Intuitionistic Logic of Proofs with dependent proof terms

Giuseppe Primiero
FWO – Centre for Logic and Philosophy of Science
Ghent University (Belgium)

Abstract

The basic logic of proofs extends the usual propositional language by expressions of the form "s is a proof of A", for any proposition A. In this paper we explore the extension of its intuitionistic fragment to a language including expressions of the form "t is a proof of B, dependent from s being a proof of A". We aim at laying down a ground comparison with equivalent constructions present in theories of dependent types, especially those similarly based on the Brouwer-Heyting-Kolmogorov semantics. We further translate this extended language to a natural deduction calculus which allows for a double interpretation of the construction on which a proof term may depend: as actually proven, or valid assumption, or as possibly proven, locally true assumption. We show meta-theoretical properties for this calculus and explain normalisation to a language with only unconditional proofs. We conclude by stating the characterization of our calculus with standard intutionistic logic of proofs.

1 Introduction

The Logic of Proofs (LP, [4], [5], recently presented in its more general epistemic setting as Justification Logic in [6]) bridges the BHK semantics to a real semantics of proofs. It extends the usual connectives of propositional logic by a a proof polynomial s attached to the appropriate proposition A, so that the formula $s : A$ reads: "s is a proof of A". Furthermore, proof terms are combined by standard propositional connectives. It has been shown how each theorem from the modal logic $S4$ corresponds to an appropriate formula in LP, and how in general the latter provides a natural interpretation of the semantics of mathematical proofs fro $S4$, a task first posed by Gödel [17]. In [5], the BHK semantics at the basis of LP is characterized in its original aspiration of determining meanings in terms of provability, going back to Brouwer's definition of truth in terms of proof. The fragment that relies on an Intuitionistic Logic has been explored [7] and proven to be complete with respect to Heyting Arithmetics in [11].

A complementary interpretation of BHK was given in [19], where the computational counterpart was introduced. A representative of this interpretation are the intuitionistic-based theories of types, satisfying the proofs-as-terms interpretation known as Curry-Howard correspondence, in the style of Martin-Löf type theory [20] or the Logical Frameworks, from Automath [12], to ELF [18]

and their many programming variations, see [22], [23]. In particular in type-theories, the generalization from the propositional to the first-order setting is justified by the extension to dependent types. A dependent type is a family of types parametrized by another type. Here, a formula of the form $b : B[x/a : A]$ with a a term and x in the set of variables for values, stands for the validity of type B, provided the substitution of a variable x in A by a term a is obtained. The latter represents the reconstruction of the missing computational information by renaming all the closed variables in the type A from which B depends. In this way one constructs the (set of) propositional function(s) from A to B. In Dependent Type Theories this functional relation is used to formalize quantifiers; the rule for justifying the implication connective is then a result of explaining the elimination rule for the universal quantification, see [20, p.34].

A notion of functional expression for LP has also been given. In [14], the quantification extension of LP is obtained by having in the language not only constants, but also function symbols of various arities which are called "primitive function symbols". The quantifiers are defined then as ranging over proof terms. In [8], a first-order logic of proofs is defined, which interprets formulas of the form $p : A(x)$ in each of the two following senses:

1. p proves $A(x)$ for a given value of the parameter x;
2. p is a proof of a formula with a free variable x (hence $p : A(x)$ does not depend on x).

The relation between the provability and the computational interpretations of BHK semantics is still open to exploration. One aspect of such topic is precisely an analysis of the current relation between type theories with types depending on typed terms and its counterpart in LP. This corresponds to the introduction of functionals expressions where the variable is explicitly typed by means of a corresponding proof polynomial.

The first aim of this paper is to answer to this request, providing an extension of the Intuitionistic fragment of the Logic of Proofs which contains a notion of dependent proof that mimics that of dependent types, or functional expressions with terms explicitly typed in another proposition. We give this by a language called ILP_{dep}, which contains formulas of the form $\langle s \rangle t : B[A]$, which reads: "$t$ is a proof of B, dependent from s being a proof of A".

A second task of this paper is that of providing a Natural Deduction interpretation of ILP_{dep} – called $\mathbf{ILP}_{nd\Diamond}$ – which gives more structure in the use of dependent terms. This is obtained by distinguishing functional expressions from the corresponding implicational instances. To explain this, let us refer again to the mentioned relation between universal quantification and implication: in dependent type theories, an introduction of $A \supset B$ is obtained from the corresponding universal quantification by suppressing the proof; completeness of the rule is given by the corresponding process of reconstructing the computational value on A. In the following, we mimic this relation by introducing functions as derivability from True Assumptions, and interpreting implications as the result of discharging those assumptions, so that it corresponds to derivability from Valid Assumptions.

In [9], a Natural Deduction interpretation for ILP was already presented.[1]

[1] A previous attempt to formalize in a λ-calculus the conceptual mapping between Curry-Howard style proofs, λ-terms and proofs in the sense of the logic of proofs was given in [3].

It contains formulas expressing Hypothetical Judgements with Evidence derivable from *Valid Assumptions* of the form $v : A$ and *True Assumptions* of the form $a : A$ stating that A holds at the current world. This system for Hypothetical Judgements with Explicit Evidence internalizes the notion of proof to express in the language the sentence "s is a proof of A".[2]. The internalization rule says that provided a proof s for A holds under valid assumptions Δ, then under any additional true assumptions Γ, there is a proof that s is a proof of A. We shall introduce in $\mathbf{ILP}_{nd\Diamond}$ a new expression $\Delta; \Gamma \vdash s :: A$ to express that s is a proof of A dependent on some true assumption contained in Γ. Correspondingly, the internalization rule says that provided a dependent proof s for A holds, then there is a proof $?s$ that s is a proof of A, conditional on the verification of assumptions in Γ. We claim that this offers an appropriate provability interpretation of a possibility operator in the intuitionistic setting, and it allows representing contents whose proof is dependent locally on the proof of some other proposition.[3]

We suggest here to combine these two issues:

1. provide a distinction between a proof term holding under Valid Assumptions and one holding under True Assumptions; the former corresponds to an unconditional proof term for implication, the latter a dependent one (as term from term) simulating a functional structure;

2. use the notion of dependent term to formulate a basic provability interpretation for a $\Diamond$ operator.

A final task of this paper is to establish the correct characterization of ILP_{dep} with respect to the standard intuitionistic fragment of LP.

In section 2 we shall present ILP extended with the notion of dependent proof, defining equality rules, dependence from more than one term, quantified expressions and some examples. In section 3 we shall introduce the Natural Deduction system whose crucial property is to separate derivability from valid and true assumptions, allowing an interpretation of implication and of functional expressions. In section 3.1 we further extend this language with an appropriate equality relation on terms. The two final sections are dedicated to metathereotical issues: in section 4 we prove that all connectives are locally sound and complete by contractions and expansions derivations; in section 5 we prove that the extension by the notion of dependent proof is 'safe', by showing weak and strong normalization and confluence for the system. On this basis, we shall be able to state our main result in section 7, where ILP_{dep} is characterized in view of ILP.

[2]For preserving uniformity of notation and readability in the formulation of the two calculi introduced in this paper, I will privilege the more common notation $s : A$ for expressions of LP in place of the one given in [9] which uses expressions of the form $[\![s]\!]A$. I will also make uniform use of the term 'proof' in place of the term 'evidence', which currently bears a larger range of meanings related to its epistemic uses.

[3]In [10], a semantics derived from $\mathbf{LP}_{nd}$ is used to interpret mobile code; the authors mention that using a possibility modality one can interpret code at remote locations, provided a provability interpretation of $\Diamond$ in the intuitionistic setting is given. Cf. [10, §8].

2 ILP_{dep}: Axioms and Inference Schemes

In the following, we shall refer to an extension of the usual interpretation of ILP, obtained by introducing a dependent term to formulate expressions of the form:

> "t is a proof of B, dependent from s being a proof of A".

Definition 1 (Language)**.** *We denote with ILP_{dep} a language that contains a countable set of symbols $A, B, \ldots$ for propositions; individual variables $x, y, \ldots$ and constants $s, t, \ldots$ for proof terms; functional expressions $B[A]$; functional symbols for operations on proof terms: $\cdot, !, +, ., \exists, \forall$.*

Definition 2 (Proof Terms)**.** *In ILP_{dep} each proof variable or proof constant is a proof term; we denote the fact that s is the proof term of proposition A by the formula $s\!:\!A$; if s and t are proof terms, so are: $s \cdot t$, $!s$, $s + t$. Moreover, we denote the fact the t is a proof term of proposition B dependent from the fact that s is a proof of proposition A by the formula $\langle s \rangle t : B[A]$; if $\langle s \rangle t$ is a proof term, so are $(s)t$ and $t.s$.*

ILP_{dep} extends therefore the standard set of proof polynomials of ILP composed by s, $s \cdot t$, $s + t$, $!s$, with some more: the dependent term $\langle s \rangle t$ reads: "t is a proof provided s is a proof" (of distinct propositions); the term $(s)t$ reads: "the proof term t, abstracting from the computational content of proof term s"; the applied term $t.s$ reads: "the proof term t (with no dependencies) is applied to the reconstructed computational content of proof term s (with no dependencies)". The intended meaning of the latter two terms is to express within a proof polynomial the computational processes that dependent terms perform in multiple steps.

Definition 3 (Axioms and Rules of ILP_{dep})**.** *Axioms and Rules of the system are:*

A0. *Axioms schemes of intuitionistic logic in the the language of LP*

A1. $s\!:\!A \supset A$ *(Unconditional Proof)*

A2. $s\!:\!A \supset !s\!:\!(s\!:\!A)$ *(Proof Checker)*

A3. $s\!:\!(A \supset B) \supset (t\!:\!A \supset s \cdot t\!:\!B)$ *(Application for Unconditional Proof)*

A4. $s\!:\!A \supset s + t\!:\!A;\ t\!:\!A \supset s + t\!:\!A$ *(Sum)*

A5. $A \vdash t\!:\!B \supset \langle s \rangle t\!:\!B[A]$ *(Dependent Proof)*

A6. $\langle s \rangle t\!:\!B[A] \supset (s)t.s\!:\!B$ *(Application for Dependent Proof)*

R1. $\Gamma \vdash A \supset B$ *and* $\Gamma \vdash A$ *implies* $\Gamma \vdash B$ *(Modus Ponens)*

R2. *If* $\mathbf{A}$ *is an axiom* $\mathbf{A0.} - \mathbf{A6.}$ *and c is a proof constant, then* $\vdash c\!:\!\mathbf{A}$ *(Necessitation)*

Unconditional Proof reads: "if s is a proof of A, then A holds". Proof checking reads: "if s is a proof of A, then $!s$ is a proof of the sentence 's is a proof of A' ". Application for Unconditional Proof reads: "If there is a proof s that A implies B, then a proof t of A implies that $s{\cdot}t$ is a proof of B". Sum reads: "if s is a proof of A, then $s+t$ is also a proof of A" (alternatively, for t a proof of A). Dependent Proof reads: "If t is a proof of B under assumption A, then there is a proof t of B dependent on a proof s of A". Application for Dependent Proof reads: "If there is a proof t of B dependent on a proof s of A, then a proof t abstracted from s followed by an application to the reconstructed proof s is a proof of B". This application is reducible to a different proof of the same type, obtained by the Application for Unconditional proof: this shows that the notion of unconditional proof supports a polymorphism of proofs embedding procedural steps in one polynomial. Modus Ponens holds, and Necessitation remains valid for axioms.

