \to B$ consist in a proof of B from the hypothesis A; therefore, the elimination

$$\dfrac{A \to B \qquad A}{B}$$

is automatically justified: if we have a proof of A we can plug it into the proof of B from A, whose existence is warranted by the meaning of $A \to B$, and obtain a proof of B. The reverse approach works as well: we may consider elimination rules as meaning constitutive and treat introduction rules as consequences of the meaning fixed by eliminations. In other words, meaning is determined by *how we use* a statement, by what we can directly *obtain* from the statement; we shall adopt this *pragmatist* standpoint, elaborated by Dummett himself [17].

This idea of internal justification, as it is, cannot be generalized straight away for extensions of intuitionistic logic: new inferences tend to break the harmony between introductions and eliminations. It is at this point that Brouwer's view of logic comes into play. According to Brouwer [12], the string of "logical" steps appearing in a mathematical proof is in reality a sequence of mathematical constructions. What we perceive as inference rules are instead transformations of constructions for the premises into constructions for the conclusion. This insight finds a precise formalization by means of the Curry-Howard isomorphism: a proof is indeed isomorphic to an effective construction, in fact, *it is*, in and of itself, a construction.

Since proofs are constructions, the role of semantics is just *explaining what these constructions do*. Hence, a proof-theoretic semantics of an intermediate logic is in principle always possible and is made of two ingredients: a formalization of *proofs as programs* and a semantical description of what these programs achieve with their calculations. The first is obtained through the decoration of deduction trees with lambda terms, the second is the task of *realizability*.

Realizability was introduced by Kleene [23] to computationally interpret intuitionistic first-order Arithmetic, but it is Kreisel's [24] later version with typed terms which embodies the modern perspective on the subject. Though it was initially conceived just for intuitionistic theories, realizability can be extended to intuitionistic Arithmetic with Markov's principle [6], to intuitionistic Arithmetic with the simplest excluded middle EM_1 [4] and even all the way up to the strongest classical theories [1, 2, 26]. Realizability replaces the notion of truth with the notion of *constructive evidence*. A formula holds if it is *realized* by some typed program, providing some constructive information about the formula.

In the following, we shall build a realizability interpretation for Dummett's LC, inspired by Krivine's realizability [26, 15]. By construction, every realizer always terminates its computations and, in particular, whenever it realizes an existentially quantified formula $\exists\alpha\, A$, it reduces to a term of the shape

$$(m_0, v_0) \parallel_{a_1} (m_1, v_1) \parallel_{a_2} \dots \parallel_{a_k} (m_k, v_k)$$

with the property that

$$\mathsf{LC} \vdash A[m_1/\alpha] \vee \dots \vee A[m_k/\alpha]$$

The circle is closed by a soundness theorem, the Adequacy Theorem: every formula provable in LC is realized by a closed program, which immediately implies the Normalization Theorem – every proof reduces to a normal form – and that LC is Herbrand constructive. Therefore, to extract an Herbrand disjunction it suffices to reduce any proof of any existentially quantified formula to a normal form, according to a very simple set of reduction rules.

1.4. **Reduction Rules.** To find simple and terminating reduction rules for a natural deduction system is always tricky, but once the job is done, the reductions often look so natural that they appear inevitable. It is the effort of removing obstacles toward a good normal form what inevitably leads to these reductions, as the flow of a river leads to the sea. In the case of LC, the main obstacles toward witness extraction for a formula $\exists\alpha\, A$ are configurations in which one of the hypotheses introduced by the Dummett inference blocks the reduction. For example, let us consider this proof shape:

$$\dfrac{\dfrac{\dfrac{\vdots}{[A\to B]\qquad A}}{\dfrac{B}{\dfrac{}{\exists\alpha\, C}\,\mathsf{EL}}}\qquad [B\to A]\quad\mathcal{D}\quad\exists\alpha\, C}{\exists\alpha\, C}\,\mathsf{D}$$

where $\exists\alpha\, C$ has been obtained from B by a series of elimination rules. It is clear that no witness can be retrieved in the left branch of the proof above, because there is just a proof of A and, magically, a "void" proof of B obtained by modus ponens from A and the arbitrary hypothesis $A\to B$. But can't we just send the proof of A to the right branch of the Dummett rule and obtain a direct proof of $\exists\alpha\, C$, like this?

$$\dfrac{\dfrac{\vdots}{A}}{B\to A}\quad\mathcal{D}\quad\exists\alpha\, C$$

No! In fact, the proof of A too might depend on the hypothesis $A\to B$, so that the original proof could be

$$[A\to B]$$

$$\dfrac{\dfrac{\dfrac{\vdots}{[A\to B]\qquad A}}{\dfrac{B}{\dfrac{}{\exists\alpha\, C}\,\mathsf{EL}}}\qquad [B\to A]\quad\mathcal{D}\quad\exists\alpha\, C}{\exists\alpha\, C}\,\mathsf{D}$$

and thus the previous transformation is unsound. But the idea of sending the proof of A to the right branch can work if the right branch is in turn moved on the left like this

$$\dfrac{\dfrac{[A\to B]}{\dfrac{\vdots}{A}}}{B\to A}\qquad [B\to A]\quad\mathcal{D}\quad\exists\alpha\, C}{\exists\alpha\, C}\,\mathsf{D}$$

The reductions that we shall give generalize this transformation in order to work in every situation.

1.5. **Curry-Howard Correspondence.** It is more convenient to express proof reductions in terms of program reductions, because for that purpose the lambda notation is superior to the proof tree notation. For this reason, we shall define a lambda calculus isomorphic to natural deduction for LC and then define an *head reduction strategy* for lambda terms, inspired by Krivine's strategy [26]. The termination of head reduction will just be a consequence of soundness of LC with respect to realizability, while the perfect match between program reductions and proof reductions will as usual be consequence of the Subject Reduction Theorem. The decoration of intuitionistic inferences with programs is standard and Dummett's rule will be decorated in the following way

$$\dfrac{\begin{array}{cc}[a^{A\to B}: A\to B] & [a^{B\to A}: B\to A]\\ \vdots & \vdots\\ u : C & v : C\end{array}}{u \parallel_a v : C}\ \mathsf{D}$$

The parallel operator $\parallel_a$ is inspired by the exception operator studied in [7] and keeps using the variable a for communication purposes. The variable a has the task of sending terms from u to v and viceversa, as well as allowing u to call the process v whenever it needs it and viceversa.

1.6. **Plan of the Paper.** In Section §2 we introduce a Curry-Howard interpretation of intuitionistic first-order natural deduction extended with the Dummett rule D. We first describe the calculus together with its computational rules and then discuss its proof theoretical interpretation.
In Section §3 we prove the Normalization Theorem and the soundness of realizability with respect to LC.
In Section §4, we prove that LC is Herbrand constructive and in particular that from any closed term having as type an existentially quantified formula, one can extract a corresponding Herbrand disjunction.
In Section §5 we extend the previous results to the second-order LC_2, achieving its first computational interpretation, for there is no known cut-elimination procedure for second-order hypersequent calculus.

2. The System LC

In this section we describe a standard natural deduction system for intuitionistic first-order logic, with a term assignment based on the Curry-Howard correspondence (e.g. see [31]), and add on top of it an operator which formalizes Dummett's axiom. First, we shall describe the lambda terms and their computational behavior, proving as main result the Subject Reduction Theorem, stating that the reduction rules preserve the type. Then, we shall analyze the logical meaning of the reductions and present them as pure proof transformations.

We start with the standard first-order language of formulas.

Definition 2.1 (Language of LC)**.** The language $\mathcal{L}$ of LC is defined as follows.

(1) The **terms** of $\mathcal{L}$ are inductively defined as either variables $\alpha, \beta, \ldots$ or constants c or expressions of the form $\mathsf{f}(m_1, \ldots, m_n)$, with f a function constant of arity n and $m_1, \ldots, m_n \in \mathcal{L}$.
(2) There is a countable set of **predicate symbols**. The *atomic formulas* of $\mathcal{L}$ are all the expressions of the form $\mathcal{P}(m_1, \ldots, m_n)$ such that $\mathcal{P}$ is a predicate symbol of arity n and $m_1, \ldots, m_n$ are terms of $\mathcal{L}$. We assume to have a 0-ary predicate symbol $\perp$ which represents falsity.
(3) The **formulas** of $\mathcal{L}$ are built from atomic formulas of $\mathcal{L}$ by the logical constants $\vee, \wedge, \to, \forall, \exists$, with quantifiers ranging over variables $\alpha, \beta, \ldots$: if A, B are formulas, then $A \wedge B$, $A \vee B$, $A \to B$, $\forall\alpha\, A$, $\exists\alpha\, B$ are formulas. The logical negation $\neg A$ can be introduced, as usual, as a shorthand for the formula $A \to \perp$.

In Figure 1 we define a type assignment for lambda terms, called **proof terms**, which is isomorphic to natural deduction for intuitionistic logic extended with Dummett's axiom.

Axioms: $x^A : A$

Conjunction: $\dfrac{u : A \quad t : B}{\langle u, t\rangle : A \wedge B}$ $\qquad \dfrac{u : A \wedge B}{u\,\pi_0 : A}$ $\qquad \dfrac{u : A \wedge B}{u\,\pi_1 : B}$

Implication: $\dfrac{t : A \to B \quad u : A}{tu : B}$ $\qquad \dfrac{\begin{array}{c}[x^A : A]\\ \vdots \\ u : B\end{array}}{\lambda x^A u : A \to B}$

Disjunction Introduction: $\dfrac{u : A}{\iota_0(u) : A \vee B}$ $\qquad \dfrac{u : B}{\iota_1(u) : A \vee B}$

Disjunction Elimination: $\dfrac{\begin{array}{ccc} & [x^A : A] & [y^B : B] \\ & \vdots & \vdots \\ u : A \vee B & w_1 : C & w_2 : C\end{array}}{u\,[x^A.w_1, y^B.w_2] : C}$

Universal Quantification: $\dfrac{u : \forall\alpha\, A}{um : A[m/\alpha]}$ $\qquad \dfrac{u : A}{\lambda\alpha\, u : \forall\alpha\, A}$

where m is any term of the language $\mathcal{L}$ and α does not occur free in the type B of any free variable x^B of u.

Existential Quantification: $\dfrac{u : A[m/\alpha]}{(m, u) : \exists\alpha\, A}$ $\qquad \dfrac{\begin{array}{cc} & [x^A : A] \\ & \vdots \\ u : \exists\alpha\, A & t : C\end{array}}{u\,[(\alpha, x^A).t] : C}$

where α is not free in C nor in the type B of any free variable of t.

Dummett's Axiom D**:** $\dfrac{\begin{array}{cc} [a^{A\to B} : A \to B] & [a^{B\to A} : B \to A] \\ \vdots & \vdots \\ u : C & v : C\end{array}}{u \parallel_a v : C}\,\mathsf{D}$

Ex Falso Quodlibet: $\dfrac{\Gamma \vdash u : \bot}{\Gamma \vdash \mathsf{efq}_P(u) : P}$

with P atomic.

Figure 1: Term Assignment Rules for LC

We assume that in the proof terms two distinct classes of variables appear. The first class of variables is made by the variables for the proof terms themselves: for every formula A, we have variables $x_0^A, x_1^A, x_2^A, \ldots$ of type A; these variables will be denoted as $x^A, y^A, z^A \ldots, a^A, b^A$ and whenever the type is not important simply as $x, y, z, \ldots, a, b$. For

clarity, the variables introduced by the Dummett's inference rule will be denoted with letters $a, b, \ldots$, but they are not in any syntactic category apart. The second class of variables is made by the quantified variables of the formula language $\mathcal{L}$ of LC, denoted usually as $\alpha, \beta, \ldots$.

The free and bound variables of a proof term are defined as usual and for the new term $u \parallel_a v$, all of the free occurrences of a in u and v are bound in $u \parallel_a v$. In the following, we assume the standard renaming rules and alpha equivalences that are used to avoid capture of variables in the reduction rules that we shall give.

Whenever $\Gamma = x_1 : A_1, \ldots, x_n : A_n$ and the list $x_1, \ldots, x_n$ includes all the free variables of a proof term $t : A$, we shall write $\Gamma \vdash t : A$. From the logical point of view, the notation means that t represents a natural deduction of A from the hypotheses $A_1, \ldots, A_n$. We shall write $\mathsf{LC} \vdash t : A$ whenever $\vdash t : A$, and the notation means provability of A in intuitionistic logic with Dummett's axiom.

We are now going to explain the basic reduction rules for the proof terms of LC, which are given in Figure 2. To understand them, we need the notions of parallel context and stack. If we omit parentheses, any term t can be written, not uniquely, in the form

$$t = t_1 \parallel_{a_1} t_2 \parallel_{a_2} \ldots \parallel_{a_n} t_{n+1}$$

If we replace some t_i with a "hole" [] to be filled, the expression above becomes a parallel context.

Definition 2.2 (Parallel Contexts)**.** Omitting parentheses, a **parallel context** $\mathcal{C}[\ ]$ is an expression of the form

$$u_1 \parallel_{a_1} u_2 \parallel_{a_2} \ldots u_i \parallel_{a_i} [] \parallel_{a_{i+1}} u_{i+1} \parallel_{a_{i+2}} \cdots \parallel_{a_n} u_n$$

where [] is a placeholder and $u_1, u_2, \ldots, u_n$ are proof terms. For any proof term u, $\mathcal{C}[u]$ denotes the replacement in $\mathcal{C}[\ ]$ of the placeholder [] with u:

$$u_1 \parallel_{a_1} u_2 \parallel_{a_2} \ldots u_i \parallel_{a_i} u \parallel_{a_{i+1}} u_{i+1} \parallel_{a_{i+2}} \cdots \parallel_{a_n} u_n$$

A stack represents, from the logical perspective, a series of elimination rules; from the lambda calculus perspective, a series of either operations to be performed or arguments to be given as input to some program. A stack is also known as a *continuation*, because it embodies a series of tasks that wait to be executed, and corresponds to Krivine's stacks [26].

Definition 2.3 (Stack)**.** A **stack** is a sequence

$$\sigma = \sigma_1 \bullet \sigma_2 \bullet \ldots \bullet \sigma_n$$

such that for every $1 \leq i \leq n$, exactly one of the following holds:

- $\sigma_i = t$, with t proof term.
- $\sigma_i = m$, with $m \in \mathcal{L}$.
- $\sigma_i = \pi_j$, with $j \in \{0, 1\}$.
- $\sigma_i = [x.u, y.v]$, with u, v proof terms of the same type.
- $\sigma_i = [(\alpha, x).v]$, with v proof term.

If no confusion with other sequences of terms arises, σ will often be written without intermediate dots, that is, as $\sigma_1\, \sigma_2\, \ldots\, \sigma_n$. The *empty sequence* is denoted with ϵ and with $\xi, \xi', \ldots$ we will denote stacks of length 1. If t is a proof term, as usual in lambda calculus $t\, \sigma$ denotes the term $(((t\, \sigma_1)\, \sigma_2) \ldots \sigma_n)$.

We find among the reductions in Figure 2 the ordinary reductions for the intuitionistic constructs together with Prawitz-style permutation rules [30] for D, as in [7]. The reduction rules for D model the communication mechanism explained in Section §1. In the reduction

$$\mathcal{C}[a^{A\to B}\, u\, \sigma] \parallel_a v \;\mapsto\; \mathcal{C}[\, v[\lambda y^B\, u/a^{B\to A}]\,] \parallel_a v$$

we see that the term on the left is in some way stuck: the variable $a^{A\to B}$ faces an argument u of type A; of course, it has no idea how to use u to produce a term of type B! On the contrary, the term v knows very well how to use u to produce something useful, because it contains the variable $a^{B\to A}$, which waits for a term of type $B \to A$. Thus, $a^{A\to B}$ sends the term $\lambda y^B\, u$, with y dummy, to v, yielding the term $v[\lambda y^B\, u/a^{B\to A}]$. This program is called to replace the useless $a^{A\to B}\, u\, \sigma$ and computation can go ahead. We require the context $\mathcal{C}[\ ]$ to be parallel, because in this way types are not needed to define the reductions for D and the calculus makes sense also in its untyped version and with Curry-style typing. We have chosen Church-typing only to make clearer the intended meaning of the operations: had we omitted all the types from the terms, everything would have still worked just fine. In Theorem 2.7, we shall prove that indeed our reduction rules for D are logically correct and preserve the type.

Reduction Rules for Intuitionistic Logic:

$$(\lambda x\, u)t \mapsto u[t/x]$$
$$(\lambda \alpha\, u)m \mapsto u[m/\alpha]$$
$$\langle u_0, u_1\rangle \pi_i \mapsto u_i, \text{ for } i = 0, 1$$
$$\iota_i(u)[x_1.t_1, x_2.t_2] \mapsto t_i[u/x_i], \text{ for } i = 0, 1$$
$$(m, u)[(\alpha, x).v] \mapsto v[m/\alpha][u/x], \text{ for each term } m \text{ of } \mathcal{L}$$

Permutation Rules for D:

$$(u \parallel_a v)w \mapsto uw \parallel_a vw, \text{ if } a \text{ does not occur free in } w$$
$$(u \parallel_a v)\pi_i \mapsto u\pi_i \parallel_a v\pi_i$$
$$(u \parallel_a v)[x.w_1, y.w_2] \mapsto u[x.w_1, y.w_2] \parallel_a v[x.w_1, y.w_2], \text{ if } a \text{ does not occur free in } w_1, w_2$$
$$(u \parallel_a v)[(\alpha, x).w] \mapsto u[(\alpha, x).w] \parallel_a v[(\alpha, x).w], \text{ if } a \text{ does not occur free in } w_1, w_2$$

Reduction Rules for D:

$$\mathcal{C}[a^{A\to B}\, u\, \sigma] \parallel_a v \;\mapsto\; \mathcal{C}[\, v[\lambda y^B\, u/a^{B\to A}]\,] \parallel_a v$$
$$v \parallel_a \mathcal{C}[a^{A\to B}\, u\, \sigma] \mapsto\; v \parallel_a \mathcal{C}[\, v[\lambda y^B\, u/a^{B\to A}]\,]$$

for some parallel context $\mathcal{C}$, stack σ, variable a free in $\mathcal{C}[a^{A\to B}\, u\, \sigma]$, dummy variable y not occurring in u

Figure 2: Basic Reduction Rules for LC

Our goal now is to define a reduction strategy for typed terms of LC: a recipe for selecting, in any given term, the subterm to which apply one of our basic reductions. As most typed lambda calculi are strongly normalizing and our reduction rules look fairly innocuous, one cannot help but conjecture that any reduction strategy eventually terminates; in other words, that reduction strategies are not necessary. We do conjecture that the fragment with $\forall, \to, \wedge, \vee$ is indeed strongly normalizing. Yet, already the proof of this weaker result appears excessively complex, to such an extent that arbitrary reduction strategies start to feel wrong, that is, to perform unnecessary computations.

We therefore leave strong normalization as an open problem and follow a more standard approach: Krivine's (weak) head reduction strategy. The difference is: in Krivine's calculus

each process has a unique head; in our calculus each process has several heads, like the Hydra monster. This is due to the presence of the parallel operator $\|_a$. Indeed, if we omit parenthesis, any term t can be written, not uniquely, in the form

$$t = t_1 \,\|_{a_1}\, t_2 \,\|_{a_2}\, \dots \,\|_{a_n}\, t_{n+1}$$

The terms $t_1, \dots, t_n$ are parallel processes; each one has its own head and may have an head redex. And as with the Hydra monster, if we contract some head t_i, more heads to contract might grow. We now formally define what are the parallel processes that appear in a term and what is the head redex of a term.

Definition 2.4 (Parallel Processes, Head)**.**

- Removing the parentheses, whenever a proof term t can be written as
$$t = t_1 \,\|_{a_1}\, t_2 \,\|_{a_2}\, \dots \,\|_{a_n}\, t_{n+1}$$
each term t_i, for $1 \leq i \leq n+1$, is said to be a **parallel process** of t and is said to be an **elementary process** of t in case it is not of the form $u \,\|_a\, v$.
- A **redex** is a term u such that $u \mapsto v$ for some v and basic reduction of Figure 2.
- Let σ be any stack. A *redex* h is said to be the **head redex** of a proof term t in the following cases:
  (1) $t = (\lambda x\, u)v\,\sigma$ and $h = (\lambda x\, u)v$;
  (2) $t = (\lambda \alpha\, u)m\,\sigma$ and $h = (\lambda \alpha\, u)m$;
  (3) $t = \langle u, v\rangle \pi_i\,\sigma$ and $h = \langle u, v\rangle \pi_i$;
  (4) $t = \iota_i(u)\,[x_1.t_1, x_2.t_2]\,\sigma$ and $h = \iota_i(u)\,[x_1.t_1, x_2.t_2]$;
  (5) $t = (m, u)\,[(\alpha, x).v]\,\sigma$ and $h = (m, u)\,[(\alpha, x).v]$;
  (6) $t = ((u \,\|_a\, v)\,\xi\,)\,\sigma$ and $h = (u \,\|_a\, v)\,\xi$;
  (7) $t = u \,\|_a\, v$ and $h = t$.

We now define the head reduction of a proof term: the notion generalizes Krivine's head reduction to parallel contexts. The idea is to look for the leftmost among the head redexes of the parallel processes of a term and contract that redex. The only subtlety is to determine exactly where the new redexes for D start. Since the reduction for $u \,\|_a\, v$ is completely localized either in u or v, it is reasonable to say that the redex starts where the subterm $a\,u\,\sigma$ to be replaced is located.

Definition 2.5 (Letfmost Redex, Head Reduction)**.**

(1) The **starting symbol** of a redex r is the symbol "(" when $r = (u\,\xi)$ for some stack ξ of length 1; it is the leftmost occurrence of the symbol "a" such that $a\,t\,\sigma$ is an elementary process of r, when $r = (u \,\|_a\, v)$. The **leftmost redex** among some redexes of a term t is the redex whose starting symbol is the leftmost in t among the starting symbols of those redexes.

(2) We say that a term t **head reduces** to t' and write
$$t \succ t'$$
when t' is obtained from t by contracting the leftmost among the head redexes of the parallel processes of t, using one of the basic reductions in Figure 2.

For readability, parentheses are often omitted, but in order to spot the head redex of a term, one must mentally restore the parentheses that have been suppressed. In order to train our eye, we consider three examples of head reduction:

$$(\lambda x\, a\, (\lambda z\, z)\, x)\, u \;\|_a\; z_0 \succ a\, (\lambda z\, z)\, u \,\|_a\, z_0 \succ z_0 \,\|_a\, z_0$$

$$(\lambda x\, \iota_o(x))\, u\, [x_0.t_1, x_1.t_1]\, \sigma \;\succ\; \iota_o(u)\, [x_0.t_0, x_1.t_1]\, \sigma \;\succ\; t_0[u/x_0]\, \sigma$$

$$a\, ((\lambda x\, x)\, z_0) \,\|_a\, a\, z_1 \succ (\lambda y\, (\lambda x\, x)\, z_0)\; z_1 \,\|_a\, a\, z_1 \succ (\lambda x\, x)\, z_0 \,\|_a\, a\, z_1 \succ z_0 \,\|_a\, a\, z_1 \succ z_0 \,\|_a\, z_0$$

In the first case, the reduction for D is used as third step of the head reduction, while in the third case, as first and last step.

We define the concept of normal form and normalizable term in the usual way.

Definition 2.6 (Normal Forms and Normalizable Terms)**.**

- A term t is called a **head normal form** if there is no t' such that $t \succ t'$. We define NF to be the set of head normal forms.
- A sequence, finite or infinite, of proof terms $u_1, u_2, \ldots, u_n, \ldots$ is said to be a reduction of t, if $t = u_1$, and for all i, $u_i \succ u_{i+1}$. A proof term u of LC is (head) **normalizable** if there is no infinite reduction of u. We denote with HN the set of normalizable terms of LC.

The reductions defined in Figure 2 satisfy the important Subject Reduction Theorem: reduction steps at the level of proof terms preserve the type, which is to say that they correspond to logically sound transformations at the level of proofs. We first give the simple proof of the theorem, then analyze in detail its logical meaning in the next subsection.

Theorem 2.7 (Subject Reduction)**.** *If $t : C$ and $t \succ u$, then $u : C$. Moreover, all the free variables of u appear among those of t.*

Proof. It is enough to prove the theorem for basic reductions: if $t : C$ and $t \mapsto u$, then $u : C$. The proof that the intuitionistic reductions and the permutation rules preserve the type is completely standard. Thus we are left with the D-reductions, which require straightforward considerations as well. Suppose

$$\mathcal{C}[a^{A\to B}\, u\, \sigma] \,\|_a\, v \;\mapsto\; \mathcal{C}[\, v[\lambda y^B\, u/a^{B\to A}]\,] \,\|_a\, v$$

Since $\mathcal{C}$ is a parallel context, $a^{A\to B}\, u\, \sigma$ and v have both type C. Now, u must be of type A, so $\lambda y^B u$ is of type $B \to A$ and thus $v[\lambda y^B\, u/a^{B\to A}]$ is a correct term of type C. Moreover, all the occurrences of $a^{B\to A}$ in v are eliminated by the substitution $[\lambda y^B\, u/a^{B\to A}]$, so no new free variable is created. □

2.1. **Reduction Rules: Logical Interpretation.** So far, in studying the system LC, we have given priority to the underlying lambda calculus and characterized it as a functional language endowed with parallelism and a communication mechanism. The explanation of the reductions had little to do with logic and much with computation. However, thanks to the Subject Reduction Theorem, we know we could have proceeded the other way around. Namely, we could have given priority to logic and dealt only with transformation of proofs, in the style of Prawitz natural deduction trees [30]. Since it is instructive to explain directly this point of view, we are finally going to do so.