In the following section, equality rules for proof terms and reduction of the previous rules to computational ones are given.

2.1 Equality Rules

By the notion of dependent proof $\langle s\rangle t : B[A]$, the language is able to mimic the notion of functional expression: "t is a proof of B, dependent from s being a proof of A". This gives a way to formulate in ILP expressions of the form: "Let A be a proposition with proof s, and B a proposition dependent on A" as a formal assumption over term s. This is obtained by constructing the proof for B *assuming* a proof in A. The next step is defining extensional equivalence over such terms:[4]

$$\frac{A \equiv A' \supset s : A \equiv s' : A' \qquad A \vdash t : B \equiv A' \vdash t' : B'}{\langle s\rangle t : B[A] \equiv \langle s'\rangle t' : B'[A']}\ \text{Equality on DepProof}$$

which says that we get identical dependent terms $\langle s\rangle t$, $\langle s'\rangle t'$ for $B[A], B'[A']$ when equivalent proof terms s, s' respectively for equivalent A, A' are formulated. Discharging of such dependency relation is given by an instance of the Application for Dependent Proof, with the corresponding Equality Rule:

$$\frac{\langle s\rangle t : B[A] \equiv \langle s'\rangle t' : B'[A'] \qquad s \equiv s' : A}{(s)t.s : B \equiv (s')t'.s' : B'}\ \text{Equality on Application}$$

When the construction in the depending term is discharged, Dependent Proof reduces to an implicational relation; hence, from it one can define an abstraction on terms:

$$\frac{\langle s\rangle t : B[A]}{(s)t : A \supset B}\ \text{Abstraction}$$

[4]In the following the equivalence sign $\equiv$ is used to refer to formal equivalence among formulas; the identity sign $=$ to express identity of values. A first approach to an interpretation of types dependent on terms for LP was given in a λ-calculus version in [2].

which reads as follows: "if t is a proof for B, depending on s being a proof for A, then there is a proof t abstracting from s of $A \supset B$". Its explanation is given by reduction which equals the corresponding Application:

$$\frac{s:A \qquad \langle s\rangle t:B[A]}{(s)t.s = s\cdot t:B}\ \beta\text{-rule}$$

where the first polynomial in the conclusion can be read as an abstraction followed by an application. It also has appropriate identity rules:

$$\frac{\langle s\rangle t \equiv \langle s\rangle t':B[A]}{(s)t \equiv (s)t':A\supset B}\ \xi\text{-rule}$$

$$\frac{\langle s\rangle t:B[A]}{(s)t = (s')t:A\supset B}\ \alpha\text{-rule}$$

where variables in s' are not free in s.

$$\frac{\langle s\rangle t:B[A]}{(s)t.s \equiv \langle s\rangle t:B[A]}\ \eta\text{-rule}$$

where variables in s are not free variables in t. In the latter rule, the formal identity is obtained for the right-left direction is given by Axiom $A6$ and in the opposite direction by constructing $s:A$ from an Abstraction Rule, followed by a construction for $A \vdash t:B$ and an instance of Axiom $A5$.

2.2 Proof Dependent from more than one term

We can now provide the formal translation between dependent terms and truth under more than one assumption. The basic case with two assumptions is of the from $\langle s_1, s_2\rangle t:B[A_2[A_1]]$ which reads informally as follows:

> "t is a proof of B, dependent from s_2 being a proof of A_2, which in turn depends from s_1 being a proof of A_1".

This means that the structure of a context of assumptions is given recursively by sets of dependent proofs. In the following, for purposes of readability, we shall adopt the convention that multiple dependencies are written in increasing order of index, their reading remaining the same as the above. The general format is obtained by repeated abstractions on dependent terms:

$$\frac{\langle s_1 \ldots s_n\rangle t:B[A_1,\ldots,A_n]}{(s_1\ldots(s_{n-1}(s_n)))t:(A_1,\ldots,A_n)\supset B}$$

The informal semantics of this rule says: "if t is a proof for B dependent on proofs for $A_1,\ldots,A_n$, then t abstracting from $s_1,\ldots s_n$ is a proof that $A_1,\ldots A_n$ imply B".

By repeated application we obtain the inverse operation:

$$\frac{(s_1(s_2(\ldots,(s_{n-1}(s_n))))t:(A_1,\ldots,A_n)B \qquad s_1:A_1, s_1\cdot s_2:A_2\cdot,\ldots,\cdot s_{n-1}\cdot s_n:A_n}{t.(s_1.s_2.,\ldots,.s_n):B}$$

which reads:

> "if t is a proof for B provided proofs for $A_1,\ldots,A_n$ and provided s_n is a proof of A_s provided s_{n-1} is a proof of S_{n-1} up to s_1 is a proof of A_1, then t is a proof for B dependent on proofs s_1 applied to s_2, then applied to s_3 up to s_n".

2.3 Rules for Quantifiers

By our notion of dependent proof we have a direct translation of $\langle s\rangle t:B[A]$ in terms of $A\vdash B$ and so $(a)b:(A)B$ which reduces to $b:B(a)$. We hence use quantifiers to express the validity of the formula $B(x)$ parametrised with $x:A$.

$$\frac{a:A \qquad \langle a\rangle b:B[A]}{(x)b.x:\forall x:A.B(x)}\ \forall\text{-introduction}$$

$$\frac{a:A \qquad \langle a\rangle b\equiv\langle a\rangle b':B[A]}{(x)b.x\equiv(x)b'.x:\forall x:A.B(x)}\ \forall\text{-equality}$$

$$\frac{d:\forall x:A.B(x) \qquad a:A}{d\cdot a:B(x)}\ \forall\text{-elimination}$$

$$\frac{a:A \qquad a\cdot b:B}{(x)b.a:\exists x:A.B(x)}\ \exists\text{-introduction}$$

$$\frac{a\equiv a':A \qquad \langle a\rangle b\equiv\langle a'\rangle b:B[A]}{(x)b.a\equiv(x)b.a':\exists x:A.B(x)}\ \exists\text{-equality}$$

$$\frac{d:\exists x:A.B(x) \qquad \langle d\rangle c:C[\exists x:A.B(x)]}{c\cdot d:C}\ \exists\text{-elimination}$$

2.4 Some Examples

2.4.1 Example 1: Relations as Functions

Show that a function $C(y)[y:(B(x)[x:A]]$ is given by a function $C(x,y)$ with arguments respectively in A and $B[A]$. So we assume $a:A$, construct $\langle a\rangle b:B[A]$ and $\langle a,b\rangle c:C[B[A]]$. We want an element of

$$C(x,y)[y:(B(x)[x:A])] \supset (x)(y)c.(a,b):C$$

So we construct

$$\dfrac{\dfrac{\dfrac{\dfrac{A\vdash b:B \qquad B\vdash c:C}{\langle a,b\rangle c:C[B[A]]}\,\text{A5}}{(a,b)c:(A\supset(B\supset C))}\,\text{Repeated Abstraction}}{c.((a).b)):C}\,\text{Repeated Application}}{(a\cdot b)\cdot c:C}\,\beta\text{-rule}$$

2.4.2 Example 2: Identity of Functions

Show that for any elements a, a', if $a \equiv a' : A$, then $\langle a\rangle b : B[A] \supset \langle a'\rangle b : B[A]$. For this it is enough to show that $\langle a\rangle b:B[A] \equiv \langle a'\rangle b:B[A]$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\langle a\rangle b:B[A] \qquad \dfrac{a:A \qquad a':A \qquad A\equiv A}{a\equiv a':A}}{\langle a'\rangle b:B[A]}}{(a')b:A\supset B}}{(a\equiv a':A)b.a\equiv b.a':B}}{\langle a\rangle b:B[A]\equiv\langle a'\rangle b:B[A]}}{\langle a\rangle b:B[A]\supset\langle a'\rangle b:B[A]}$$

2.4.3 More Examples: Axioms

Prove that for all propositions A, $A \supset A$:

$$\dfrac{\dfrac{\dfrac{a:A}{A}\,\text{A1} \qquad A\vdash A}{\langle a\rangle a:A[A]}\,\text{A5}}{(a)a:A\supset A}\,\text{Abstraction}$$

Prove that for all propositions A, B: $A \supset (B \supset A)$ from $\langle a\rangle b:B[A]$:

$$\dfrac{\dfrac{\dfrac{\dfrac{a:A \qquad A\vdash b:(B\supset A))}{\langle a\rangle b:(B\supset A)[A]}\,\text{A5}}{\langle b,a\rangle b:A[B[A]]}\,\text{A5}}{(b)a:(A\supset(B\supset A))}\,\text{Abstraction}}{(a)b.a:A\supset(B\supset A)}\,\beta\text{-rule}$$

This is equivalent to the following derivation using ∀-introduction:

$$\dfrac{\dfrac{a:A \qquad \langle a\rangle b:B[A]}{(b).x:B(x)\supset A}\ \text{Abstraction}}{(a,b).x:(\forall x:A.B(x))}\ \forall\text{-intro}$$

Prove that for all propositions A, B: $A \supset (B \supset (A \wedge B))$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A\supset(B\supset(A\wedge B))}{c:(A\supset(B\supset(A\wedge B)))}\ \text{R2}}{c:=(a)s:(B\supset(A\wedge B))}\ \text{Abstraction}}{\langle a\rangle s:(B\supset(A\wedge B))[A]}\ \text{A5}}{(a)s.t:(A)B\supset(A\wedge B)}\ \text{Application}}{(b,a)s.t:(B)(A\wedge B))}\ \text{Abstraction}}{\langle(a,b)\rangle s.t:(A\wedge B)[B[A]]}\ \text{A5}}{(a,b)s.t.(a,b):A\wedge B}\ \text{Application}$$

A derivation which does not make use of the rule of constant specification $R2$:

$$\dfrac{\dfrac{\dfrac{\dfrac{a:A \qquad \langle a\rangle b:B[A]}{(x)b.a:\exists x:A.B(x)}\ \exists\text{-intro}}{(b)(x)b.a:B(x)\supset\exists x:A.B(x)}\ \text{Abstraction}}{(a)(b)(x)b.a:\forall x:A.B(x)\supset\exists x:A.B(x)}\ \forall\text{-intro}}{a:A\supset(b:B\supset(x)b.a:A\wedge B)}\ \text{Application}$$

where B does not depend on x. Notice that $(x)b.a$ can be further reduced to $a \cdot b$ by β-rule.