First of all, the following proof of $\neg A \vee \neg\neg A$ is an example of natural deduction tree in LC:

$$\dfrac{\dfrac{\dfrac{[\neg A]\quad[A]}{\dfrac{\bot}{\neg\neg A}}}{\dfrac{[\neg\neg A\to\neg A]\qquad\neg\neg A}{\neg A}}\qquad[A]}{\dfrac{\dfrac{\bot}{\neg A}}{\neg A\vee\neg\neg A}}\qquad\dfrac{\dfrac{\dfrac{[\neg A\to\neg\neg A]\qquad[\neg A]}{\neg\neg A}\qquad[\neg A]}{\dfrac{\bot}{\neg\neg A}}}{\neg A\vee\neg\neg A}$$

$$\neg A\vee\neg\neg A\ \mathsf{D}$$

The standard reductions for lambda calculus still correspond to the ordinary conversions for all the logical constants of first-order logic:

$$\dfrac{\dfrac{[A]}{\vdots}\atop B}{\dfrac{A\to B\qquad A}{B}}\quad\text{converts to:}\quad\begin{array}{c}\vdots\\A\\\vdots\\B\end{array}$$

$$\dfrac{\dfrac{\vdots}{A_i}}{\dfrac{A_1\vee A_2}{C}}\,(i\in\{1,2\})\qquad\begin{array}{cc}[A_1]&[A_2]\\\vdots&\vdots\\C&C\end{array}\quad\text{converts to:}\quad\begin{array}{c}\vdots\\A_i\\\vdots\\C\end{array}$$

$$\dfrac{\dfrac{\vdots\quad\vdots}{A_1\qquad A_2}}{\dfrac{A_1\wedge A_2}{A_i}}\,(i\in\{1,2\})\quad\text{converts to:}\quad\begin{array}{c}\vdots\\A_i\end{array}$$

$$\dfrac{\dfrac{\dfrac{\vdots}{A[m/\alpha]}}{\exists\alpha\,A}\qquad\begin{array}{c}[A]\\\pi\\C\end{array}}{C}\quad\text{converts to:}\quad\begin{array}{c}\vdots\\A[m/\alpha]\\\pi[m/\alpha]\\C\end{array}$$

$$\dfrac{\dfrac{\pi}{A}}{\dfrac{\forall\alpha\,A}{A[m/\alpha]}}\quad\text{converts to:}\quad\begin{array}{c}\pi[m/\alpha]\\A[m/\alpha]\end{array}$$

The permutation reductions for the terms of the form $u\parallel_a v$, are just instances of Prawitz-style permutations for disjunction elimination. From the logical perspective, they are used to systematically transform, whenever possible, the logical shape of the conclusion. This reduction is essential because the Dummett inference rule does not yield much when employed to prove implications or disjunctions; but it becomes Herbrand constructive, whenever used to prove existentially quantified statements. As an example of permutation for D, we consider the one featuring an implication as conclusion:

$$\dfrac{\dfrac{\begin{array}{cc}[A\to B] & [B\to A]\\ \vdots & \vdots\\ F\to G & F\to G\end{array}}{F\to G}\,\mathsf{D} \qquad \begin{array}{c}\vdots\\ F\end{array}}{G} \quad \text{converts to:} \quad \dfrac{\dfrac{\begin{array}{cc}[A\to B] & \\ \vdots & \\ F\to G & F\end{array}}{G} \qquad \dfrac{\begin{array}{cc}[B\to A] & \\ \vdots & \\ F\to G & F\end{array}}{G}}{G}$$

There are similar permutations for all other elimination rules, as one can see translating in natural deduction the permutations of Figure 2. With the following notation

$$\dfrac{\begin{array}{ccccc}\mathcal{D}_1 & & \mathcal{D}_i & & \mathcal{D}_n\\ C & \cdots & C & \cdots & C\end{array}}{C}\,\mathsf{D}$$

we denote a deduction of C that, in order to obtain its final conclusion, combines the deductions $\mathcal{D}_1,\ldots,\mathcal{D}_i,\ldots,\mathcal{D}_n$ of C using *only* the Dummett rule $n-1$ times. In other words, below the conclusions C of the deductions $\mathcal{D}_1,\ldots,\mathcal{D}_i,\ldots,\mathcal{D}_n$ only the Dummett rule is used. This configuration corresponds to a parallel context in our lambda calculus, as in Definition 2.2. With the notation

$$\dfrac{B}{C}\,\mathsf{EL}$$

we denote a deduction of C that, starting from B, applies only elimination rules to obtain C; in particular, B must be the main premise of the first elimination rule which concludes B_1, which must be the main premise of the second elimination rule which concludes B_2 and so on down to C. This configuration corresponds to the concept of stack of Definition 2.3.

Finally, we can look at the two reductions for proofs containing the Dummett rule. Let us consider just the first conversion for D, the second being perfectly symmetric:

$$\dfrac{\dfrac{\begin{array}{ccccc}\mathcal{D}_1 & & \dfrac{\begin{array}{cc}[A\to B] & A\\ B & \end{array}}{C}\,\mathsf{EL} & & \mathcal{D}_n\\ C & \cdots & & \cdots & C\end{array}}{C}\,\mathsf{D} \qquad \begin{array}{c}[B\to A]\\ \mathcal{D}\\ C\end{array}}{C}\,\mathsf{D} \quad \text{converts to:} \quad \dfrac{\dfrac{\begin{array}{ccccc}\mathcal{D}_1 & & \dfrac{\begin{array}{c}[A\to B]\\ \vdots\\ A\\ B\to A\end{array}}{C} & & \mathcal{D}_n\\ C & \cdots & & \cdots & C\end{array}}{C}\,\mathsf{D} \qquad \begin{array}{c}[B\to A]\\ \mathcal{D}\\ C\end{array}}{C}\,\mathsf{D}$$

The conversion above focuses first on the deduction $\mathcal{D}$ on the left branch of the proof; it replaces the hypothesis $B\to A$ of $\mathcal{D}$ with a proof of $B\to A$ directly obtained from the proof of A found on the left branch; afterwards, it takes the deduction so generated and replaces with it the old proof of C obtained from B by elimination rules.

There is a crucial assumption about the structure of the first proof. In the left branch of the Dummett rule, the hypothesis $A\to B$ is used together with A to obtain B, which is in turn used to infer C *by means only of a main branch of elimination rules*, as called by Prawitz. Thanks to this restriction, the proof of A does not end up having more open assumptions in the second proof than it has in the first proof.

But what have we gained with this reduction? It looks like we made no progress at all. The hypothesis $A\to B$ may be actually used more times in the second proof than in the first, because the hypothesis $B\to A$ might be used several times in the deduction $\mathcal{D}$! Actually, the gain is subtle. In the left branch of the first proof the formula B was derived in a

fictitious way: by an arbitrary hypothesis $A \to B$, bearing no relationship with C. Since B is *used* to obtain C, we cannot expect B to provide constructive content to C, in particular no witness if C is an existential formula. The conversion above gets rid of this configuration and provide a more direct proof of C: in the new proof, if $B \to A$ is employed to derive A by modus ponens, one can discard B and use the proof of A coming from the first proof.

The main difficulty that we face with our reduction rules for D is *termination*. There is hardly any decrease in complexity from before to after the reduction and the road toward a combinatorial termination proof looks barred. We are thus forced to employ a far more abstract technique: realizability.

3. Classical Realizability

In this section we prove that each term of LC *realizes* its type and is normalizing. To this end, we make a detour into a logically inconsistent, yet computationally sound world: the system $\mathsf{LC}^\star$, a type system which extends LC . The idea that extending a system can make easier rather than harder to prove its normalization might not seem very intuitive, but it is well tested and very successful (see [32], [5], [3], [7]). $\mathsf{LC}^\star$ will be our calculus of realizers. It is indeed typical of realizability, the method we shall use, to set up a calculus with more realizers than the actual proof terms [24, 26, 6]. The idea is that a realizer is defined as a proof term that defeats every opposer and passes every termination test; but proof terms, as opposers and testers of proof terms themselves, are not enough; proof terms must be opposed and tested also by "cheaters", terms that do satisfy the same definition of realizability, but only because they have some advantage. These extra tests make proof terms stronger realizers than they otherwise would be. We may imagine a realizer as a tennis player that trains himself to return fast balls thrown by a robot: if he withstands the attacks of the robot, he will perform all the more well against real weaker humans.

3.1. **The Abort Operator.** The system $\mathsf{LC}^\star$ is not meant to be a logical system: it would be inconsistent! The purpose of the system is not logical, but computational: to simulate the reduction rules for D by an abort operator $\mathcal{A}$. We define the typing rules of $\mathsf{LC}^\star$ to be those of LC plus a new term formation scheme:

Abort Axiom: $\mathcal{A}^{A\to B} : A \to B$

With $\mathcal{A}$, $\mathcal{A}_1, \ldots, \mathcal{A}_k$, we shall denote some generic constant $\mathcal{A}^{A\to B}$. The reduction rules for the terms of $\mathsf{LC}^\star$ are those for LC with the addition of a new reduction rule defined in Figure 3.

Reduction Rules for $\mathcal{A}$:

$$\mathcal{A}\, u\, \sigma \mapsto u$$

whenever $\mathcal{A}\, u\, \sigma$ and u have the same type

Figure 3: Extra Reduction Rules for $\mathsf{LC}^\star$

The abort computational construct reminds Krivine's k_π, which removes the current continuation ρ and restore a previously saved continuation π:

$$\mathsf{k}_\pi \star t \bullet \rho \succ t \star \pi$$

There is indeed an analogy with Krivine's realizability: the terms of LC correspond to Krivine's proof-like terms, whereas the terms of $\mathsf{LC}^\star$ correspond to Krivine's inconsistent terms that may contain k_π and may realize any formula. But in our case $\mathsf{LC}^\star$ is just a *tool for defining realizability*, not a tool for implementing reductions, like k_π in Krivine case. The role of $\mathcal{A}$ will emerge later on in the proof of Propositions 3.6 and 3.7. However, by now, the intuition should be pretty clear: in the reduction

$$\mathcal{C}[a\,u\,\sigma] \parallel_a v \;\mapsto\; \mathcal{C}[\,v[\lambda y\,u/a]\,] \parallel_a v$$

the term $a\,u\,\sigma$ aborts the local continuation σ. The difficulty is that the new continuation $v[\lambda y\,u/a]$, from the perspective of a, is created out of nowhere! Therefore proving by induction that $\mathcal{C}[a\,u\,\sigma]$ is a realizer would not be of great help for proving that the whole term $\mathcal{C}[a\,u\,\sigma] \parallel_a v$ is realizer. With terms of the form $\mathcal{A}\,w$ we can instead simulate locally the global reduction above and get a stronger induction hypothesis.

The Definition 2.3 of stack is of course extended to $\mathsf{LC}^\star$ and the Definition 2.4 of head redex is extended to the terms of $\mathsf{LC}^\star$ by saying that $\mathcal{A}\,u\,\sigma$ is the **head redex** of $\mathcal{A}\,u\,\sigma$ whenever u and $\mathcal{A}\,u\,\sigma$ have the same type. The reduction relation $\succ$ for the terms of $\mathsf{LC}^\star$ is then defined as in Definition 2.5. In the following, we define $\mathsf{HN}^\star$ to be the set of normalizing proof terms of $\mathsf{LC}^\star$.

As usual in lambda calculus, a value represents the result of the computation: a function for arrow and universal types, a pair for product types, a boolean for sum types and a witness for existential types and in our case also the abort operator.

Definition 3.1 (Values, Neutrality)**.**

- A proof term is a **value** if it is of the form $\lambda x\,u$ or $\lambda\alpha\,u$ or $\langle u, t\rangle$ or $\iota_i(u)$ or (m, u) or $\mathsf{efq}(u)$ or $\mathcal{A}$.
- A proof term is **neutral** if it is neither a value nor of the form $u \parallel_a v$.

We now prove a property of head normal forms that we will be crucial in the following. It is a generalization of the well known head normal form Theorem for lambda calculus and tells us that if we decompose a proof term into its elementary parallel processes, then each of them is either a value or some variable or constant applied to a list of argument.

Proposition 3.2 (Head Normal Form Property)**.** *Suppose t is in head normal form and*

$$t = t_1 \parallel_{a_1} t_2 \parallel_{a_2} \dots \parallel_{a_n} t_{n+1}$$

and that each t_i is an elementary process. Then for every $1 \leq i \leq n+1$, there is some stack σ such that either $t_i = x\,\sigma$, with $x \neq a_1, \dots, a_n$, or $t_i = \mathcal{A}\,u\,\sigma$, with the type of u different from the type of $\mathcal{A}\,u\,\sigma$, or $t_i = a_j$ or t_i is a value.

Proof. By induction on t. If t is a value, we are done. There are two other cases to consider.

(1) $t = u \parallel_a v$. By Definition 2.4, the parallel processes of u and v are parallel processes of $u \parallel_a v$ as well, so they cannot have head redexes; hence u and v are in head normal form. By induction hypothesis, u and v are of the desired form, thus we just have to check that if

$$u = u_1 \parallel_{a_1} u_2 \parallel_{a_2} \dots \parallel_{a_k} u_{k+1}$$

and for some i, $u_i = x\,\sigma$, with $\sigma \neq \epsilon$, then $x \neq a$ (and symmetricallly for v). Indeed, if for some i, $u_i = a\,\sigma$, then $u = \mathcal{C}[a\,\sigma]$ for some parallel context $\mathcal{C}[\ ]$, and therefore $u \parallel_a v$ would be the leftmost head redex of itself, which is impossible since by assumption it is in head normal form.

(2) t is neutral. Then t can be written, for some stack σ, as $r\,\sigma$ where r is a value or $r = u \parallel_a v$ or $r = x$. In the third case, we are done; in the first and second case, $\sigma = \xi \bullet \rho$, so $r\,\xi$ would be the head redex of t, unless $t = \mathcal{A}\,\xi\,\rho$, with the type of ξ different from the type of $\mathcal{A}\,\xi\,\rho$, which is the thesis. □

3.2. **Definition of Classical Realizability.** Our main goal now is to prove the Normalization Theorem for LC: every proof term of LC reduces in a finite number of head reduction steps to a head normal form. We shall employ a notion of *classical realizability*, a generalization of the Tait-Girard reducibility method [19] that works for classical type systems. The origins of classical realizability can be traced all the way back to Parigot [28] and Krivine [25] classical reducibility, but we present it in a fashion popularized later by Krivine in his work on realizability [26], which is indeed a generalization of classical reducibility. Thanks to the fact that one considers only head reduction, Krivine-style classical realizability is slightly simpler than the notions usually employed to derive strong normalization.