Prove the axiom $(A \supset B) \supset A \supset B$:

$$\dfrac{\dfrac{\dfrac{a:A \qquad \langle a\rangle b:B[A]}{(b).x:B(x)\supset A}\ \text{Abstraction}}{(a)b.a:B}\ \text{Application}}{(a)b.a:(A\supset B)\supset a:A\supset b:B}\ \text{R1}$$

3 The Language $\mathbf{LP}_{nd\Diamond}$

We now proceed with defining a natural deduction version of our calculus for Dependent Proof, called $\mathbf{LP}_{nd\Diamond}$.[5] The main aim is to translate unconditional proof as a term derivable under valid assumptions and dependent proof as a term derivable under true assumptions. This will give us even more structure

[5]The use of the subscript $\Diamond$ is intentional in explicitly relating our ? internalization operator for dependent proof to a possibility reading.

to be able to characterize the relation between dependent and unconditional proofs. In the new calculus $\mathbf{LP}_{nd\Diamond}$ the formula $s : A$ expresses the validity of A by proof s holding under satisfied conditions; the formula $s :: A$ expresses validity of A, dependently from the verification of a locally valid s. In terms of the system $\mathbf{LP}_{nd\Diamond}$ we establish some meta-theoretical properties as confluence and strong normalisation. This in turn will allow to give a full characterization also for ILP_{dep}.

We shall consider first Judgements with Unconditional Proof as those derived from Valid Assumptions only:

$$\Delta; \cdot \vdash s : A \qquad \text{UnProof}$$

which reads "s is a proof of A, derivable from valid assumptions Δ"; an alternative, but equivalent, reading is: "s is a proof of A holding at all states accessible from Δ". $UnProof$ further induces A valid under any True Assumption.

Then we consider Judgements with Dependent Proof as those derived from Valid *and* True Assumptions:

$$\Delta; \Gamma \vdash s :: A \qquad \text{DepProof}$$

which reads "s is a proof of A, derivable from Valid Assumptions Δ and dependent on True Assumptions Γ"; an alternative, but equivalent reading is: "s is a proof of A holding at states accessible from Δ where Γ holds". $DepProof$ induces A valid under no further true assumptions.

Definition 4 (Language)**.** *The syntax is defined by the following alphabet:*

Proof Terms $s := x \mid \lambda v : A.s \mid t \cdot s \mid \langle s \rangle t \mid (s)t.s \mid !s \mid XTRT\ s\ AS\ v : A\ IN\ s \mid ?s \mid ASSM\ s\ AS\ a : A\ IN s$

Propositions $A := P \mid A \wedge B \mid A \supset B \mid B[A]$

Truth Contexts $\Gamma := \cdot \mid \Gamma, a : A$

Validity Contexts $\Delta := \cdot \mid \Delta, v : A$

A proposition is either a propositional variable P, a conjunction, an implication or a function $B[A]$ where a proposition B depends from a proposition A. A proposition can be constructed as a valid proposition $s : A$, or as a true proposition $s :: A$, or by a term t for B dependently valid from another term s for A, $\langle s \rangle t : B[A]$. Truth and validity contexts remain as usual sequences of propositions with "$\cdot$" denoting the empty context. We reserve the notation $a/v : A$ for (true/valid) Assumptions only.

Definition 5 (The Logic $\mathbf{LP_{nd\Diamond}}$)**.** $\mathbf{LP_{nd\Diamond}}$ *is defined by the following schemes:*

$$\frac{}{\Delta, v : A; \Delta' \vdash s : A}\ ValVar$$

$$\frac{\Delta, v : A; \cdot \vdash s : B}{\Delta; \Gamma \vdash \lambda v : A.s : A \supset B} \supset I \qquad \frac{\Delta; \cdot \vdash s : A \supset B \qquad \Delta; \cdot \vdash t : A}{\Delta; \Gamma \vdash s \cdot t : B} \supset E$$

$$\frac{}{\Delta, a : A; \cdot \vdash s :: A}\ TruVar$$

$$\frac{\Delta, a:A \vdash t :: B}{\Delta; \cdot \vdash \langle s \rangle t : B[A]}\ DepProof\ \ Formation$$

$$\frac{\Delta; \Gamma \vdash \langle s \rangle t : B[A] \qquad \Delta; \Gamma \vdash s : A}{\Delta; \Gamma \vdash (s)t.s : B}\ Application$$

$$\frac{\Delta; \cdot \vdash s : A}{\Delta; \Gamma \vdash !s : (s : A)}\ !I \qquad \frac{\Delta; \cdot \vdash !s : (s : A) \qquad \Delta, v : A \vdash t : B}{\Delta; \Gamma \vdash XTRT\ !s\ AS\ v : A\ IN\ t : B^v_s}\ !E$$

$$\frac{\Delta; \Gamma \vdash s :: A}{\Delta; \Gamma; \cdot \vdash ?s : (s :: A)}\ ?I \qquad \frac{\Delta; \Gamma \vdash ?s : (s :: A) \qquad \Delta, a : A; \cdot \vdash t :: B}{\Delta; \Gamma; \cdot \vdash ASSM\ ?s\ AS\ a : A\ IN\ t :: B^a_s}\ ?E$$

The axiom scheme $ValVar$ states that a valid assumption v for A is a proof term for A under no further conditions, hence it can be derived from valid assumptions only and it will remain valid under all accessible states. The schemes for $\supset$ are for implication from valid or discharged assumptions, i.e. from unconditional proof, as a proof derivable from $ValVar$. The introduction rule takes a pair satisfying the implication to establish the equivalence over all terms in the antecedent to a term in the consequent; the elimination extracts precisely such a pair from an instance of an implication. $TruVar$ states that whenever a is a true assumption for A, it can be taken as a proof term for A, dependent on its own verification. Dependent Proof Formation constructs a proof polynomial for a functional expressions from dependent proof, as a proof derivable from $TrueVar$: if t is a proof of B dependent on a true assumption on A, then there is a proof t of B dependent on a term s for A. Application is the corresponding function elimination rule: if t is a proof for B provided s is a proof for A and given $s : A$ is true, then the process of constructing t abstracting from s and then applied to s is a witness of B. These schema are here used to express the type of all functions for $b[a] \in B[A]$ for any $a \in A$. The scheme $!I$ internalises the metalevel of proofs by using unconditional proof: it says that if s is an unconditional proof of A, then $!s$ is a witness to the validity of A given by s. The corresponding elimination allows to discharge valid assumptions: to do so, one needs to prove that $s : A$ is true by some unconditional proof; this is then used to build some unconditional proof t of C by taking s as a valid assumption in A. Introduction and elimination schemes for $?s$ are used to internalise the meta-level of proofs with true assumptions by using dependent proof: the introduction rule says that if s is a proof of A derivable from valid assumptions Δ and dependent on true assumptions Γ, then $?s$ is a witness to the dependent validity of A. The corresponding elimination allows to use true assumptions (without discharging): to do so, one needs to prove that $s :: A$ is true by some dependent proof term $?s$; this is then used to build some dependent proof term t of C by taking s as a true assumption in A.

The meaning of hypothetical judgements is standardly given by substitution principles fro validity and truth with proof terms:

Theorem 1 (Substitution on terms)**.** *The following substitutions hold:*

1. *If* $\Delta; \cdot \vdash s : A$ *and* $\Delta, v : A, \Delta' \vdash t : B$*, then* $\Delta; \Gamma; \Delta' \vdash t^v_s : B^v_s$*.*
2. *If* $\Delta, a : A \vdash t :: B$ *and* $\Delta, \Delta' \vdash s : A$*, then* $\Delta; \Gamma, \Delta'; \cdot \vdash tB^a_s$*.*

Proof.

1. by induction on the first given derivation, where $s:A$ can be of the form
   (a) $s:A$ (base case): derived by $ValVar$; similarly so for $t:B$; immediate by subsitution v/t;
   (b) $\lambda v:A.s:A \supset B$: derived from $v:A \vdash t:B$ by $\supset I$; then $\Delta, s \cdot v:A \supset B, \Delta' \vdash u:C$ and $\Delta, \Gamma, \Delta' \vdash u:C^{s\cdot v}_{\lambda v A.s}$;
   (c) $!s:(s:A)$: derived from $\Delta;\cdot \vdash s:A$ by $!I$; then $\Delta, !s:(s:A), \Delta' \vdash t:B$ and $\Delta, \Gamma, \Delta' \vdash t:B^{!s}_{XTRT\ !s\ AS\ vA\ IN\ t}$.
2. by induction on the first given derivation, where $t::B$ can be of the form:
   (a) $t::B$ (base case): derived by $TruVar$ from $a:A$; then use $1.a$ above to infer $v:A$ and perform substitutions v/a in t;
   (b) $\langle s\rangle t:B[A]$: derived from $\Delta, a:A \vdash t::B$ by *Function Formation*; then by the second derivation obtain $\Delta;\Gamma \vdash !s:(s:A)$ and use *Function Application* to obtain: $\Delta;\Gamma;\Delta' \vdash (s)t.s:B^{a}_{!s}$;
   (c) $?t:(t::B)$: derived from $\Delta;\Gamma \vdash t:B$ by $?I$; the second derivation allows $\Delta;\Gamma,\Delta' \vdash !t:(t:B)$ by $!I$ and $\Delta,\Gamma,\Delta' \vdash t:B^{a}_{XTRT\ !t\ AS\ aB\ IN\ t}$.

□

3.1 Extending with Proof Equality

To obtain normalisation of derivations one needs to show equality of terms, both for the Unconditional and the Dependent Proof cases to reflect derivation identity in the object language. Standard identity rules hold for judgements with unconditional proof $\Delta;\cdot \vdash s \equiv t:A$ which reads "s and t are provably equivalent proof terms of the truth of A under valid assumptions Δ", obtained by the following schema:

$$\frac{\Delta;\cdot \vdash s:A \qquad \Delta;\cdot \vdash s \equiv t:A}{\Delta;\Gamma \vdash t:A}\ EqUnProof$$

The meaning is given by the standard β,η and $!\eta$ equivalences,[6] with the due conservative restriction that allows using valid assumptions only in the first premise.

$$\frac{\Delta;\cdot \vdash s:A}{\Delta;\Gamma \vdash s \equiv s:A}\ Reflexivity$$

$$\frac{\Delta; v:A \vdash t:B \qquad \Delta;\cdot \vdash s:A}{\Delta;\Gamma \vdash t^{v}_{s} \equiv (\lambda v:A.s).t:B}\ Eq\beta$$

[6]Similarly to what done in [9] for the □ operator.

$$\frac{\Delta;\cdot\vdash !s:(s:A) \qquad \Delta;v:A\vdash t:B}{\Delta;\Gamma\vdash t_s^v\equiv XTRT\ !s\ AS\ v:A\ IN\ t:B_s^v}\ Eq!\beta$$

$$\frac{\Delta;\cdot\vdash s:A\supset B}{\Delta;\Gamma\vdash(\lambda v:A.s)\cdot v\equiv s:A\supset B}\ Eq\eta$$

$$\frac{\Delta;\cdot\vdash !s:(s:A) \qquad \Delta,v:A\vdash s:A}{\Delta;\Gamma\vdash !s\equiv XTRT\ !s\ AS\ v:A\ IN\ s:A}\ Eq!\eta$$

In the following we shall define the meaning of the equality for judgements with dependent proof $\Delta;\Gamma\cdot\vdash s\equiv t::A$, which reads "$s$ and t are provably equal proof terms of the truth of A under valid assumptions Δ and dependent on true assumptions Γ", obtained by the following schema:

$$\frac{\Delta;\Gamma,\cdot\vdash s::A \qquad \Delta;\cdot\vdash s\equiv t:A}{\Delta;\Gamma,\cdot\vdash t::A}\ EqDepProof$$

Standardly, η-conversions for Unconditional Proof guarantee strong normalization when paired with β-reduction. To obtain the same for Dependent Proof, one needs explicit parameterisation by the current context of true assumptions of the proof term. There might be (equivalent) terms that satisfy different propositions once their contexts of true assumptions change; moreover, β normal forms are no longer preserved by the corresponding η expansions alone. One needs therefore to formulate a basic additional requirement on the η-expansions (besides the usual one on the free variables of the unconditional proof) which says that the context in which the new (dependent) term is taken as valid is normalizable, hence standard $\beta\eta$ normal forms hold for the new dependent proof by reducing it to its unconditional counterpart. Without this additional requirement, semantically distinct terms are equated and our intended polymorphism lost, as shown in the following reduction:

$$\Delta;\Gamma,a:A\vdash s:B\Rightarrow_\beta\Delta;\Gamma\vdash s_t^a\equiv(\lambda a:A.s)\cdot t:B\Rightarrow_\eta\Delta;\Gamma\vdash s:A\supset B$$

This reduction does no longer hold once the first term of the reduction is expressed as $\Delta;\Gamma,a:A\vdash s::B$, as the contractum for $B[A]$ presents a different kind of proof term. A counterpart in our language has $\Delta;\Gamma,v:A\vdash s:B$ as the first term, i.e. with the additional condition that the dependent term reduces to an unconditional one.

In the following we shall treat true assumptions and dependent proofs as pre-terms and valid assumptions and unconditional proofs as terms. A context Γ of True Assumptions is therefore called a *pre-context*. An empty (pre-)context is denoted as previously by '$\cdot$'. First let us formulate a substitution principle with identity for valid and true assumptions:

Theorem 2 (Substitution on terms and pre-terms with identity)**.** *The following substitutions hold:*

1. *Let* $\Delta;\cdot\vdash s:A$ *and* $\Delta;\cdot\vdash s\equiv s':A$. *If* $\Delta,v:A,\Delta'\vdash t:B_s^v$, *then* $\Delta,v':A,\Delta'\vdash t':B_{s'}^{v'}$ *and* $\Delta,\Gamma,\Delta'\vdash t_{s'}^{v'}:B_{s'}^{v'}$.

2. *Let* $\Delta;\Gamma \vdash s :: A$ *and* $\Delta,\Delta' \vdash s' : A$. *If* $\Delta, a : A, \Gamma \vdash t :: B$, *then* $\Delta, v : A, \Gamma \vdash tB^{a}_{s'}$ *and* $\Delta,\Delta' \vdash t' B^{v}_{s'}$.

Proof.

1. By induction on $s : A$, which can be of the form:
   (a) $s : A$ (base case): derived from $v : A$ by $ValVar$; perform appropriate substitutions on s'/v' in t of B, directly by equivalence of unconditional proof;
   (b) $\lambda v : A.s : A \supset B$: derived from $v : A \vdash s : B$ by $\supset I$; then $\Delta, s \cdot v : A \supset B, \Delta' \vdash t : C$ by $\supset E$; so $\Delta;\Gamma \vdash t^{s\cdot v}_{\lambda v A.s'} \equiv (\lambda v : A.s) \cdot t : B^{s\cdot v}_{\lambda v A.s}$ by $Eq\beta$; perform now substitutions on t with $(\lambda v : A.s) \cdot t$ as by Theorem 1, Part 1.*b*;
   (c) $!s : (s : A)$: derived from $\Delta;\cdot \vdash s : A$ by $!I$; then $\Delta, !s : (s : A), \Delta' \vdash B \mid t$ and $\Delta, \Gamma, \Delta' \vdash t^{v}_{!s} \equiv XTRT\ \lambda v : A.s\ AS\ v.s : A\ IN\ t : B^{v}_{!s}$ by $Eq!\beta$; perform now substitutions on t with $(\lambda v : A.s) \cdot t$ as by Theorem 1, Part 1.*c*;
2. By induction on $s :: A$, which can be of the form:
   (a) $s :: A$ (base case): derived by $TruVar$ from $a : A$; then use $s : A$ from the second derivation and perform substitutions v/s in t by $EqDepProof$;
   (b) $\langle s \rangle t : B[A]$: derived from $\Delta, a : A \vdash t :: B$ by *Function Formation*; then by the second derivation obtain $\Delta;\Gamma,\Delta' \vdash !s' : (s' : A)$; so $\Delta;\Gamma,\Delta' \vdash t^{v}_{s'} \equiv XTRT\ !s'\ AS\ s : A\ IN\ t : s' : A$ by $Eq!\beta$; perform now substitutions on t with $!s'$ in B;
   (c) $?s : (s :: A)$: derived from $\Delta;\Gamma \vdash s :: A$ by $?I$; the second derivation allows $\Delta;\Gamma,\Delta' \vdash !s : (s' : A)$ by $!I$ and $\Delta,\Gamma,\Delta' \vdash !s \equiv XTRT\ !s'\ AS\ v : A\ IN\ v : s' : A$ by $Eq!\eta$; perform now substitutions on t with $!s'$ in B.

□

Definition 6 (Context Equivalence)**.** *Identity of (pre-)terms inductively generates (pre-)context equivalence by the following inference rules (for* i *either* 1 *or* 2*):*

$$\frac{}{\cdot =_{\beta\eta} \cdot} \qquad \frac{\Gamma_1 =_{\beta\eta} \Gamma_2 \qquad \Gamma_1 \vdash A =_{\beta\eta} A' \qquad \Gamma_2 \vdash A =_{\beta\eta} A'}{\Gamma_1 \vdash s_i :: A =_{\beta\eta} \Gamma_2 \vdash s_i :: A'}$$

Under the conditions of equality for pre-terms, pre-contexts are sound for equivalent propositions and for equivalent terms; this is needed for the standard equivalence properties to extend to dependent proofs:

Lemma 1. *If* $\Gamma_1 =_{\beta\eta} \Gamma_2$ *and* $\Gamma_1 \vdash s :: A$, *then* $\Gamma_2 \vdash s :: A$.

Lemma 2. *If* $\Gamma \vdash s_1 =_{\beta\eta} s_2 :: A$*, then* $\Gamma \vdash s_1 :: A$ *and* $\Gamma \vdash s_2 :: A$*.*

Proof. Proofs are by induction on Γ_1, Γ_2 using Definition 6 and on A using Theorem 2. □

We can now define the meaning of terms depending from local assumptions by laying down the full set of axioms and inference schemes that regulate the two forms of judgement presented at the beginning of this section.

Definition 7 (Axiom and Inference Schemes for Identity with Unconditional and Dependent Proof)**.** *Identity for Unconditional and Dependent Proof is given by the following:*

Reflexivity

$$\frac{\Delta; \cdot \vdash s : A}{\Delta; \Gamma \vdash s \equiv s : A}\ EqReflUnProof \qquad \frac{\Delta; \Gamma \vdash s :: A}{\Delta; \Gamma \vdash s \equiv s :: A}\ EqReflDepProof$$

Symmetry. (With Γ *possibly empty and* $s : A$*).*

$$\frac{\Delta; \Gamma \vdash s \equiv t :: A}{\Delta; \Gamma \vdash t \equiv s :: A}\ EqSymm$$

Transitivity. (With Γ *possibly empty and* $s : A$*).*

$$\frac{\Delta; \Gamma \vdash s_1 \equiv s_2 :: A \qquad \Delta; \Gamma \vdash s_2 \equiv s_3 :: A}{\Delta; \Gamma \vdash s_1 \equiv s_3 :: A}\ EqTrans$$

$\supset$

$$\frac{\Delta; v : A \vdash s \equiv t : B}{\Delta; \Gamma \vdash \lambda v : A.s \equiv \lambda v : A.t : A \supset B}\ Eq\supset\ I$$

$$\frac{\Delta; \Gamma \vdash s_1 \equiv s_2 : A \supset B \qquad \Delta; \cdot \vdash t_1 \equiv t_2 : A}{\Delta; \Gamma \vdash s_1 \cdot t_1 \equiv s_2 \cdot t_2 : B}\ Eq\supset\ E$$

$\supset\ \beta\eta$

$$\frac{\Delta; \cdot \vdash s : A \qquad \Delta, v : A \vdash t : B}{\Delta; \Gamma \vdash s_t^v \equiv s \cdot t : B}\ Eq\beta$$

$$\frac{\Delta, \Gamma \vdash t : A \supset B \qquad v \notin \mathit{fv}(s)}{\Delta; \Gamma \vdash \lambda v : A.s \equiv s \cdot t : B}\ Eq\eta$$

DepProof

$$\frac{\Delta; a : A \vdash t_1 \equiv t_2 :: B}{\Delta; \cdot \vdash \langle s \rangle t_1 \equiv \langle s \rangle t_2 : B[A]}\ EqDepProof$$

$$\frac{\Delta;\Gamma \vdash \langle s_1\rangle t_1 \equiv \langle s_2\rangle t_2 : B[A] \qquad \Delta;\cdot \vdash s_1 \equiv s_2 : A}{\Delta;\Gamma \vdash (s_1)t_1.s_1 \equiv (s_2)t_2.s_2 : B}\ EqApplication$$

!

$$\frac{\Delta;\cdot \vdash s \equiv t : A}{\Delta;\Gamma \vdash !s \equiv !t : (s : A)}\ Eq!I_l \qquad \frac{\Delta;\cdot \vdash s \equiv t : A}{\Delta;\Gamma \vdash !s \equiv !t : (t : A)}\ Eq!I_r$$

$$\frac{\Delta;\cdot \vdash !s_1 \equiv !s_2 : (r : A) \qquad \Delta; v : A;\cdot \vdash t_1 \equiv t_2 : B}{\Delta;\Gamma \vdash XTRT\ s_1\ AS\ v : A\ IN\ t_1 \equiv XTRT\ s_2\ AS\ v : A\ IN\ t_2 : B^v_r}\ Eq!E$$