Given a logic, we raise a question: what kind of evidence does a proof provide other than the tiny bit "1" declaring the truth of the proven statement? Realizability is a semantics explaining what is to be taken as *constructive evidence for a statement* and a technique for showing that proofs can provide such an evidence. Formally, realizability is a relation between terms of $\mathsf{LC}^\star$ and formulas, with terms playing the role of constructions and formulas determining what properties a construction should satisfy. In particular, to each formula C is associated a set of stacks $||C||$, which represents a collection of *valid tests*: whenever a term passes all these tests, in the sense that it maps them into terminating programs, it is a realizer. As prescribed by the pragmatist viewpoint, the clauses that defines realizability follow the shape of elimination rules, in order to make sure that no matter how a program is used, it always terminates.

Definition 3.3 (Valid Tests, Classical Realizability)**.** Assume t is a term of $\mathsf{LC}^\star$ and C is a formula of $\mathcal{L}$. We define by mutual induction the relation $t \Vdash C$ ("t realizes C") and a set $||C||$ of stacks of $\mathsf{LC}^\star$ (the "valid tests for C") according to the form of C:

- $t \Vdash C$ if and only if $t : C$ and for all $\sigma \in ||C||$, $t\,\sigma \in \mathsf{HN}^\star$
- $||\mathsf{P}|| = \{\epsilon\}$
- $||A \to B|| = \{u \bullet \sigma \mid u \Vdash A \wedge \sigma \in ||B||\} \cup \{\epsilon\}$
- $||A \wedge B|| = \{\pi_0 \bullet \sigma \mid \sigma \in ||A||\} \cup \{\pi_1 \bullet \sigma \mid \sigma \in ||B||\} \cup \{\epsilon\}$
- $||A \vee B|| = \{[x.u, y.v] \bullet \sigma \mid \forall t.\ (t \Vdash A \implies u[t/x]\,\sigma \in \mathsf{HN}^\star) \wedge (t \Vdash B \implies v[t/y]\,\sigma \in \mathsf{HN}^\star)\} \cup \{\epsilon\}$
- $||\forall \alpha\, A|| = \{m \bullet \sigma \mid m \in \mathcal{L} \wedge \sigma \in ||A[m/\alpha]||\} \cup \{\epsilon\}$
- $||\exists \alpha\, A|| = \{[(\alpha, x).v] \bullet \sigma \mid \forall t.\ t \Vdash A[m/\alpha] \implies v[m/\alpha][t/x]\,\sigma \in \mathsf{HN}^\star\} \cup \{\epsilon\}$

3.3. **Properties of Realizers.** In this section we prove the basic properties of classical realizability. They are all we need to prove the Adequacy Theorem 3.8, which states that typable terms are realizable. The arguments for establishing the properties are in many

cases standard (see Krivine [26]). We shall need extra work for dealing with terms of the form $u \parallel_a v$.

The first task is to prove that realizability is sound for all introduction and elimination rules of LC. We start with the eliminations.

Proposition 3.4 (Properties of Realizability: Eliminations)**.**

(1) *If* $t \Vdash A \to B$ *and* $u \Vdash A$*, then* $tu \Vdash B$*.*
(2) *If* $t \Vdash \forall\alpha\, A$*, then for every term* m *of* $\mathcal{L}$*,* $tm \Vdash A[m/\alpha]$*.*
(3) *If* $t \Vdash A \wedge B$*, then* $t\,\pi_0 \Vdash A$ *and* $t\,\pi_1 \Vdash B$*.*
(4) *If* $t \Vdash A \vee B$ *and for every* $w \Vdash A$*,* $u[w/x] \Vdash C$ *and for every* $w \Vdash B$*,* $v[w/y] \Vdash C$*, then* $t\,[x.u, y.v] \Vdash C$*.*
(5) *If* $t \Vdash \exists\alpha\, A$ *and for every* $m \in \mathcal{L}$ *and for every* $w \Vdash A[m/\alpha]$*,* $u[m/\alpha][w/x] \Vdash C$*, then* $t\,[(\alpha, x).u] \Vdash C$*.*

Proof.

(1) Assume $t \Vdash A \to B$ and $u \Vdash A$. Let $\sigma \in ||B||$; we must show $tu\,\sigma \in \mathsf{HN}^\star$. Indeed, since $t \Vdash A$, by Definition 3.3 $u \bullet \sigma \in ||A \to B||$ and since $t \Vdash A \to B$, we conclude $tu\,\sigma \in \mathsf{HN}^\star$.
(2) Similar to 1.
(3) Assume $t \Vdash A \wedge B$. Let $\sigma \in ||A||$; we must show $t\,\pi_0\,\sigma \in \mathsf{HN}^\star$. Indeed, by Definition 3.3 $\pi_0 \bullet \sigma \in ||A \wedge B||$ and since $t \Vdash A \wedge B$, we conclude $t\,\pi_0\,\sigma \in \mathsf{HN}^\star$. A symmetrical reasoning shows that $t\,\pi_1 \Vdash B$.
(4) Let $\sigma \in ||C||$. We must show that $t\,[x.u, y.v]\,\sigma \in \mathsf{HN}^\star$. By hypothesis, for every $w \Vdash A$, $u[w/x]\,\sigma \in \mathsf{HN}^\star$ and for every $w \Vdash B$, $v[w/y]\,\sigma \in \mathsf{HN}^\star$; by Definition 3.3, $[x.u, y.v]\,\sigma \in ||A \vee B||$. Since $t \Vdash A \vee B$, we conclude $t\,[x.u, y.v]\,\sigma \in \mathsf{HN}^\star$.
(5) Similar to 4. □

Realizability is also sound for introduction rules and the abort operator realizes any implication.

Proposition 3.5 (Properties of Realizability: Introductions)**.**

(1) *If for every* $t \Vdash A$*,* $u[t/x] \Vdash B$*, then* $\lambda x\, u \Vdash A \to B$*.*
(2) *If for every term* m *of* $\mathcal{L}$*,* $u[m/\alpha] \Vdash B[m/\alpha]$*, then* $\lambda\alpha\, u \Vdash \forall\alpha\, B$*.*
(3) *If* $u \Vdash A$ *and* $v \Vdash B$*, then* $\langle u, v\rangle \Vdash A \wedge B$*.*
(4) *If* $t \Vdash A_i$*, with* $i \in \{0, 1\}$*, then* $\iota_i(t) \Vdash A_0 \vee A_1$*.*
(5) *If* $t \Vdash A[m/\alpha]$*, then* $(m, t) \Vdash \exists\alpha\, A$*.*
(6) *If* A *and* B *are any two formulas, then* $\mathcal{A} \Vdash A \to B$*.*

Proof.

(1) Suppose that for every $t \Vdash A$, $u[t/x] \Vdash B$. Let $\sigma \in ||A \to B||$. We have to show $(\lambda x\, u)\,\sigma \in \mathsf{HN}^\star$. If $\sigma = \epsilon$, indeed $\lambda x\, u \in \mathsf{HN}^\star$. Suppose then $\sigma = t \bullet \rho$, with $t \Vdash A$ and $\rho \in ||B||$. Since by hypothesis $u[t/x] \Vdash B$, we have $u[t/x]\,\rho \in \mathsf{HN}^\star$; moreover, $(\lambda x\, u)t\,\rho \succ u[t/x]\,\rho$. Therefore, $(\lambda x\, u)t\,\rho \in \mathsf{HN}^\star$.
(2) Similar to 1.
(3) Suppose $u \Vdash A$ and $v \Vdash B$. Let $\sigma \in ||A \wedge B||$. We have to show $\langle u, v\rangle\,\sigma \in \mathsf{HN}^\star$. If $\sigma = \epsilon$, indeed $\langle u, v\rangle \in \mathsf{HN}^\star$. Suppose then $\sigma = \pi_i \bullet \rho$, with $i \in \{0, 1\}$ and $\rho \in ||A||$, when $i = 0$, and $\rho \in ||B||$, when $i = 1$. We have two cases.
   (a) $i = 0$. Since by hypothesis $u \Vdash A$, we have $u\,\rho \in \mathsf{HN}^\star$; moreover, $\langle u, v\rangle\,\pi_i\,\rho \succ u\,\rho$. Therefore, $\langle u, v\rangle\,\pi_i\,\rho \in \mathsf{HN}^\star$.

(b) $i = 1$. Since by hypothesis $u \Vdash B$, we have $v\,\rho \in \mathsf{HN}^\star$; moreover, $\langle u, v\rangle\,\pi_i\,\rho \succ v\,\rho$. Therefore, $\langle u, v\rangle\,\pi_i\,\rho \in \mathsf{HN}^\star$.

(4) Suppose $t \Vdash A_i$. Let $\sigma \in ||A_0 \vee A_1||$. We have to show $\iota_i(t)\,\sigma \in \mathsf{HN}^\star$. If $\sigma = \epsilon$, indeed $\iota_i(t) \in \mathsf{HN}^\star$. Suppose then $\sigma = [x_0.u_0, x_1.u_1]_\bullet\,\rho$ and for all w, if $w \Vdash A_0$, then $u_0[w/x_0]\,\rho \in \mathsf{HN}^\star$ and if $w \Vdash A_1$, then $u_1[w/x_1]\,\rho \in \mathsf{HN}^\star$. We have to show $\iota_i(t)\,[x_0.u_0, x_1.u_1]\,\rho \in \mathsf{HN}^\star$. By hypothesis $u_i[t/x_i]\,\rho \in \mathsf{HN}^\star$; moreover,

$$\iota_i(t)\,[x_0.u_0, x_1.u_1]\,\rho \succ u_i[t/x_i]\,\rho$$

Therefore, $\iota_i(t)\,[x_0.u_0, x_1.u_1] \in \mathsf{HN}^\star$.

(5) Similar to 4.

(6) Let $\sigma \in ||A \to B||$. We have to show that $\mathcal{A}\,\sigma \in \mathsf{HN}^\star$. If $\sigma = \epsilon$, indeed $\mathcal{A} \in \mathsf{HN}^\star$. Suppose then $\sigma = u_\bullet\,\rho$, with $u \Vdash A$ and $\rho \in ||B||$. We have to show that $\mathcal{A}\,u\,\rho \in \mathsf{HN}^\star$. Since $u \Vdash A$ and $\epsilon \in ||A||$, we have $u = u\,\epsilon \in \mathsf{HN}^\star$. Moreover, if $\mathcal{A}\,u\,\rho$ is not a redex, we are done, and if $\mathcal{A}\,u\,\rho \succ u$, the thesis follows. □

It is now that the abort operator really enters the scene. Thanks to it, any reduction $u \parallel_a v \succ u' \parallel_a v$ can be simulated in a purely local way. This is possible because any such reduction affects only what is inside u and leaves v untouched. Then, in order to replicate the reduction is enough to substitute to a a term $\mathscr{A}$ that throws away any stack of terms it is applied to, like a does, and then restores v, with some substitution depending on the context. Of course, symmetrical considerations hold true for any reduction $u \parallel_a v \succ u \parallel_a v'$.

Proposition 3.6 (Local Simulation)**.** *Define*

$$\mathscr{A} := \lambda x\,\mathcal{A}\,v[\lambda y\,x/a]$$
$$\mathscr{B} := \lambda z\,\mathcal{A}\,u[\lambda y\,z/a]$$

with x, y, z and $\mathcal{A}$ occurring with the right type. Then

$$(u \parallel_a v \succ u' \parallel_a v) \implies u[\mathscr{A}/a] \succ^+ u'[\mathscr{A}/a]$$
$$(u \parallel_a v \succ u \parallel_a v') \implies v[\mathscr{B}/a] \succ^+ v'[\mathscr{B}/a]$$

Proof. We prove the first statement, the other being perfectly symmetric. The only trouble is to formalize precisely the argument, which is otherwise intuitively obvious. To this end, we first need some simple, but tedious to prove, claims.

- *Claim 1.* Every parallel process of $u[\mathscr{A}/a]$ is of the form $t[\mathscr{A}/a]$, where t is a parallel process of u.
- *Claim 2.* For every parallel process $t[\mathscr{A}/a]$ of $u[\mathscr{A}/a]$, if the starting symbol of the head redex of t is in the n-th, from left to right, elementary process of u, then the starting symbol of the head redex of $t[\mathscr{A}/a]$ is in the n-th elementary process, from left to right, of $u[\mathscr{A}/a]$.

Proof of Claim 1. By induction on u. Let t' be a parallel process of $u' = u[\mathscr{A}/a]$. If $t' = u'$, since u is a parallel process of itself, we are done. Suppose now $u' = u'_1 \parallel_{b'} u'_2$. Then, assuming $u = u_1 \parallel_b u_2$, we have

$$u'_1 \parallel_{b'} u'_2 = (u_1 \parallel_b u_2)[\mathscr{A}/a] = (u_1[\mathscr{A}/a] \parallel_b u_2[\mathscr{A}/a])$$

Therefore, $b' = b$, $u'_1 = u_1[\mathscr{A}/a]$ and $u'_2 = u_2[\mathscr{A}/a]$. Now, t' is a parallel process of either u'_1 or u'_2; by induction hypothesis, $t' = t[\mathscr{A}/a]$, where t is a parallel process of u_1, in the first case, and of u_2 in the second.

Proof of Claim 2. By induction on u. Let $t' = t[\mathscr{A}/a]$ be a parallel process of $u' = u[\mathscr{A}/a]$, where t is a parallel process of u. We have two cases.