$!\beta\eta$

$$\frac{\Delta;\cdot \vdash !s : (s : A) \qquad \Delta; v : A \vdash t : B}{\Delta;\Gamma \vdash t^v_s \equiv XTRT\ !s\ AS\ v : A\ IN\ t : B^v_s}\ Eq!\beta$$

$$\frac{\Delta;\cdot \vdash !s : (s : A) \qquad \Delta, v : A \vdash s : A}{\Delta;\Gamma \vdash !s \equiv XTRT\ !s\ AS\ v : A\ IN\ s : A}\ Eq!\eta$$

?

$$\frac{\overset{\pi}{\Delta;\Gamma \vdash s \equiv t :: A}}{\Delta;\Gamma;\cdot \vdash ?s \equiv ?t : (s :: A)}\ Eq?I_l \qquad \frac{\overset{\pi}{\Delta;\Gamma \vdash s \equiv t :: A}}{\Delta;\Gamma;\cdot \vdash ?s \equiv ?t : (t :: A)}\ Eq?I_r$$

where π is a derivation according to the Theorem 2.

$$\frac{\Delta;\Gamma \vdash ?s_1 \equiv ?s_2 : (r :: A) \qquad \Delta';\cdot \vdash u : A \qquad \Delta, a : A;\Gamma \vdash t_1 \equiv t_2 : B}{\Delta;\Gamma;\Delta' \vdash ASSM\ s_1\ AS\ a : A\ IN\ t_1 \equiv ASSM\ s_2\ AS\ a : A\ IN\ t_2 :: B^a_r}\ Eq?E$$

$?\beta\eta$

$$\frac{\Delta;\Gamma \vdash ?s : (s :: A) \qquad \Delta';\cdot \vdash s : A \qquad \Delta, a : A;\Gamma \vdash t :: B}{\Delta;\Gamma;\Delta' \vdash t^a_s \equiv ASSM\ ?s\ AS\ a : A\ IN\ t :: B^a_s}\ Eq?\beta$$

$$\frac{\Delta;\Gamma \vdash ?s : (s :: A) \qquad \Delta';\cdot \vdash s : A \qquad s \notin \mathit{fv}(t)}{\Delta;\Gamma;\Delta';\cdot \vdash ?s \equiv ASSM\ ?s\ AS\ a : A\ IN\ s :: A}\ Eq?\eta$$

Completing this schema with appropriate $\beta\eta$ rules for dependent proof – via appropriate expansions and contractions to the given ones – will allow us to prove Normalization and Confluence, to characterize safely $ILP_{nd\Diamond}$ (and so ILP_{dep}) in view of standard ILP.

3.2 Structural Properties

Structural properties are validated in $ILP_{nd\Diamond}$ by the following results:

Lemma 3 (Properties)**.** *The system satisfies*

1. *(Exchange) If* $\Delta; u:A, v:B; \cdot \vdash s:C$ *then If* $\Delta; v:B, u:A; \cdot \vdash s:C$
2. *(Weakening) If* $\Delta; \cdot \vdash s:A$ *then* $\Delta, v:B \vdash s:A$
3. *(Weakening) If* $\Delta; \Gamma \vdash s::A$ *then* $\Delta, a:B, \Gamma \vdash s::A$
4. *(Contraction) If* $\Delta; u:A, v:A; \Delta', \cdot \vdash s:A$ *then* $\Delta; w:A; \Delta', \cdot \vdash s_w^{u,v} : A_w^{u,v}$ *for* w *fresh*
5. *(Contraction) If* $\Delta; \Gamma, a:A, b:A \vdash s::A$ *then* $\Delta; \Gamma, c:A \vdash s_c^{a,b} :: A_c^{a,b}$ *for* c *fresh.*

Proof. The proof for every item is by induction.

- *Exchange* for Unconditional Proof, by induction on C, starting by:

  1. $s:C$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Delta; u:A; v:B \vdash s:C}\,\text{ValVar}}{\Delta; u:A, \cdot \vdash \lambda v:B.t:B \supset C}\supset I \qquad \Delta; u:A; \cdot \vdash v:B}{\Delta, u:A, \cdot \vdash s \cdot v:C}\supset E}{\Delta; u:A; \cdot \vdash !r:(s \cdot v:C)}\,!I \qquad \Delta; v:B, u:A \vdash s:C}{\dfrac{\Delta; u:A; \cdot \vdash XTRT!r\ AS\ v:B\ IN\ (s \cdot v:C)}{\Delta; v:B, u:A \vdash s:C}\,\text{Exchange}}\,!E$$

  2. $C := (D \supset E)$: immediate with additional construction;
  3. $!s:(s:C)$: immediate with additional construction;

- *Weakening* for unconditional proof, with a standard proof by deriving the same formula with or without the additional unconditional proof. The simple construction is by induction on B of the form:

  1. $v:B$:

$$\dfrac{\dfrac{}{\Delta; \cdot \vdash s:A}\,\text{ValVar} \qquad \Delta'; \cdot \vdash t:B}{\Delta, v:B \vdash s:A}\,\text{Weak}$$

  2. $B \supset C$:

$$\dfrac{\dfrac{\Delta; v:B \vdash t:C}{\Delta, \Gamma \vdash \lambda v:B.t:B \supset C}\supset I \qquad \Delta; \cdot \vdash s:A}{\Delta, v:B \supset C \vdash s:A}\,\text{Weak}$$

  3. $!t:(t:B)$:

$$\dfrac{\dfrac{\dfrac{\Delta, \cdot \vdash t : B}{\Delta, \Gamma \vdash !t : (t : B)}\, !I \qquad \Delta; \cdot \vdash s : A}{\Delta; \cdot \vdash XTRT\ !t\ AS\ v : B\ IN\ s : A}\, !E}{\Delta, v : B, \vdash s : A}\, \text{Weak}$$

- *Contraction* for unconditional proof, with a standard proof by deriving the same formula with a copy of the additional unconditional proof in the antecedent. In the newly formulated derivation, the novel proof term is obtained by substitution of terms according to Theorem 2, part 1.
- *Weakening* for dependent term, on B:
  1. $t :: B$:

$$\dfrac{\dfrac{}{\Delta, a : B; \cdot \vdash t :: B}\, \text{TruVar} \qquad \Delta; \Gamma \vdash s :: A}{\Delta; a : B, \Gamma \vdash s :: A}\, \text{Weak}$$

  2. $\langle t \rangle v : B[C]$:

$$\dfrac{\dfrac{\Delta, a : B; \cdot \vdash t : C}{\Delta; \cdot \vdash \langle t \rangle v : B[C]}\, DepEv\ Formation \qquad \Delta; \Gamma \vdash s :: A}{\Delta, a : B[C], \Gamma \vdash s :: A}\, \text{Weak}$$

  3. $?t : (t :: B)$:

$$\dfrac{\dfrac{\Delta; \Gamma \vdash t :: B}{\Delta, \Gamma; \cdot \vdash ?t : (t :: B)}\, ?I \qquad \Delta; \cdot \vdash s :: A}{\Delta, a : t :: B, \Gamma \vdash s :: A}\, \text{Weak}$$

- *Contraction* for unconditional proof: entirely similar to the cases above, with the additional use of substitutions on terms as by Theorem 2, part 2.

□

The following easy construction shows why Exchange cannot be satisfied for dependent terms:

$$\dfrac{\dfrac{\Delta, a : A, b : B \vdash t :: B}{\Delta; a : A \vdash \langle b \rangle t : C[B]}}{\Delta, \cdot \vdash \langle a \rangle (b) t : C[B(A)]}$$

which is obtained by using abstraction and dependent proof formation; allowing exchange on true assumptions induces mixed constructions of dependent terms.

4 Transformations

We now define transformation of derivations by contractions and expansions on connectives, adding appropriate operations for dependent proof terms.

Contraction for $\supset$

$$\dfrac{\dfrac{\overline{\Delta; v\!:\!A; \cdot \vdash s\!:\!B}}{\Delta; \Gamma \vdash \lambda v\!:\!A.s\!:\!A \supset B}\supset I \qquad \Delta; \cdot \vdash t\!:\!A}{\Delta; \Gamma \vdash (\lambda v\!:\!A.s) \cdot t\!:\!B}\supset E$$

contracts to

$$\dfrac{\dfrac{\pi}{\Delta; \Gamma \vdash s_t^a\!:\!B} \qquad \dfrac{\Delta; v\!:\!A \vdash s\!:\!B \qquad \Delta, \cdot \vdash t\!:\!A}{\Delta, \Gamma \vdash s_t^v \equiv (\lambda v\!:\!A.s) \cdot t\!:\!B}\,Eq\beta}{\Delta; \Gamma \vdash (\lambda v\!:\!A.s) \cdot t\!:\!B}\,\text{EqUnProof}$$

where π is a derivation according to Theorem 2, point 1.

Contraction for !

$$\dfrac{\dfrac{\Delta; \cdot \vdash s\!:\!A}{\Delta; \Gamma \vdash !s\!:\!(s\!:\!A)}\,!I \qquad \Delta; v\!:\!A; \cdot \vdash t\!:\!B}{\Delta; \Gamma \vdash XTRT\ s\ AS\ v\!:\!A\ IN\ t\!:\!B_t^v}\,!E$$

contracts to

$$\dfrac{\dfrac{\pi}{\Delta; \Gamma \vdash t_s^v\!:\!B_s^v} \qquad \dfrac{\Delta; \cdot \vdash s\!:\!A \qquad \Delta, v\!:\!A; \cdot \vdash t\!:\!B}{\Delta, \Gamma \vdash t_s^v \equiv XTRT\ t\ AS\ v\!:\!A\ IN\ t\!:\!B_s^v}\,Eq!\beta}{\Delta; \Gamma \vdash XTRT\ !s\ AS\ v\!:\!A\ IN\ t\!:\!B_s^v}\,\text{EqUnProof}$$

where π is a derivation according to Theorem 2, point 1.

Contraction for ?

$$\dfrac{\dfrac{\Delta; \Gamma \vdash s :: A}{\Delta; \Gamma, \cdot \vdash ?s\!:\!(s :: A)}\,?I \qquad \Delta, a\!:\!A \vdash t :: B}{\Delta; \Gamma, \cdot \vdash ASSM\ ?s\ AS\ a\!:\!A\ IN\ t :: B_s^a}\,?E$$

contracts to

$$\dfrac{\dfrac{\pi}{\Delta; \Gamma \vdash t_s^a\!:\!B_s^a} \qquad \dfrac{\Delta; \Gamma \vdash s :: A \qquad \Delta, a\!:\!A; \Gamma; \cdot \vdash t :: B}{\Delta, \Gamma; \cdot \vdash t_s^a \equiv ASSM\ ?s\ AS\ a\!:\!A\ IN\ t :: B_s^a}\,Eq?\beta}{\Delta; \Gamma, \cdot \vdash ASSM\ ?s\ AS\ a\!:\!A\ IN\ t :: B_s^a}\,\text{EqDepProof}$$

where π is a derivation according to Theorem 2, point 2, which in turn guarantees that the contraction terminates if the pre-term s can be equated to a term t.

Contraction for DepProof

$$\dfrac{\dfrac{\Delta; a:A \vdash t :: B}{\Delta; \cdot \vdash \langle s \rangle t : B[A]}\ \mathit{DepProof\ Formation} \qquad \Delta; \cdot \vdash s : A}{\Delta; \Gamma \vdash (s)t.s : B}\ \mathit{Appl}$$

contracts to

$$\dfrac{\dfrac{\Delta; \Gamma; \cdot \vdash ?s : (s :: A) \qquad \Delta; a:A \vdash t :: B}{\Delta; \Gamma; \cdot \vdash \mathit{ASSM}\ ?s\ \mathit{AS}\ a:A\ \mathit{IN}\ t :: B^a_s \equiv \langle s \rangle t : B[A]}\ \text{EqDepProof} \qquad \Delta; \cdot \vdash s : A}{\Delta; \Gamma \vdash (s)t.s : B}\ \text{Application}$$

where the rule $\mathit{EqDepProof}$ takes in one step a $?E$ rule and a $\mathit{DepProof\ Formation}$ rule to equate their proof terms.

Expansion for $\supset$

$$\Delta; \Gamma \vdash s : A \supset B$$

expands to

$$\dfrac{\dfrac{\dfrac{\Delta; \cdot \vdash s : A \supset B \qquad \Delta; \cdot \vdash v : A}{\Delta; v:A; \Gamma \vdash s \cdot v : B}\ \supset E}{\Delta; \Gamma \vdash \lambda v : A.(s \cdot v) : A \supset B}\ \supset I \qquad \dfrac{\Delta; \Gamma \vdash s : A \supset B \qquad v \notin \mathtt{fv}(t)}{\Delta; \Gamma \vdash \lambda v : A.(s \cdot v) \equiv s : A \supset B}\ \text{Eq}\supset\text{I}}{\Delta; \Gamma \vdash s : A \supset B}\ \mathit{EqUnProof}$$

Expansion for !

$$\Delta; \Gamma \vdash !s : (s : A)$$

expands to

$$\dfrac{\dfrac{\Delta; \cdot \vdash t : A \qquad \dfrac{\Delta; \cdot \vdash s : A}{\Delta; \Gamma \vdash !s : (s : A)}\ !I}{\Delta; \Gamma \vdash \mathit{XTRT}\ !s\ \mathit{AS}\ v:A\ \mathit{IN}\ t : A^v_s}\ !E \qquad \dfrac{\Delta; \cdot \vdash !t : (s : A) \qquad v \notin \mathtt{fv}(t)}{\Delta; \Gamma \vdash !(\mathit{XTRT}\ !t\ \mathit{AS}\ v:A\ \mathit{IN}\ !v \equiv s) : s : A}\ \text{Eq}!\eta}{\Delta; \Gamma \vdash !s : (s : A)}\ \mathit{EqUnProof}$$

Expansion for ?

$$\Delta; \Gamma; \cdot \vdash ?t : (s :: A)$$

expands to

$$\dfrac{\dfrac{\Delta; a:A; \cdot \vdash t :: A \qquad \dfrac{\Delta; a:A; \cdot \vdash s :: A}{\Delta; \Gamma \vdash ?s : (s :: A)}\ ?I}{\Delta; \Gamma \vdash \mathit{ASSM}\ ?s\ \mathit{AS}\ a:A\ \mathit{IN}\ t :: A^a_s}\ ?E \qquad \dfrac{\Pi \qquad \Sigma \qquad T}{\Delta; \Gamma; \Delta'; \cdot \vdash ?(r \equiv t) : (s :: A)}\ \text{Eq}?\eta}{\Delta; \Gamma; \cdot \vdash ?t : (s :: A)}\ \mathit{EqDepProof}$$

where $\Pi = \Delta;\Gamma \vdash ?t : (s :: A)$; $\Sigma = \Delta';\cdot \vdash u : A$; $T = u \notin \mathtt{fv}(t)$; $r = ASSM\ ?t\ AS\ a{:}A\ IN\ s{::}A^a_s$.

Expansion for DepProof

$$\Delta;\cdot \vdash \langle s\rangle t : B[A]$$

expands to

$$\dfrac{\dfrac{\dfrac{\dfrac{\Delta;\Gamma \vdash \langle s\rangle t : B[A] \qquad \Delta;\Gamma \vdash s : A}{\Delta;\Gamma \vdash (s)t.s : B}\ \text{Application} \qquad \Delta; v : A \vdash t : B}{\Delta;\Gamma \vdash XTRT\ s\ AS\ v{:}A\ IN\ t : B^s_{(s)t.s}}\ !\text{E}}{\Delta;\Gamma;\cdot \vdash t^v_s \equiv (s)t.s : B}\ \text{Eq}\beta \qquad \Delta;\Gamma \vdash s :: A}{\Delta;\cdot \vdash \langle s\rangle t : B[A]}\ \textit{DepEv Formation}$$

5 Normalisation

To show that (Weak and Strong) Normalisation holds for this system, we need a detour imposed by the newly added dependent terms. For these terms, $\beta\eta$ equivalent redexes might not be equivalent, as this two diagrams show:

$$\begin{array}{lll} \Delta;\cdot \vdash \langle s\rangle t : B[A] & \Rightarrow_{Eq\beta} & \Delta;\Gamma \vdash s^v_t \equiv (s)t.s : B \\ \Downarrow_{Eq\eta} & & \Uparrow_{EqUnProof} \\ \Delta;\Gamma \vdash s \cdot t : A \supset B & \Rightarrow_{Eq!\beta} & \Delta;\Gamma \vdash t^v_s \equiv XTRT\ !s\ AS\ v{:}A\ IN\ t : B^v_s \end{array}$$

$$\begin{array}{lll} \Delta;\cdot \vdash \langle s\rangle t : B[A] & \Rightarrow_{Eq\beta} & \Delta;\Gamma \vdash s^v_t \equiv (s)t.s : B \\ \Downarrow_{Eq\eta} & & \\ \Delta;\Gamma \vdash s \cdot t : A \supset B & \Rightarrow_{Eq?\beta} & \Delta;\Gamma;\cdot \vdash t^a_s \equiv ASSM\ ?s\ AS\ a{:}A\ IN\ t{::}B^a_s \end{array}$$

The first diagram commutes, as the antecedent in the implication is converted in a valid assumption, hence the two terms on the right-hand side are equivalent. In the second diagram, the transformation by $Eq?\beta$ abstracts from the valid assumption in A to a locally valid one which becomes strictly dependent on Γ and it requires explicit subsititution to be shown to reduce to its valid counterpart $v : A$; hence the diagram does not commute. To show that

the extension by the notion of dependent proof term is preserving with respect to normalisation, one needs to prove that expressions with dependent proof are valid only by reduction to a normal form that corresponds to an unconditional proof for the same proposition.[7]

5.1 Reduction of Normal Forms

A standard η-expansion $\Gamma \vdash t \Rightarrow \lambda x\!:\!A.t.x$ for a variable x which is not among the free ones in t (which shall be indicated as $x \notin \mathtt{FV}(t)$), holds if A is neither a λ-abstraction itself, nor is applied to another term. For dependent terms, as in dependent type theories, these restrictions on the applicability of expansions further require that the term A is a long $\beta\eta$-normal form, so that the crucial procedure to obtain term normalization is to proceed on evaluation on terms, two terms or types being equal if and only if their normal forms are identical.[8] The solution can be adapted also for our notion of dependent proof term, following the idea of explicit substitution that was already required as additional premise in the rules of identity.[9]

We shall consider a Reduction (or Rewrite) relation $\rightarrow_R$; with $\rightarrow_R^+$ we indicate its transitive closure; with $\rightarrow_R^*$ its reflexive and transitive closure; with $\rightarrow_R^=$ its reflexive, transitive and symmetric closure. A term t is said to be in *normal form* if there is no term t' such that $t \rightarrow_R t'$. A relation is *strongly normalising* if there are no infinite reduction sequences $t \rightarrow_R t' \rightarrow_R t'' \ldots$. A relation is *weakly normalising* if for every term t there is a normal form term t' such $t \rightarrow_R t'$. Moreover, $t \rightarrow_{\beta\eta} t'$ means that t' represents the $\beta\eta$-long normal form of t.

Our strategy to obtain normalisation for the system $ILP_{nd\Diamond}$ is inspired by a similar procedure for a system with dependent types, pursued in [16]. It consists of two steps. First, define two normal form predicates, called respectively *internal normal form* (INF) and *full normal form* (FNF) such that the former is used in defining the latter when dependent proofs occur:

Definition 8 (Predicates INF and FNF)**.** *The normal form predicates INF and FNF are defined according to the following schemas:*

$$\frac{\Delta;\cdot \vdash s\!:\!A}{\Delta;\Gamma \vdash FNF(s)} \qquad \frac{\Gamma \vdash s::A}{\Gamma;\cdot \vdash INF(s)}$$

$$\frac{\Delta;\cdot \vdash FNF(A) \qquad \Delta;a\!:\!A \vdash FNF(t)}{\Delta;\Gamma;\cdot \vdash FNF([a/v]\cdot t)}$$

$$\frac{\Gamma;\cdot \vdash INF(A) \qquad \Delta;a\!:\!A \vdash FNF(t)}{\Delta;\Gamma;\cdot \vdash INF(t[a\!:\!A])}$$

[7] In the system with simple proofs presented in [9], weak normalisation is obtained by induction on the length of derivations with contraction, whereas strong normalisation is obtained by translation of the Intensional Lambda Calculus $\lambda^{\mathbf{I}}$ to an Abstract Reduction System composed by the set of derivations and a rewriting system composed by the corresponding contractions.

[8] See e.g. [1].

[9] This recalls a standard proviso on reductions for system with for constructive modalities, see for example [15, 13, 21, 24].