(1) $t' = u[\mathscr{A}/a]$. If $u = h\,\sigma$ and h is its head redex, then u is the first and unique elementary process of u; moreover, $t' = (h[\mathscr{A}/a])\,(\sigma[\mathscr{A}/a])$, thus $h[\mathscr{A}/a]$ is its head redex and indeed t' is the first and unique elementary process of t. If $u = u_1 \parallel_b u_2$ and is a redex, then by Definition 2.5, the starting symbol of u is the occurrence of b such that $b\,w\,\sigma$ is the n-th elementary process of u and for every $m < n$, the m-th elementary process of u does not start with b. Since $t' = u_1[\mathscr{A}/a] \parallel_b u_2[\mathscr{A}/a]$ and $a \neq b$, the starting symbol of t' is the occurrence of b such that $b\,(w[\mathscr{A}/a])\,(\sigma[\mathscr{A}/a])$ is the n-th elementary process of t'.

(2) $t' \neq u[\mathscr{A}/a]$. Since $t' = t[\mathscr{A}/a]$ and t is a parallel process of u with $t \neq u$, there is a parallel context $\mathcal{C}[\ ]$ such that $u = \mathcal{C}[t]$. By induction hypothesis, assuming that the starting symbol of the head redex of t is in the n-th elementary process of t, then the starting symbol of the head redex of t' is in the n-th elementary process of t'. Since $u' = (\mathcal{C}[\mathscr{A}/a])[t']$, if m is the number of elementary processes on the left of t in $\mathcal{C}[t]$, then the starting symbol of the head redex of t is in the $(m+n)$-th elementary process of u and the starting symbol of the head redex of t' is in the $(m+n)$-th elementary process of u', which is the thesis.

Let now us return to the main line of the proof. Suppose

$$u \parallel_a v \succ u' \parallel_a v$$

Then for some parallel context $\mathcal{C}$, we have $u = \mathcal{C}[q]$ and $u' = \mathcal{C}[q']$, where q' is either obtained from q by contracting the head redex r of q or $q' = v[\lambda y\, t/a]$ and $q = a\,t\,\sigma$; in the first case, it is r that is the leftmost among the head redexes of the parallel processes of $u \parallel_a v$, whereas in the second case, it is $u \parallel_a v$. With this notation, we have

$$\mathcal{C}[q] \parallel_a v \succ \mathcal{C}[q'] \parallel_a v$$

We must show

$$u[\mathscr{A}/a] \succ^+ u'[\mathscr{A}/a]$$

There are several cases.

- $q = (\lambda x\, s)t\,\sigma$ and $r = (\lambda x\, s)t$. Let $r' = s[t/x]$. We first need to show that $r[\mathscr{A}/a]$ is the leftmost among the head redexes of the parallel process of $u[\mathscr{A}/a]$ as well. Assume that the starting symbol of r is in the n-th elementary processes of u. By Claim 2, the starting symbol of $r[\mathscr{A}/a]$ is in the n-th elementary process of $u[\mathscr{A}/a]$. Suppose by the way of contradiction, that there is a parallel process p' of $u[\mathscr{A}/a]$ whose head redex has a starting symbol more on the left, that is, in the m-th elementary process of $u[\mathscr{A}/a]$, with $m < n$. By Claim 1, $p' = p[\mathscr{A}/a]$, where p is a parallel process of u. By Claim 2, the starting symbol of the head redex of p is in the m-th elementary process of u, which contradicts the assumption on q and r. Now, letting $s' = s[\mathscr{A}/a]$, $t' = t[\mathscr{A}/a]$, $\mathcal{C}' = \mathcal{C}[\mathscr{A}/a]$, $\sigma' = \sigma[\mathscr{A}/a]$, we get
$$u[\mathscr{A}/a] = \mathcal{C}'[(\lambda x\, s')t'\,\sigma'] \succ \mathcal{C}'[s'[t'/x]\,\sigma'] = \mathcal{C}[s[t/x]\,\sigma][\mathscr{A}/a] = u'[\mathscr{A}/a]$$
- $q = r\,\sigma$ and $r = \langle s_0, s_1\rangle\,\pi_i$ or $r = \iota_i(s)[x_0.t_0, x_1.t_1]$ or $r = (m,s)[(\alpha,x).t]$ or $r = (w_1 \parallel_b w_2)\,\rho$ and let, respectively, $r' = s_i$ or $r' = t_i[s/x_i]$ or $r' = t[m/\alpha][s/x]$ or $r' = w_1\,\rho \parallel_b w_2\,\rho$. By exactly the same considerations of the previous case, we get
$$u[\mathscr{A}/a] = \mathcal{C}[\mathscr{A}/a][r\,\sigma[\mathscr{A}/a]] \succ \mathcal{C}[\mathscr{A}/a][r'\,\sigma[\mathscr{A}/a]] = \mathcal{C}[r'\,\sigma][\mathscr{A}/a] = u'[\mathscr{A}/a]$$

- $q = r = \mathcal{A}_k\, w\, \sigma$ or $q = r = \mathcal{C}_1[b\, w\, \rho] \parallel_b s$ for some variable $b \neq a$ (the other case is symmetric); let respectively $r' = w$ or $r' = \mathcal{C}_1[s[\lambda y\, w/b]] \parallel_b s$. By exactly the same considerations of the previous case, we get
$$u[\mathscr{A}/a] = \mathcal{C}[\mathscr{A}/a][r[\mathscr{A}/a]] \succ \mathcal{C}[\mathscr{A}/a][r'[\mathscr{A}/a]] = \mathcal{C}[r'][\mathscr{A}/a] = u'[\mathscr{A}/a]$$
- $q' = v[\lambda y\, t/a]$ and $q = a\, t\, \sigma$. Let $t' = t[\mathscr{A}/a]$, $\sigma' = \sigma[\mathscr{A}/a]$ and $\mathcal{C}' = \mathcal{C}[\mathscr{A}/a]$. We first need to show that the head redex of $q[\mathscr{A}/a] = \mathscr{A}\, t'\, \sigma'$ is the leftmost among the head redexes of the parallel processes of $u[\mathscr{A}/a]$. Assume that $a\, t\, \sigma$ is the n-th elementary process of u, so that $\mathscr{A}\, t'\, \sigma'$ is the n-th elementary process of $u[\mathscr{A}/a]$ as well. Then, no parallel process of u has an head redex whose starting symbol is in the m-th elementary process of u, with $m < n$. By Claims 1 and 2, no parallel process $p[\mathscr{A}/a]$ of $u[\mathscr{A}/a]$, where p is a parallel process of u, has an head redex whose starting symbol is in the m-th elementary process of u, with $m < n$, otherwise the starting symbol of the head redex of p would be in the m-th elementary process of u as well (Claim 2 applies, since p cannot be of the form $a\, w\, \rho$, given that a is the starting symbol of the redex $u \parallel_a v$). Finally, we conclude
$$u[\mathscr{A}/a] = \mathcal{C}'[\mathscr{A}\, t'\, \sigma'] = \mathcal{C}'[(\lambda x\, \mathcal{A}\, v[\lambda y\, x/a])\, t'\, \sigma'] \succ$$
$$\mathcal{C}'[\mathcal{A}\, v[\lambda y\, t'/a]\, \sigma'] \succ \mathcal{C}'[v[\lambda y\, t'/a]] = \mathcal{C}[v[\lambda y\, t/a]][\mathscr{A}/a] = u'[\mathscr{A}/a]$$ □

We are now able to tackle the most difficult case of the Adequacy Theorem 3.8 for realizability: proving that realizability is also sound for the Dummett rule. The idea is that Proposition 3.6 allows us to use in a very strong manner an inductive hypothesis that will naturally be granted when proving the Adequacy Theorem. This hypothesis is knowing that for every $t \Vdash A \to B$, $u[t/a] \in \mathsf{HN}^\star$; since one can prove that the term $\mathscr{A}$ realizes $A \to B$, one can conclude with simple reasoning that the head reduction reduces u in $u \parallel_a v$ only a finite number of times and a symmetric reasoning holds for v. Hadn't we the abort operator and thus the possibility of local simulation, the hypothesis that for every $t \Vdash A \to B$, $u[t/a] \in \mathsf{HN}^\star$, would not be enough to conclude a great deal. Details follow.

Proposition 3.7 (Preservation of Realizability by Parallel Composition)**.**

(1) *If for every $t \Vdash A \to B$, $u[t/a] \in \mathsf{HN}^\star$ and for every $t \Vdash B \to A$, $v[t/a] \in \mathsf{HN}^\star$, then $u \parallel_a v \in \mathsf{HN}^\star$.*
(2) *If for every $t \Vdash A \to B$, $u[t/a] \Vdash C$ and for every $t \Vdash B \to A$, $v[t/a] \Vdash C$, then $u \parallel_a v \Vdash C$.*

Proof.

(1) Define
$$\mathscr{A} := \lambda x\, \mathcal{A}(v[\lambda y\, x/a])$$
We start by showing that $\mathscr{A} \Vdash A \to B$, which establishes by means of the hypothesis that $u[\mathscr{A}/a] \in \mathsf{HN}^\star$. Let $\rho \in ||A \to B||$; the case $\rho = \epsilon$ is trivial, so we assume $\rho = t \bullet \sigma$, with $t \Vdash A$ and $\sigma \in ||B||$. We must show $\mathscr{A}\, t\, \sigma \in \mathsf{HN}^\star$. We have
$$\mathscr{A}\, t\, \sigma = (\lambda x\, \mathcal{A}(v[\lambda y\, x/a]))t\, \sigma \succ \mathcal{A}(v[\lambda y\, t/a])\, \sigma \succ v[\lambda y\, t/a]$$
(assuming the last reduction is possible: if not, the thesis is trivial). In order to obtain $v[\lambda y\, t/a] \in \mathsf{HN}^\star$, which is what we wanted, it is enough to show that $\lambda y\, t \Vdash B \to A$. Let $\rho' \in ||B \to A||$; again, the case $\rho' = \epsilon$ is trivial, so we assume $\rho' = t' \bullet \sigma'$, with $t' \Vdash B$ and $\sigma' \in ||A||$. We must show $(\lambda y\, t)\, t'\, \sigma' \in \mathsf{HN}^\star$. Indeed, since $t \Vdash A$,
$$(\lambda y\, t)\, t'\, \sigma' \succ t\, \sigma' \in \mathsf{HN}^\star$$

We now prove that $u \parallel_a v \in \mathsf{HN}^\star$ by induction on the length of the reduction of $u[\mathscr{A}/a]$ in head normal form. We have two cases.

(a) Assume
$$u \parallel_a v \succ u \parallel_a v'$$
so that in particular u is in head normal form. Define
$$\mathscr{B} := \lambda x\, \mathcal{A}(u[\lambda y\, x/a])$$
Since we are going again to prove the thesis by induction on the length of the reduction of $v[\mathscr{B}/a]$ in head normal form, we first need to show that $\mathscr{B} \Vdash B \to A$, which allows us to conclude that indeed $v[\mathscr{B}/a] \in \mathsf{HN}^\star$. Let $\rho \in ||B \to A||$; the case $\rho = \epsilon$ is trivial, so we assume $\rho = t \bullet \sigma$, with $t \Vdash B$ and $\sigma \in ||A||$. We have to show $\mathscr{B}\, t\, \sigma \in \mathsf{HN}^\star$. We have
$$\mathscr{B}\, t\, \sigma = (\lambda x\, \mathcal{A}(u[\lambda y\, x/a]))t\, \sigma \succ \mathcal{A}(u[\lambda y\, t/a])\, \sigma \succ u[\lambda y\, t/a]$$
Now, u is in head normal form and thus by Proposition 3.2,
$$u = u_0 \parallel_{a_1} u_1 \parallel_{a_2} \dots \parallel_{a_n} u_n$$
and for each $0 \le i \le n$, either $u_i = x\, \sigma$, with $x \neq a_1, \dots, a_n, a$, or $u_i = \mathcal{A}\, w\, \sigma$, with the type of w different from the type of $\mathcal{A}\, w\, \sigma$, or $u_i = a_j$ or $u_i = a$ or u_i is a value. Therefore, $u[\lambda y\, t/a]$ is in head normal form, because the substitution does not create head redexes in any parallel process of u.
We now prove the main thesis. By Proposition 3.6, $v[\mathscr{B}/a] \succ^+ v'[\mathscr{B}/a]$, so by induction hypothesis we conclude $u \parallel_a v' \in \mathsf{HN}^\star$ and thus $u \parallel_a v \in \mathsf{HN}^\star$.

(b) Assume
$$u \parallel_a v \succ u' \parallel_a v$$
By Proposition 3.6, $u[\mathscr{A}/a] \succ^+ u'[\mathscr{A}/a]$, so by induction hypothesis we conclude $u' \parallel_a v \in \mathsf{HN}^\star$ and thus $u \parallel_a v \in \mathsf{HN}^\star$.

(2) Let $\sigma \in ||C||$. We must show that $(u \parallel_a v)\, \sigma \in \mathsf{HN}^\star$. By hypothesis, for every $t \Vdash A \to B$, $u[t/a]\, \sigma \in \mathsf{HN}^\star$ and for every $t \Vdash B \to A$, $v[t/a]\, \sigma \in \mathsf{HN}^\star$. By point 1., $u\, \sigma \parallel_a v\, \sigma \in \mathsf{HN}^\star$. Since
$$(u \parallel_a v)\, \sigma \succ^* u\, \sigma \parallel_a v\, \sigma$$
we are done. □

3.4. **The Adequacy Theorem.** We finally prove that realizability is sound for LC: if we replace all free proof term variables of any proof term with realizers, then we get a realizer.