Second, define the standard reduction relation $\rightarrow_R$ including one rewrite relation for pure η-expansions that involve only FNFs ($\rightarrow_{\eta FNF}$) and by establishing $\rightarrow_{\eta INF}$ as a subrelation of $\rightarrow_{\eta FNF}$, where the former does not include top-level expansions in order to avoid infinite reductions:

Definition 9 (Rewriting Rules for INF/FNF predicates)**.** *The η-expansion rewriting rules for INF/FNF predicates are defined according to the following rules:*

$$\frac{\Delta;\Gamma \vdash s \cdot t : A \supset B \qquad \Delta;\Gamma \vdash FNF(s \cdot t)}{\Delta;\Gamma \vdash s \rightarrow_{\eta FNF} v : A.s}$$

$$\frac{\Delta;\cdot \vdash t \rightarrow_{\eta FNF} t'}{\Delta;\Gamma \vdash t \rightarrow_{\eta INF} t'} \qquad \frac{\Delta; v : A \vdash t \rightarrow_{\eta FNF} t'}{\Delta;\Gamma \vdash t[a : A] \rightarrow_{\eta INF} t'[a : A]}$$

$$\frac{\Delta;\Gamma \vdash A \rightarrow_{\eta INF} A'}{\Delta;\Gamma \vdash B[A] \rightarrow_{\eta INF} B[A']} \qquad \frac{\Delta;\cdot \vdash A \rightarrow_{\eta FNF} A'}{\Delta;\Gamma \vdash A[B] \rightarrow_{\eta INF} A'[B]}$$

$$\frac{\Delta;\cdot \vdash s : A \qquad \Delta; v : A \vdash t : B \rightarrow_{\eta FNF} t' : B'}{\Delta;\Gamma \vdash XTRT\ s\ AS\ v : A\ IN\ t : B^v_t \rightarrow_{\eta FNF} XTRT\ s\ AS\ v : A\ IN\ t' : B^v_t}$$

$$\frac{\Delta;\Gamma \vdash s :: A \qquad \Delta; a : A;\Gamma \vdash t : B \rightarrow_{\eta FNF} t' : B'}{\Delta;\Gamma \vdash ASSM\ s\ AS\ a : A\ IN\ t : B^a_t \rightarrow_{\eta INF} ASSM\ s\ AS\ a : A\ IN\ t' : B^a_t}$$

Rewriting corresponds therefore to either η-expansions on INF/FNF or β-reductions:

Definition 10 (Reductions)**.** *Let $\rightarrow_\beta$ be standard β-reduction on terms, then*

1. $\Delta;\Gamma;\cdot \vdash t \rightarrow_{INF} t'$ *iff* $\Delta;\Gamma;\cdot \vdash t \rightarrow_\beta t'$ *or* $\Delta;\Gamma;\cdot \vdash t \rightarrow_{\eta INF} t'$*;*
2. $\Delta;\cdot \vdash t \rightarrow_{FNF} t'$ *iff* $\Delta;\Gamma \vdash t \rightarrow_\beta t'$ *or* $\Delta;\Gamma \vdash t \rightarrow_{\eta FNF} t'$*;*

Definition 11 (Equality for $\beta\eta$-reducts with Dependent Proof)**.** *The closure of $\beta\eta$-normal forms for identity of dependent proofs requires the following rule:*

$$\frac{\Delta;\cdot \vdash A \rightarrow_{\beta\eta} A' \qquad \Delta; a : A \vdash B \equiv B'}{\Delta;\Gamma \vdash a :: B \equiv a' :: B'}\ \beta\eta Eq\ DepProof$$

Lemma 4 (Equivalence of $\beta\eta$NF under $\rightarrow_{INF/FNF}$)**.** *Let $\Delta;\Gamma \vdash t \rightarrow_{INF/FNF} t'$, then there is a judgement $\Delta;\Gamma \vdash t \rightarrow_{\beta\eta} t'$. If $\Delta;\Gamma \rightarrow_{\beta\eta} \Delta';\Gamma'$, then $\Delta';\Gamma' \vdash t \rightarrow_{INF/FNF} t'$. Finally, let t be in βNF and $\Delta;\Gamma \vdash t \rightarrow_{\eta FNF} t'$, then t' is in βNF.*

Proof. The first implication is satisfied by using Definition 8 to show that every INF is internal to a FNF and by Definitions 9 and 10 for the equivalence to a corresponding $\beta\eta$ reduction. For the second implication, the construction will be done similarly by induction on Δ and using Definition 11 for Γ. ηFNF just preserves β Normal Forms. □

Lemma 5 (β Normal Forms)**.** *If* $\Delta;\Gamma \vdash t \equiv t'$*, then* $\Delta;\Gamma \vdash t \rightarrow_{\beta\eta} t'$*, for every term* t*.*

Proof. Immediate by Lemma 4. □

Lemma 6 (Normalisation)**.** *If* $\Delta;\Gamma \vdash FNF(t)$*, then there is no* t' *such that* $\Delta;\Gamma \vdash t \rightarrow_{\beta\eta} t'$

Proof. By induction on the derivation. By using Definition 8, one shows that schemas of INF are included into FNF; by Definition 9, one shows that every INF normal form can be η-reduced to a FNF normal form; applying Lemma 4, every η equivalent NF has a unique β-$FNF(t)$. Then by Lemma 5, there is a finite number of steps leading to an equivalent $\beta\eta$ term which must be in normal form. □

Dependent proof is now defined by an *internal normal form* which occurs in the definition of a *full normal form*; normalisation obtains by reduction to the latter. At this stage the rewriting relation induced by the FNF/INF predicates is only weakly normalising. In order to show that it induces also strong normalisation, one needs to prove that every term can be reduced to full normal form, i.e. the reverse of Lemma 6. The interesting part concerns the judgement

$$\Delta;\Gamma;\cdot \vdash t^a_s \equiv ASSM\ ?s\ AS\ a{:}A\ IN\ t{::}B^a_s$$

for which one needs to show that there is another judgement

$$\Delta;\Gamma \vdash t^v_s \equiv XTRT\ !s\ AS\ v{:}A\ IN\ t{:}B^v_s$$

and $FNF(A \supset B)$.

6 (Strong) Normalisation and Confluence

We now want to show that every expression (including those with dependent terms) reduces to a unique confluent η-expanded full normal form. To do so, we shall prove that terms in (either) normal form derivable under equivalent assumptions, are equal; and that $\beta\eta$ equivalent terms in (either) normal form are equal. In this way, one can prove a confluence relation up to equivalence of terms that can be different up to $\beta\eta$-equivalence.

We start by defining a notion of Type Equivalence which applies to both terms and pre-terms (i.e. to both terms with INF and FNF):

Definition 12 (Type Equivalence)**.** *Two (pre-)terms are said to be type equivalent* $\Delta;\Gamma \vdash b \approx b'$ *if their (pre-)contexts respect context equivalence (by Lemmas 1 and 2) and their* $\beta\eta$*-normal forms are equivalent (by Lemma 6).*

This definition is given for terms by the following inference rules:

$$\frac{}{\Delta;\cdot \vdash \cdot \approx \cdot}$$

$$\frac{\Delta \vdash s : A}{\Delta; \cdot \vdash s \approx s}$$

$$\frac{\Delta; \cdot \vdash s : A =_{\beta\eta} s' : A' \qquad \Delta; v : A \vdash B \approx B'}{\Delta; \Gamma \vdash \lambda v : A.s : A \supset B \approx \lambda v : A'.s' : A' \supset B'}$$

$$\frac{\Delta; \cdot \vdash s : A =_{\beta\eta} s' : A'}{\Delta; \Gamma \vdash !s : s : A \approx !s' : (s' : A)}$$

Type equivalence requires preservation by substitution of True Assumptions by Valid Assumptions:

Lemma 7. *Let* $\Delta; a : A; \cdot \vdash t :: B$ *and* $\Delta; \cdot \vdash s : A$. *If* $\Delta; \Gamma \vdash t \approx t'$ *and* $\Delta; \Gamma \vdash s \equiv s'$ *then* $\Delta; \Gamma[a/s] \vdash t[a/s] \approx t'[a/s']$.

Proof. The proof goes by induction on $t :: B$, using substitution on terms as by Theorem 2. □

Now we can extend the previous set of rules to include pre-terms:

$$\frac{}{\Delta; \Gamma \vdash \cdot \approx \cdot}$$

$$\frac{\Gamma \vdash s :: A \qquad \Delta \vdash s' : A}{\Delta; \Gamma \vdash s \approx s'}$$

$$\frac{\Delta; \cdot \vdash s : A =_{\beta\eta} s' : A' \qquad \Delta; a : A \vdash B \approx B'}{\Delta; \Gamma \vdash \langle s \rangle t : B[A] \approx \langle s' \rangle t' : B'[A']}$$

$$\frac{\Delta; \Gamma \vdash s :: A \qquad \Delta; \cdot \vdash s : A =_{\beta\eta} s' : A'}{\Delta; \Gamma \vdash ?s : (s :: A) \approx ?s' : (s' :: A')}$$

From this it follows immediately that $\approx$ is an equivalence relation and hence a sub-relation of a β-reduction, hence it is strongly normalising. We can prove now that our rewriting relation is coherent with the type equivalence relation.

Lemma 8. *The rewriting relation* $\rightarrow_{\beta FNF}$ *is coherent with type equivalence: if* $\Delta; \Gamma \vdash t \rightarrow_{\beta FNF} t'$ *and* $\Delta; \Gamma \vdash t \approx u$, *then* $\Delta; \Gamma \vdash u \rightarrow_{\beta FNF} t'$.

Proof. The proof is by induction on the rewrite relation $\rightarrow_{\beta FNF}$, with one interesting case: if $\Delta; \Gamma \vdash t \rightarrow_{\beta FNF} t[a : A]$ when $\Delta; \cdot \vdash s : A$ and $a \notin \mathtt{FV}(t)$. Then a reduction is required $\Delta; \Gamma \vdash t[a : A] \rightarrow_{\beta FNF} ([a/v] \cdot t)$ to obtain $\Delta; v : A', \Gamma \vdash t \rightarrow_{\beta FNF} t'$. Then u must be of the form $t''[a' : A']$ and $\Delta; \Gamma \vdash t' \approx t''$ and $\Delta; a : A, \Gamma \vdash t \approx t'$. Hence, there must be a reduction $\Delta; \Gamma \vdash t''[a' : A'] \rightarrow_\beta ([a'/v'] \cdot t'')$. Then by Lemma 7, $t''[a' : A'] \equiv ([a'/v'] \cdot t'')$ so it is a reduction holding under type equivalence, so by substitution $([a/v] \cdot t) \approx ([a'/v'] \cdot t'')$ and as required $\Delta; \Gamma \vdash t''[a' : A'] \rightarrow_{\beta FNF} ([a/v] \cdot t)$. □

It follows that $\rightarrow_{\beta FNF}$ is strongly normalising. By extending with INF-normal forms, we prove confluence:

Lemma 9 (Confluence of β-expanded expressions)**.** *If* $\Delta;\Gamma \vdash t \to^*_{\beta FNF/INF} t'$ *then there is* $\Delta;\Gamma \vdash t \to^*_{\beta FNF/INF} t''$ *such that* $\Delta;\Gamma \vdash t' \approx t''$.