Theorem 3.8 (Adequacy Theorem)**.** *Suppose that $w : A$ in the system LC, with w having free variables among $x_1^{A_1}, \dots, x_n^{A_n}$. For all terms $r_1, \dots, r_k$ of $\mathcal{L}$, if there are terms $t_1, \dots, t_n$ such that*
$$\text{for } i = 1, \dots, n,\ t_i \Vdash \overline{A}_i := A_i[r_1/\alpha_1 \cdots r_k/\alpha_k]$$
then
$$w[r_1/\alpha_1 \cdots r_k/\alpha_k][t_1/x_1^{\overline{A}_1} \cdots t_n/x_n^{\overline{A}_n}] \Vdash A[r_1/\alpha_1 \cdots r_k/\alpha_k]$$

Proof. For any term v and formula B, we define

$$\overline{v} := v[r_1/\alpha_1 \cdots r_k/\alpha_k][t_1/x_1^{\overline{A}_1} \cdots t_n/x_n^{\overline{A}_n}]$$

and

$$\overline{B} := B[r_1/\alpha_1 \cdots r_k/\alpha_k]$$

We proceed by induction on w. Consider the last rule $\mathscr{R}$ in the derivation of $w : A$:

(1) If $\mathscr{R} = x_i^{A_i} : A_i$, for some i, then $w = x_i^{A_i}$ and $A = A_i$. So $\overline{w} = t_i \Vdash \overline{A}_i = \overline{A}$.
(2) If $\mathscr{R}$ is the $\to E$ rule, then $w = t\,u$,
$$t : B \to A \qquad u : B$$
So by Proposition 3.4, $\overline{w} = \overline{t}\,\overline{u} \Vdash \overline{A}$, for $\overline{t} \Vdash \overline{B} \to \overline{A}$ and $\overline{u} \Vdash \overline{B}$ by induction hypothesis.
(3) If $\mathscr{R}$ is the $\to I$ rule, then $w = \lambda x^B\, u$, $A = B \to C$ and $u : C$. So, $\overline{w} = \lambda x^{\overline{B}}\, \overline{u}$, because by renaming of bound variables we can assume $x^{\overline{B}} \neq x_1^{\overline{A}_1}, \ldots, x_k^{\overline{A}_k}$. For every $t \Vdash \overline{B}$, by induction hypothesis on u, $\overline{u}[t/x^{\overline{B}}] \Vdash \overline{C}$. Therefore, by Proposition 3.5, $\lambda x^{\overline{B}}\, \overline{u} \Vdash \overline{B} \to \overline{C} = \overline{A}$.
(4) If $\mathscr{R}$ is a $\vee I$ rule, say left (the other case is symmetric), then $w = \iota_0(u)$, $A = B \vee C$ and $u : B$. So, $\overline{w} = \iota_0(\overline{u})$ and by induction hypothesis $\overline{u} \Vdash \overline{B}$. Hence, by Proposition 3.5 we conclude $\iota_0(\overline{u}) \Vdash \overline{B} \vee \overline{C} = \overline{A}$.
(5) If $\mathscr{R}$ is a $\vee E$ rule, then
$$w = u[x^B.w_1, y^C.w_2]$$
and
$$u : B \vee C \qquad w_1 : D \qquad w_2 : D$$
with $A = D$. By induction hypothesis, we have $\overline{u} \Vdash \overline{B} \vee \overline{C}$; moreover, for every $t \Vdash \overline{B}$, we have $\overline{w}_1[t/x^{\overline{B}}] \Vdash \overline{D}$ and for every $t \Vdash \overline{C}$, we have $\overline{w}_2[t/y^{\overline{C}}] \Vdash \overline{D}$. By Proposition 3.4, we obtain $\overline{w} = \overline{u}\,[x^{\overline{B}}.\overline{w}_1, y^{\overline{C}}.\overline{w}_2] \Vdash \overline{D}$.
(6) The cases $\mathscr{R} = \wedge E$ and $\mathscr{R} = \wedge I$ are straightforward.
(7) The cases $\mathscr{R} = \exists I$ and $\mathscr{R} = \exists E$ are similar respectively to $\vee I$ and $\vee E$.
(8) If $\mathscr{R}$ is the $\forall E$ rule, then $w = u\,m$, $A = B[m/\alpha]$ and $u : \forall\alpha\, B$. So, $\overline{w} = \overline{u}\,\overline{m}$. By inductive hypothesis $\overline{u} \Vdash \forall\alpha\, \overline{B}$ and so $\overline{u}\,\overline{m} \Vdash \overline{B}[\overline{m}/\alpha]$ by Proposition 3.4.
(9) If $\mathscr{R}$ is the $\forall I$ rule, then $w = \lambda\alpha\, u$, $A = \forall\alpha\, B$ and $u : B$ (with α not occurring free in the types $A_1, \ldots, A_n$ of the free variables of u). So, $\overline{w} = \lambda\alpha\, \overline{u}$, since we may assume $\alpha \neq \alpha_1, \ldots, \alpha_k$. Let m be a term of $\mathcal{L}$; by Proposition 3.5, it is enough to prove that $\overline{u}[m/\alpha] \Vdash \overline{B}[m/\alpha]$, which amounts to showing that the induction hypothesis can be applied to u. For this purpose, we observe that, since $\alpha \neq \alpha_1, \ldots, \alpha_k$, for $i = 1, \ldots, n$ we have
$$t_i \Vdash \overline{A}_i = \overline{A}_i[m/\alpha]$$
(10) If $\mathscr{R}$ is the D rule, then $w = u \parallel_a v$, $A = D$ and
$$\dfrac{\begin{array}{cc}[a^{B\to C} : B \to C] & [a^{C\to B} : C \to B] \\ \vdots & \vdots \\ u : D & v : D\end{array}}{u \parallel_a v : D}$$
By induction hypothesis, for every $t \Vdash \overline{B} \to \overline{C}$, we have $\overline{u}[t/a] \Vdash \overline{D}$ and for every $t \Vdash \overline{C} \to \overline{B}$, $\overline{v}[t/a] \Vdash \overline{D}$. By Proposition 3.7, we conclude $\overline{w} = \overline{u} \parallel_a \overline{v} \Vdash \overline{D}$.

(11) If $\mathscr{R}$ is the ex falso quodlibet rule, then $w = \mathsf{efq}_P(u)$, $A = P$ and $u : \bot$. Now, $||P|| = \{\epsilon\}$ and $\overline{w}\,\epsilon = \overline{w} = \mathsf{efq}_P(\overline{u}) \in \mathsf{HN}^\star$. We conclude $\overline{w} \Vdash A$. □

3.5. **Normalization for LC.** As corollary of the Adequacy Theorem 3.8, one obtains normalization for LC.

Corollary 3.9 (Normalization for LC)**.** *Suppose that $t : A$ is a proof term of* LC*. Then $t \in \mathsf{HN}^\star$.*

Proof. Assume $x_1 : A_1, \dots, x_n : A_n$ are the free variables of t. We observe that $x_i \Vdash A_i$, for $i = 1, \dots, n$ because, given any $\sigma \in ||A_i||$, $x\,\sigma \in \mathsf{HN}^\star$. Therefore, from Theorem 3.8, we derive that $t \Vdash A$ and since $\epsilon \in ||A||$, we conclude $t = t\,\epsilon \in \mathsf{HN}^\star$. □

4. Normal Form Property and Herbrand's Disjunction Extraction

In this section, we finally show that our Curry-Howard correspondence for LC is meaningful from the computational perspective. We already know that every execution of every program we extract always terminate; now we prove that in the case of *any* existentially quantified formula $\exists\alpha\, A$, every closed program of that type produces a complete finite sequence $m_1, m_2, \dots, m_k$ of possible witnesses for $\exists\alpha\, A$. This means that whatever first-order model we consider, there will be an i such that $A[m_i/\alpha]$ is true in it. In other terms, we have provided a proof that LC is Herbrand constructive and a Curry-Howard computational interpretation of this very strong Herbrand-like theorem.

Such statements in first-order logic are typically drawn as consequences of the Subformula Property, which is in turn a corollary of full cut-elimination when sequent calculus is available. But as in [7], a more primitive argument suffices here. This is indeed providential, since not only without permutation rules for $\vee$ and $\exists$ we can have no Subformula Property, but surprisingly even those reductions would not suffice. The topic of what reductions are needed is very non-trivial and left as subject of future research. However, in a sense, Herbrand constructiveness is already a *weak* Subformula Property and holds for the most interesting case of the existential quantifier, when there is actually some information to gain. For lambda calculus, instead, to enjoy the Subformula Property is a mere curiosity without much computational sense. In fact, if we think that in intuitionistic Logic or fragments of classical Arithmetic [4] general permutation rules are not needed to compute witnesses, it should not entirely come as a surprise that this is still the case in our framework.

If we omit parentheses, we know that every proof term in head normal form can be written as $v_0 \parallel_{a_1} v_1 \dots \parallel_{a_n} v_n$, where each v_i is not of the form $u \parallel_a v$; if for every i, v_i is of the form (m_i, u_i), then we call the whole term an *Herbrand normal form*, because it is essentially a list of the witnesses appearing in an Herbrand disjunction. Formally:

Definition 4.1 (Herbrand Normal Forms)**.** We define by induction a set of proof terms, called **Herbrand normal forms**, as follows:

- Every proof-term (m, u) is an Herbrand normal form;
- if u and v are Herbrand normal forms, $u \parallel_a v$ is an Herbrand normal form.

An Herbrand normal form represents, in a straightforward way, a proof of an Herbrand disjunction.

Proposition 4.2 (Herbrand Normal Forms and Herbrand Disjunctions)**.** *Suppose that* $\Gamma \vdash u : \exists\alpha\, A$ *in* LC *and* u *is an Herbrand normal form*

$$(m_0, v_0) \parallel_{a_1} (m_1, v_1) \parallel_{a_2} \dots \parallel_{a_k} (m_k, v_k)$$

Then for some u^+

$$\Gamma \vdash u^+ : A[m_1/\alpha] \vee \dots \vee A[m_k/\alpha]$$

Proof. We proceed by induction on k.
If $k = 0$, then $u = (m_0, v_0)$ and thus $\Gamma \vdash v_0 : A[m_0/\alpha]$, which is the thesis.
If $k > 0$, then $u = w_1 \parallel_{a_i} w_2$, for some $1 \leq i \leq n$ and

$$w_1 = (m_0, v_0) \parallel_{a_1} (m_1, v_1) \parallel_{a_2} \dots \parallel_{a_{i-1}} (m_{i-1}, v_{i-1})$$
$$w_2 = (m_i, v_i) \parallel_{a_{i+1}} (m_{i+1}, v_{i+1}) \parallel_{a_{i+2}} \dots \parallel_{a_k} (m_k, v_k)$$

$$\frac{\begin{array}{cc}[a_i^{B\to C} : B \to C] & [a_i^{C\to B} : C \to B] \\ \vdots & \vdots \\ w_1 : \exists\alpha\, A & w_2 : \exists\alpha\, A\end{array}}{w_1 \parallel_a w_2 : \exists\alpha\, A}$$

By induction hypothesis,

$$\Gamma, a_i : B \to C \vdash w_1^+ : A[m_1/\alpha] \vee \dots \vee A[m_{i-1}/\alpha]$$
$$\Gamma, a_i : C \to B \vdash w_2^+ : A[m_i/\alpha] \vee \dots \vee A[m_k/\alpha]$$

Hence, by repeated application of the $\vee I$ inference, we get, for some s_1, s_2,

$$\Gamma, a_i : B \to C \vdash s_1 : A[m_1/\alpha] \vee \dots \vee A[m_k/\alpha]$$
$$\Gamma, a_i : C \to B \vdash s_2 : A[m_1/\alpha] \vee \dots \vee A[m_k/\alpha]$$

and thus

$$\Gamma \vdash s_1 \parallel_{a_i} s_2 : A[m_1/\alpha] \vee \dots \vee A[m_k/\alpha]$$

By setting $u^+ := s_1 \parallel_{a_i} s_2$, we obtain the thesis. □

Our last task is to prove that every closed realizer of any existentially quantified statement $\exists\alpha\, A$ include an exhaustive sequence $m_1, m_2, \dots, m_k$ of possible witnesses.

Theorem 4.3 (Herbrand Disjunction and Realizability)**.** *Let* $\exists\alpha\, A$ *be any formula. Suppose* $t \Vdash \exists\alpha\, A$*,* t *contains neither free proof term variables nor* $\mathcal{A}$*, and* $t \succ^* u \in \mathsf{HNF}$*. Then* u *is an Herbrand normal form*

$$u = (m_0, v_0) \parallel_{a_1} (m_1, v_1) \parallel_{a_2} \dots \parallel_{a_k} (m_k, v_k)$$

and

$$\mathsf{LC} \vdash A[m_1/\alpha] \vee \dots \vee A[m_k/\alpha]$$

Proof. By Proposition 3.2

$$u = u_0 \parallel_{a_1} u_1 \parallel_{a_2} \dots \parallel_{a_k} u_k$$

where for each $0 \leq i \leq k$, either $u_i = x\,\sigma$, with $x \neq a_1, \dots, a_n$ or $u_i = a_j$, with $1 \leq j \leq k$, or u_i is a value. Since u_i does not contain free proof term variables other than $a_1, \dots, a_k$, it cannot be of the form $u_i = x\,\sigma$. Moreover, $u_i : \exists\alpha\, A$, hence u_i cannot be equal to some a_j, because a_j must have type $B \to C$. Therefore u_i is a value, according to Definition 3.1,

and the only possible shape compatible with its type $\exists\alpha\, A$ is (m_i, u_i). We have thus shown that u is an Herbrand normal form

$$(m_0, v_0) \parallel_{a_1} (m_1, v_1) \parallel_{a_2} \dots \parallel_{a_k} (m_k, v_k)$$

By Proposition 4.2, for some u^+

$$\mathsf{LC} \vdash u^+ : A[m_1/\alpha] \vee \dots \vee A[m_k/\alpha]$$

which is the thesis. □

As corollary, we obtain that Dummett's logic LC is Herbrand constructive.

Corollary 4.4 (Herbrand Disjunction Extraction)**.** *Let $\exists\alpha\, A$ be any formula. Suppose*

$$\mathsf{LC} \vdash t : \exists\alpha\, A$$

Then there is a proof term u such that $t \succ^* u \in \mathsf{HNF}$, $\mathsf{LC} \vdash u : \exists\alpha\, A$ and u is an Herbrand normal form

$$u = (m_0, v_0) \parallel_{a_1} (m_1, v_1) \parallel_{a_2} \dots \parallel_{a_k} (m_k, v_k)$$

Moreover,

$$\mathsf{LC} \vdash A[m_1/\alpha] \vee \dots \vee A[m_k/\alpha]$$

Proof. By the Subject Reduction Theorem 2.7, $\mathsf{LC} \vdash u : \exists\alpha\, A$. By the Adequacy Theorem 3.8, $t \Vdash \exists\alpha\, A$ and the thesis follows from Theorem 4.3. □

We suggest to interpret an Herbrand normal form

$$(m_0, v_0) \parallel_{a_1} (m_1, v_1) \parallel_{a_2} \dots \parallel_{a_k} (m_k, v_k)$$

in the following way. Each (m_i, u_i) represents the result of an intuitionistic computation of a witness in a possible universe. These witnesses have been obtained by communication coming from other intuitionistic computations in other parallel universes. It is that process of interaction and dialogue between different possible computations that generates the Herbrand normal forms.

4.1. **Parallel Reductions.** Head reduction, of course, is sequential computation. Yet, the operator $\parallel_a$ has such a strong parallel flavour that parallel reduction strategies inevitably arise as consequence of Normalization for head reduction. To see this, let us consider a proof term $u \parallel_a v$ of LC. By the Normalization Theorem 3.9, the head reduction of $u \parallel_a v$ reduces subterms inside the left part of the term until it is possible, afterwards it continues to reduce the right part and finally it stops. If we consider only the first half of the reduction, we get

$$u \parallel_a v \succ^* u' \parallel_a v \tag{4.1}$$

for some u' in head normal form and *not* of the shape $\mathcal{C}[a\, t\, \sigma]$, otherwise a further reduction inside u' would be possible. Thanks to the *perfect logical symmetry* of the term $u \parallel_a v$, also $v \parallel_a u$ is a term of the same type. Again, we can reduce

$$v \parallel_a u \succ^* v' \parallel_a u \tag{4.2}$$

for some v' in head normal form and *not* of the shape $\mathcal{C}[a\, t\, \sigma]$. The point is that the head reductions (4.1) and (2) *can be made in parallel* and what we get,

$$u' \parallel_a v'$$

not only is a term of the same type of $u \parallel_a v$ and with no more free variables, it also is a head normal form!

5. Second-Order Intuitionistic Logic with Dummett's Axiom

At the time of this writing, there is no known cut-free sequent calculus for second-order intuitionistic logic with Dummett's Axiom, which we call LC_2. Even if there were one, the situation would be similar to what happens in the hypersequent calculus for second-order Gödel-Dummett logic [27]: there is no known cut-elimination procedure, only a semantical proof that valid statements can be proved without cuts. Why? This state of thing reminds the status of Takeuti's conjecture [33], a problem which resisted the effort of the best researchers for many years in the 1950-60's, and was solved constructively in 1971 by Girard (see [19]). It asked whether the now standard second-order sequent calculus was cut-free. A cut-elimination procedure for intuitionistic second-order sequent calculus was finally obtained only through translation to natural deduction, where the powerful Tait-Girard reducibility settles the matter. This shortcoming of sequent calculus is even worse in the case of hypersequent calculus, which is more complicated and no cut-elimination procedure is known at second-order.

In this section, we consider second-order natural deduction for LC_2 and prove the Normalization of head reduction. Unlike in hypersequent calculus, where second-order cut-elimination requires climbing a steep and cold combinatorial mountain, extending classical realizability to the second-oder case is a like a quiet stroll in a peaceful and sunny countryside road. Indeed, classical realizability was introduced directly in the second-order case by Parigot [28] and Krivine [25], without even bothering with the first-order case. We follow once again Krivine's successive formulation [26].

The language $\mathcal{L}^2$ of LC_2 extends $\mathcal{L}$ in the standard way, adding second order predicate variables, representing sets of individuals.

Definition 5.1 (Language of LC_2)**.** The language $\mathcal{L}^2$ of LC_2 is defined as follows.

(1) The **terms** of $\mathcal{L}^2$ are inductively defined as either variables $\alpha, \beta, \ldots$ or constants c or expressions of the form $\mathsf{f}(m_1, \ldots, m_n)$, with f a function constant of arity n and $m_1, \ldots, m_n \in \mathcal{L}^2$.
(2) There is a set of **predicate constant symbols** and of **predicate variables**. The **atomic formulas** of $\mathcal{L}^2$ are all the expressions of the form $\mathcal{P}(m_1, \ldots, m_n)$ and $X(m)$ such that $\mathcal{P}$ is a predicate symbol of arity n, X is a predicate variable and $m, m_1, \ldots, m_n$ are terms of $\mathcal{L}^2$. We assume to have a 0-ary predicate symbol $\bot$ which represents falsity.
(3) The **formulas** of $\mathcal{L}^2$ are built from atomic formulas of $\mathcal{L}^2$ by the logical constants $\vee, \wedge, \to, \forall, \exists$, with quantifiers ranging over first-order variables $\alpha, \beta, \ldots$ and second-order variables $X, Y, \ldots$: if A, B are formulas, then $A \wedge B$, $A \vee B$, $A \to B$, $\forall \alpha\, A$, $\exists \alpha\, B$, $\forall X\, A$ are formulas. The logical negation $\neg A$ can be introduced, as usual, as an abbreviation of the formula $A \to \bot$ and the second-order existential quantification is defined as $\exists X\, A := \forall Y.\, (\forall\, X.\, A \to Y(\mathsf{c})) \to Y(\mathsf{c})$.
(4) As usual, if A and B are formulas of $\mathcal{L}^2$ and X is a predicate variable, we denote with $A[\lambda \alpha B / X]$ the formula obtained from A by replacing all its atomic subformulas of the form $X(m)$ with $B[m/\alpha]$ (without capturing free variables of B).

The natural deduction for LC_2 and $\mathsf{LC}_2^\star$ extends respectively the natural deduction for LC and the one for $\mathsf{LC}^\star$ with the following inference and reduction rules (see Girard [19]):

Second-Order Universal Quantification: $\dfrac{t : A}{\Lambda X\, t : \forall X\, A}$ $\qquad$ $\dfrac{t : \forall X\, A}{t(\lambda\alpha B) : A[\lambda\alpha B/X]}$

where in the left rule X does not occur free in the types of the free variables of A.

Reduction Rule for Universal Quantification: $(\Lambda X\, u)(\lambda\alpha B) \mapsto u[\lambda\alpha B/X]$

The Definition 2.3 of stack is of extended to $\mathsf{LC}_2^\star$ allowing expressions $(\lambda\alpha B)$, with B formula, to appear in the stack, and the Definition 2.4 of head redex is extended to the terms of $\mathsf{LC}_2^\star$ by saying that $(\Lambda X\, u)(\lambda\alpha B)$ is the **head redex** of $(\Lambda X\, u)(\lambda\alpha B)\,\sigma$ for every stack σ. The reduction relation $\succ$ for the terms of $\mathsf{LC}_2^\star$ is then defined as in Definition 2.5. In the following, we define $\mathsf{HN}^\star$ to be the set of normalizing proof terms of $\mathsf{LC}_2^\star$.

In order to define second-order realizability we need the concept of *realizability opponent*, which is nothing but a function mapping terms of $\mathcal{L}^2$ to arbitrary sets of stacks adapted to some fixed type. The idea is that an arbitrary realizability opponent represents the sets of tests that an arbitrary definition of realizability requires to pass in order to declare a term to be a realizer.

Definition 5.2 (Realizability Opponent)**.**

(1) A stack σ of $\mathsf{LC}_2^\star$ is said to be **adapted to a type** C, if for all terms t of type C, $t\,\sigma$ is still a term of $\mathsf{LC}_2^\star$.
(2) A **realizability opponent** of type $\lambda\alpha C$ is any function that maps each term m of $\mathcal{L}^2$ to a set of stacks adapted to $C[m/\alpha]$. We assume that for each realizability opponent $\mathcal{X}$ of type C there is in $\mathcal{L}^2$ an **opponent predicate constant** $\dot{\mathcal{X}}$ of type $\lambda\alpha C$ associated to it.

Realizability for $\mathsf{LC}_2^\star$ extends realizability for $\mathsf{LC}^\star$ to second-order quantification. The idea is the usual: we would like to define $t \Vdash \forall X\, A$ as: for all formulas B, $t\,(\lambda\alpha B) \Vdash A[\lambda\alpha B/X]$, but we cannot. So we define $t \Vdash \forall X\, A$ as $t \Vdash A$ for all possible definitions of realizability which X can be assigned to, that is, for all reducibility opponents that replace X.

Definition 5.3 (Classical Realizability for $\mathsf{LC}_2^\star$)**.** Assume t is a term of $\mathsf{LC}_2^\star$ and C is a formula of $\mathcal{L}^2$. We define by mutual induction the relation $t \Vdash C$ ("t is reducible of type C") and a set $||C||$ of stacks of $\mathsf{LC}_2^\star$ according to the form of C:

- $t \Vdash C$ if and only if $t : C$ and for all $\sigma \in ||C||$, $t\,\sigma \in \mathsf{HN}^\star$
- $||\dot{\mathsf{P}}|| = \{\epsilon\}$ if P is atomic
- $||\dot{\mathcal{B}}(m)|| = \mathcal{B}(m)$ for each realizability opponent $\mathcal{B}$
- $||A \to B|| = \{u \bullet \sigma \mid u \Vdash A \wedge \sigma \in ||B||\} \cup \{\epsilon\}$
- $||A \wedge B|| = \{\pi_0 \bullet \sigma \mid \sigma \in ||A||\} \cup \{\pi_1 \bullet \sigma \mid \sigma \in ||B||\} \cup \{\epsilon\}$
- $||A \vee B|| = \{[x.u, y.v] \bullet \sigma \mid \forall t.\ (t \Vdash A \implies u[t/x]\,\sigma \in \mathsf{HN}^\star) \wedge (t \Vdash B \implies v[t/y]\,\sigma \in \mathsf{HN}^\star)\} \cup \{\epsilon\}$
- $||\forall\alpha\, A|| = \{m \bullet \sigma \mid m \in \mathcal{L} \wedge \sigma \in ||A[m/\alpha]||\} \cup \{\epsilon\}$
- $||\exists\alpha\, A|| = \{[(\alpha, x).v] \bullet \sigma \mid \forall t.\ t \Vdash \dot{A}[m/\alpha] \implies v[m/\alpha][t/x]\,\sigma \in \mathsf{HN}^\star\} \cup \{\epsilon\}$
- $||\forall X\, A|| = \{(\lambda\alpha B) \bullet \sigma \mid \sigma \in ||A[\dot{\mathcal{B}}/X]||$ for some realizability opponent $\mathcal{B}$ of type $\lambda\alpha B\} \cup \{\epsilon\}$

The next proposition says that in the definition of $||A[\lambda\alpha B/X]||$, we can replace $\lambda\alpha B$ with the realizability opponent corresponding to it, transforming in this way an intensionally defined set into an extensionally defined object.

Proposition 5.4 (Comprehension)**.** *Let B a formula of $\mathcal{L}^2$. Suppose $\mathcal{B}$ is a realizability opponent such that*

$$\mathcal{B} = m \mapsto ||B[m/\alpha]||$$

Then for every formula A of $\mathcal{L}^2$

$$||A[\dot{\mathcal{B}}/X]|| = ||A[\lambda\alpha B/X]||$$

Proof. Standard, by induction on A (see Krivine [26]).

(1) $A = \mathcal{P}(m_1, \ldots, m_n)$, where $\mathcal{P}$ is a predicate constant symbol. Then, $A[\dot{\mathcal{B}}/X] = \mathcal{P}(m_1, \ldots, m_n) = A[\lambda\alpha B/X]$ and the thesis is trivial.
(2) $A = Y(m)$, where Y is a predicate variable. Then, if $Y \neq X$, the thesis is trivial, since we have
$$A[\dot{\mathcal{B}}/X] = Y(m) = A[\lambda\alpha B/X]$$
So let us suppose $Y = X$. Then
$$||A[\dot{\mathcal{B}}/X]|| = ||\dot{\mathcal{B}}(m)|| = \mathcal{B}(m) = ||B[m/\alpha]|| = ||A[\lambda\alpha B/X]||$$
(3) The other cases are straightforward. □

We extended Proposition 3.4 by showing that realizability is also sound with respect to second-order quantification elimination.

Proposition 5.5 (Properties of Realizability: ∀-Eliminations)**.**
If $t \Vdash \forall X\, A$, then for every formula B of $\mathcal{L}^2$, $t(\lambda\alpha B) \Vdash A[\lambda\alpha B/X]$.

Proof. Assume $t \Vdash \forall X\, A$. Let $\sigma \in ||A[\lambda\alpha B/X]||$; we must show $t(\lambda\alpha B)\,\sigma \in \mathsf{HN}^\star$. Let us consider a realizability opponent $\mathcal{B}$ such that

$$\mathcal{B} = m \mapsto ||B[m/\alpha]||$$

By Proposition 5.4, $\sigma \in ||A[\dot{\mathcal{B}}/X]||$. By Definition 5.3, $(\lambda\alpha B)\bullet\sigma \in ||\forall X A||$, and since $t \Vdash \forall X A$, we conclude $t(\lambda\alpha B)\,\sigma \in \mathsf{HN}^\star$. □

We extended Proposition 3.5 by showing that realizability is also sound with respect to second-order quantification introduction.

Proposition 5.6 (Properties of Realizability: ∀-Introductions)**.** *If for every realizability opponent $\mathcal{B}$ of type $\lambda\alpha B$, $u[\lambda\alpha B/X] \Vdash A[\dot{\mathcal{B}}/X]$, then $\Lambda X\, u \Vdash \forall X\, A$.*

Proof. Suppose that for every formula B of $\mathcal{L}^2$, $u[\lambda\alpha B/X] \Vdash A[\dot{\mathcal{B}}/X]$. Let $\sigma \in ||\forall X A||$. We have to show $(\Lambda X\, u)\,\sigma \in \mathsf{HN}^\star$. If $\sigma = \epsilon$, indeed $\Lambda X\, u \in \mathsf{HN}^\star$. Suppose then $\sigma = (\lambda\alpha B)\bullet\rho$, with $\rho \in ||A[\dot{\mathcal{B}}/X]||$ and $\mathcal{B}$ realizability opponent of type $\lambda\alpha B$. Since by hypothesis $u[\lambda\alpha B/X] \Vdash A[\dot{\mathcal{B}}/X]$, we have $u[\lambda\alpha B/X]\,\rho \in \mathsf{HN}^\star$; moreover,

$$(\Lambda X\, u)(\lambda\alpha B)\,\rho \succ u[\lambda\alpha B/X]\,\rho$$

Therefore, $(\Lambda x\, u)(\lambda\alpha B)\,\rho \in \mathsf{HN}^\star$. □

The Adequacy Theorem is readily extended to second-order realizability.

Theorem 5.7 (Adequacy Theorem)**.** *Suppose that $w : A$ in the system LC_2, with w having free variables among $x_1^{A_1}, \dots, x_n^{A_n}$. Let $r_1, \dots, r_k$ and $\dot{\mathcal{B}}_1, \dots, \dot{\mathcal{B}}_m$ be respectively terms of $\mathcal{L}^2$ and realizability opponents of type $\lambda\beta_1 B_1, \dots, \lambda\beta_m B_m$. For every formula C, set $\overline{C} = C[r_1/\alpha_1 \cdots r_k/\alpha_k \ \dot{\mathcal{B}}_1/X_1 \cdots \dot{\mathcal{B}}_m/X_m]$. If there are terms $t_1, \dots, t_n$ such that*

$$\text{for } i = 1, \dots, n, \ t_i \Vdash \overline{A}_i$$

then

$$w[r_1/\alpha_1 \cdots r_k/\alpha_k \ \lambda\beta_1 B_1/X_1 \cdots \lambda\beta_m B_m/X_m][t_1/x_1^{\overline{A}_1} \cdots t_n/x_n^{\overline{A}_n}] \Vdash \overline{A}$$

Proof. For any term v, we define

$$\overline{v} := v[r_1/\alpha_1 \cdots r_k/\alpha_k \ \lambda\beta_1 B_1/X_1 \cdots \lambda\beta_m B_m/X_m][t_1/x_1^{\overline{A}_1} \cdots t_n/x_n^{\overline{A}_n}]$$

We proceed by induction on w. Consider the last rule $\mathscr{R}$ in the derivation of $w : A$: we just have to deal with the second-order cases, the other ones have been settled in the proof of Theorem 3.8.

(1) If $\mathscr{R}$ is the second-order $\forall E$ rule, then $w = u\,(\lambda\alpha B)$, $A = C[\lambda\alpha B/X]$ and $u : \forall X\, C$. So, $\overline{w} = \overline{u}\,(\lambda\alpha\overline{C})$. By inductive hypothesis $\overline{u} \Vdash \forall X\, \overline{C}$ and so $\overline{u}\,(\lambda\alpha\overline{B}) \Vdash \overline{C}[\lambda\alpha\overline{B}/X]$ by Proposition 5.5.

(2) If $\mathscr{R}$ is the second-order $\forall I$ rule, then $w = \Lambda X\, u$, $A = \forall X\, B$ and $u : B$ (with X not occurring free in the types $A_1, \dots, A_n$ of the free variables of u). So, $\overline{w} = \Lambda X\, \overline{u}$, since we may assume $X \neq X_1, \dots, X_m$. By Proposition 5.6, it is enough to prove that $\overline{u}[\lambda\alpha B/X] \Vdash \overline{B}[\dot{\mathcal{B}}/X]$ for every realizability opponent $\mathcal{B}$ of type $\lambda\alpha B$, which amounts to showing that the induction hypothesis can be applied to u. For this purpose, we observe that, since $X \neq X_1, \dots, X_m$, for $i = 1, \dots, n$ we have

$$t_i \Vdash \overline{A}_i = \overline{A}_i[\dot{\mathcal{B}}/X]$$

□

As consequence of the Adequacy Theorem 5.7, we obtain that every typed term of LC_2 is normalizable by head reduction.

Corollary 5.8 (Normalization for LC_2)**.** *Suppose $t : A$ in LC_2. Then $t \in \mathsf{HN}$.*

We can finally prove that second-order Dummett's logic LC_2 is Herbrand constructive.

Theorem 5.9 (Second-Order Herbrand Disjunction Extraction)**.** *Let $\exists\alpha\, A$ be any formula. Suppose*

$$\mathsf{LC}_2 \vdash t : \exists\alpha\, A$$

Then there is a proof term u such that $t \succ^* u \in \mathsf{HNF}$, $\mathsf{LC}_2 \vdash u : \exists\alpha\, A$ and u is an Herbrand normal form

$$u = (m_0, v_0) \parallel_{a_1} (m_1, v_1) \parallel_{a_2} \dots \parallel_{a_k} (m_k, v_k)$$

Moreover,

$$\mathsf{LC}_2 \vdash A[m_1/\alpha] \vee \cdots \vee A[m_k/\alpha]$$

Proof. As the proof of Theorem 4.3. □

Acknowledgments

I would like to thank Agata Ciabattoni: this work arose, and greatly benefited, from conversations with her. I would also like to thank Francesco Genco for interesting exchanges about the topic.

References

[1] F. Aschieri, *Interactive Realizability for Classical Peano Arithmetic with Skolem Axioms*. Proceedings of CSL 2012, Leibniz International Proceedings in Informatics, vol. 16, pp. 31–45, 2012.

[2] F. Aschieri, *Interactive Realizability for Second-Order Heyting Arithmetic with $EM1$ and $SK1$*, Mathematical Structures in Computer Science, vol. 24, n. 6, 2013.

[3] F. Aschieri, *Strong Normalization for HA + EM1 by Non-Deterministic Choice*, Proceedings of First Workshop on Control Operators and their Semantics 2013 (COS 2013), Electronic Proceedings in Theoretical Computer Science, vol. 127, pp. 1–14, 2013

[4] F. Aschieri, S. Berardi, G. Birolo, *Realizability and Strong Normalization for a Curry-Howard Interpretation of HA + EM1*, CSL 2013, Leibniz International Proceeding in Computer Science, vol. 23, pp. 45–60, 2013.

[5] F. Aschieri, M. Zorzi, *Non-Determinism, Non-Termination and the Strong Normalization of System T*, Proceedings of TLCA 2013, LNCS, vol. 7941, pp. 31–47, 2013.

[6] F. Aschieri, M. Zorzi, *A "Game Semantical" Intuitionistic Realizability Validating Markov's Principle*, Post-Proceedings of TYPES 2013, Leibniz International Proceedings in Informatics, vol. 26, pp. 24–44, 2014.

[7] F. Aschieri, M. Zorzi, *On Natural Deduction in Classical First-Order Logic: Curry-Howard Correspondence, Strong Normalization and Herbrand's Theorem*, Theoretical Computer Science, vol. 625, pp. 125–146, 2016.

[8] A. Avron, *Hypersequents, logical consequence and intermediate logics for concurrency*, Annals of Mathematics and Artificial Intelligence, vol. 4, pp. 225–248, 1991.

[9] M. Baaz, A. Ciabattoni, C. Fermüller, *A Natural Deduction System for Intuitionistic Fuzzy Logic*, in: Lectures on Soft Computing and Fuzzy Logic, pp. 1–18, A. Di Nola, G. Gerla eds., Physica-Verlag, 2000.

[10] M. Baaz, A. Ciabattoni, C. Fermüller, *Hypersequent Calculi for Gödel Logics - a Survey*, Journal of Logic and Computation, vol. 13, n. 6, pp. 835–861, 2003.

[11] A. Beckmann, N. Preining, *Hyper Natural Deduction*, Proceedings of LICS 2015, pp. 547–558, 2015.

[12] L. E. J. Brouwer, *Collected works*, Philosophy and foundations of mathematics; edited by A. Heyting, North-Holland, Elsevier Science Publishing, 1975.

[13] S. Buss, *On Herbrand's Theorem*, Proceedings of Logic and Computational Complexity, LNCS, vol. 960, pp. 195–209, 1995.

[14] G. Corsi, *A Cut-free Sequent Calculus for Dummett's LC Quantified*, Mathematical Logic Quarterly, vol. 35, n. 4, pp. 289–301, 1989.

[15] V. Danos, J.-L. Krivine, *Disjunctive Tautologies as Synchronisation Schemes*, Proceedings of CSL, LCNS, vol. 1862, pp. 292–301, 2000.

[16] M. Dummett, *A Propositional Calculus with Denumerable Matrix*, Journal of Symbolic Logic, vol. 24, n. 2, pp. 97–106, 1959.

[17] M. Dummett, *The Logical Basis of Metaphysics*, Harvard University Press, 1991.

[18] G. Gentzen, *Untersuchungen über das logische Schliessen*, Mathematische Zeitschrift, vol. 39, pp. 176–210, 405–431, 1935.

[19] J.-Y. Girard and Y. Lafont and P. Taylor, *Proofs and Types*. Cambridge University Press 1989.

[20] K. Gödel, *On the Intuitionistic Propositional Calculus*, 1932, in *Kurt Gödel. Collected Works, vol. I*, Oxford University Press, pp. 223–224, 1986.

[21] P. de Groote, *Strong Normalization for Classical Natural Deduction with Disjunction*, Proceedings of TLCA 2001, pp. 182–196, 2001.

[22] Y. Hirai, *A Lambda Calculus for Gödel-Dummett Logic Capturing Waitfreedom*, Proceedings of FLOPS, pp. 151–165, 2012.

[23] S. C. Kleene, *On the Interpretation of Intuitionistic Number Theory*, Journal of Symbolic Logic vol. 10, n. 4, pp. 109–124, 1945.
[24] G. Kreisel, *On Weak Completeness of Intuitionistic Predicate Logic*, Journal of Symbolic Logic, vol. 27, n. 2, pp. 139–158, 1962.
[25] J.-L. Krivine, *Classical Logic, Storage Operators and Second-Order lambda-Calculus*, Annals of Pure and Applied Logic, vol. 68, n. 1, pp. 53–78, 1994.
[26] J.-L. Krivine, *Classical Realizability*, Interactive models of computation and program behavior, Panoramas et synthèses, pp. 197–229, 2009. Société Mathématique de France.
[27] O. Lahav, A. Avron, *A cut-free Calculus for second-order Gödel Logic*, Fuzzy Sets and Systems, vol. 276, pp. 1–30, 2015.
[28] M. Parigot, *Proofs of Strong Normalization for Second-Order Classical Natural Deduction*, Journal of Symbolic Logic, vol. 62, n. 4, pp. 1461–1479, 1997.
[29] J. von Plato, *Gentzen's Proof of Normalization for Natural Deduction*, Bulletin of Symbolic Logic, vol. 14, n. 2, pp. 204–257, 2008.
[30] D. Prawitz, *Ideas and Results in Proof Theory*, Proceedings of the Second Scandinavian Logic Symposium, pp. 235–306,1971.
[31] M. H. Sorensen, P. Urzyczyn, *Lectures on the Curry-Howard isomorphism*, Studies in Logic and the Foundations of Mathematics, vol. 149, Elsevier, 2006.
[32] W. Tait, *Normal Form Theorem for Bar Recursive Functions of Finite Type*, Proceedings of the Second Scandinavian Logic Symposium, pp. 353–267, 1971.
[33] G. Takeuti, *On a Generalized Logical Calculus*, Japanese Journal of Mathematics, vol. 23, 39–96, 1953.
[34] A. Tiu, *A Hypersequent System for Gödel-Dummett Logic with non-Constant Domains*, TABLEAUX, Lecture Notes in Computer Science, vol. 6793, pp. 248–262, 2011.