Proof. The proof is on the length of the reduction sequence for $\Delta;\Gamma \vdash t \to^*_{\beta FNF/INF} t'$:

1. length 1: then either $\Delta;\Gamma \vdash t \approx t'$ and $t'' = t$ or $\Delta;\Gamma \vdash t \not\approx t'$ and $t'' = t'$.
2. inductive step: if $\Delta;\Gamma \vdash t' \to_{\beta FNF/INF} u$, then by Lemma 8, there is u' such that $\Delta;\Gamma \vdash t' \to^*_{\beta FNF/INF} u'$ and $u' \approx u''$; and if $\Delta;\Gamma \vdash t \to^*_{\beta FNF/INF} t''$ such that $t' \approx t''$, then $\Delta;\Gamma \vdash t'' \to^*_{\beta FNF/INF} t'''$ such that $u'' \approx t'''$.

□

We now want to prove that also the η-expanded forms are confluent.

Lemma 10. *The rewriting relation* $\to_{\eta FNF}$ *is coherent with type equivalence: if* $\Delta;\Gamma \vdash t \to_{\eta FNF} t'$ *and* $\Delta;\Gamma \vdash t \approx u$*, then* $\Delta;\Gamma \vdash u \to_{\eta FNF} t'$.

Proof. The proof is by induction on the rewrite relation given by the first derivation. We consider the two interesting cases:

1. if $\Delta;\Gamma \vdash t \to_{\eta FNF} v : A.t$ when $\Delta;\cdot \vdash t : A$ and $v \notin \mathtt{FV}(t)$. Then t is not context dependent, neither is u. If $\Delta;\Gamma \vdash t \approx u$ then $\Delta;\Gamma \vdash t \to_{\beta\eta} u$ by Definition 10 and by Definition 12 there is $\Delta;\cdot \vdash u : A$ such that $\Delta;\Gamma \vdash t \to_{\beta\eta} u$ holds. This also means there is an expansion $\Delta;\Gamma \vdash u \to_{\eta FNF} v : A.t$ when $\Delta;\cdot \vdash A \mid u$ and $v \notin \mathtt{FV}(t)$. So by substitution and the hypothesis, $\Delta;\Gamma \vdash u \to_{\beta FNF} t'$.
2. if $\Delta;\Gamma \vdash t \to_{\eta INF} t[a : A]$ when $\Delta;\cdot \vdash tA$ and $a \notin \mathtt{FV}(t)$. By η-reductions on $\Delta;\Gamma \vdash INF(A)$, $\Delta; a : A \vdash FNF(t')$ and $\Delta;\cdot A \to_{\eta FNF} A'$, obtain $\Delta;\Gamma \vdash FNF([a/v]\cdot t)$ and use the latter to obtain $\Delta; v : A', \Gamma \vdash t \to_{\eta FNF} t'$. Then proceed as in the previous case to show coherence for $t \approx u$.

□

Lemma 11 (Confluence of η-expanded expressions)**.** *If* $\Delta;\Gamma \vdash t \to_{\eta FNF/INF} t'$ *then there is* $\Delta;\Gamma \vdash t \to_{\eta FNF/INF} t''$ *such that* $\Delta;\Gamma \vdash t' \approx t''$.

Proof. The proof goes by induction on the structure of terms. As above, there are two interesting cases:

1. If $\Delta;\Gamma \vdash t \to_{\eta FNF} v : A.s$ and $\Delta;\Gamma \vdash t \to_{\eta FNF} v : A'.s'$, then $\Delta;\Gamma \vdash A =_{\beta\eta} A'$ and $\Delta;\Gamma \vdash v : A.s \approx a : A'.s'$;
2. if $\Delta;\Gamma \vdash t \to_{\eta FNF} v : A.s$ and $\Delta;\Gamma \vdash t \to_{\eta INF} t'$, then t' must have a redex which is context independent and $\Delta;\Gamma \vdash t \approx t'$. Hence, there must be rewriting relations $\Delta;\Gamma \vdash t' \to_{\eta FNF} v : A.s$ and $\Delta;\Gamma \vdash v : A'.s \to_{\eta INF} v : A.t'$.

□

Lemma 12 (Confluence)**.** *If* $\Delta;\Gamma \vdash s \to_{\beta FNF} t$ *and* $\Delta;\Gamma \vdash s \to_{\eta INF} t'$*, then either*

1. $\Delta;\Gamma \vdash t' \to^{+}_{\beta FNF} t'' \approx u$ *and* $\Delta;\Gamma \vdash t \to^{*}_{\eta INF} u$*; or*
2. $\Delta;\Gamma \vdash t' \to^{+}_{\beta FNF} t'' \approx u$ *and* $\Delta;\Gamma \vdash t \to_{\eta FNF} u$*.*

If $\Delta;\Gamma \vdash s \to_{\eta FNF} t'$ *in the second premise, then* $\Delta;\Gamma \vdash t \to^{*}_{\eta FNF} u$*.*

Proof. One needs to show that the transitively closed β-reduction $t' \to^{+}_{\beta FNF} t''$ is preserved under type-equivalence. Proof is by induction on t, with the usual two relevant cases:

1. If $\Delta;\Gamma \vdash t \to_{\eta FNF} t[a:A]$, then reduction is performed as for Lemma 10. If $t' := t'[a':A']$, then by Definitions 6, 11 it holds $\Delta;\Gamma \vdash A =_{\beta\eta} A'$, so $\Delta;\Gamma \vdash t[a:A] \to_{\beta} t'[a':A'] \to_{\beta} t \approx t'$.
2. If $\Delta;\Gamma \vdash ([a/x]:A)t \to_{\beta} v:A.t$, then any rewrite on A terminates, and so does any $\to_{\eta INF}$ from a or t, which makes them both preserving type-equivalence.

□

All the previous results taken together show that our reduction relation $\to_{FNF/INF}$ for both β and η conversions is confluent under type equivalence. This is all that is needed to prove uniqueness of $\to_{FNF/INF}$ normal forms.

Lemma 13 (Uniqueness of FNF/INF normal forms)**.** *For* $\Delta;\Gamma \vdash t \approx t'$ *such that there are* $s \to_{\beta\eta INF} t$ *and* $s' \to_{\beta\eta INF} t'$*, then* $t = t'$*. For* $\Delta;\Gamma \vdash t =_{\beta\eta} t'$ *such that there are* $s \to_{\beta\eta FNF} t$ *and*$s' \to_{\beta\eta FNF} t'$*, then* $t = t'$*.*

Proof. By induction on t, t', with the interesting cases:

1. $t := [a/x]:A.t$, then $t' := [a'/x']:A'.t'$ and $\Delta;\Gamma \vdash A =_{\beta\eta} A'$ and $\Delta; a:A;\Gamma \vdash t \approx t'$. Since t, t' are INF-redexes, by induction $t = t'$.
2. As t, t' are FNF-redexes and $\beta\eta$-equivalent, they must be type equivalent. Then from them infer the corresponding INF-redexes and apply the argument as by the previous point.

□

It remains to prove strong normalisation for $\to_{INF/FNF}$, i.e. that each termination chain for such predicate has no infinite series of terms. As for theories of types with dependencies, the crucial step is to show calculation of variables to their reducts (see [16], [1]) to extend recursively a standard subject reduction lemma for terms by one reduction step for each occurrence of a dependent term.

Definition 13. *The set of typable terms* $\mathcal{T}_z(S)$ *for each typing judgment* $\Delta;\Gamma \vdash S$*, with* S *either a with variable* $z(s) \notin dom(\Gamma)$ *is given by:*

1. $\mathcal{T}_z(B[A]) = \{z\} \cup \{[x/v:A] \cdot [s/t:B]$ *such that* $v \in \mathcal{T}_x(A)$ *and* $t \in \mathcal{T}_s(B)$

2. $\mathcal{T}_z(S) = \{z\}$ *iff for no* $A \subseteq S$, $A := B[C]$.

with variables v, s fresh.

Lemma 14 (Subject reduction for terms)**.** *Let* $\Delta; \cdot \vdash FNF(S)$ *and* $s : S$*, then* $\Delta; a : S \vdash a \to_{\eta FNF} t$ *iff* $t \in \mathcal{T}_x(S)$.

Proof. This is established by induction on S.

$\Rightarrow$ The set of typable terms in S is closed under $\to_{\eta FNF}$ reduction: if for no $A \subseteq S$, $s :: A$, then there is no judgement $\Gamma \vdash \langle s \rangle t : B[A]$ and $FNF(B[A])$ as by term identity and Context Equivalence A already is a reduct of a dependent proof, hence its term s is already in normal form; if $S \equiv B[A]$ and $\Delta; \Gamma \vdash \langle s \rangle t : B[A] \to_{\eta FNF} (s)t.s : B[A]$, then $[s/v] \cdot t \in \mathcal{T}_x(S)$.

$\Leftarrow$ Any ηFNF reduction provides typable terms: with $S \equiv B[A]$, the following reduction sequence hold:

$$\Gamma \vdash t \to_{\eta FNF} \langle s \rangle t : B[A] \to_{\eta FNF} (s)t.s : B \to_{\eta FNF} (s)t : B.s[a/v]$$

with $a \in \mathcal{T}_x(A)$ and $t \in \mathcal{T}_y(B)$. As the last step is an application, it preserves typing.

□

Lemma 15. *There are no infinite sequences of reductions* $\Delta; \Gamma \vdash t \to_{\eta INF/FNF} t' \to_{\eta INF/FNF} t'' \ldots$

Proof. The proof is by induction on t. To prove the general form, is enough to show that there is no infinite reduction sequence for $\Delta; \Gamma \vdash t \to_{\eta INF} t' \to_{\eta INF} t'' \ldots$, because if t contains no local variable $a : A$, then each term is in FNF-normal form and hence a β-reduced term. If for some term a reduction $\Delta; \Gamma \vdash t \to_{\eta INF} t'$ applies, then there are at most a finite number of reduction steps with $B[A] \in S$ from $\vdash s : S$; any other compound term reduces by an $\to_{\eta INF}$ or $\to_{\eta FNF}$ step, including by Lemma 14 that every assumption $a : A$ requires at most one extra induction step to reduce to the previous finite case. Hence the rewriting relation $\to_{\eta INF/FNF}$ is strongly normalising. □

7 Realization of ILP_{dep}

Normalization has shown that every formula of ILP_{dep} of the form $\langle s \rangle t : B[A]$ when reduced corresponds to one of the form $r : A \supset B$; this can be further broken down to $r \equiv (s)t.s$ for B. This full normalization of dependent proof terms, allows us to compare with the standard language of ILP.[10]

For every proof variable r of an intuitionistic logic-admissible rule, there is an ILP polynomial. We shall give as a last result the proof that this is true for ILP_{dep} and intuitionistic logic with the internalization property:

[10] See [7], in particular section 5.

$$\frac{A_1, \ldots, A_n \vdash_{ILP} B}{a_1 : A_1 \ldots, a_n : A_n \vdash_{ILP} t(a_1, \ldots, a_n) B}$$

The proof is given by adapting the usual Lifting Lemma for ILP to the presence of dependent proof terms:

Lemma 16 (Lifting for Dependent Proof Terms)**.** *Let* $\Delta = a_1 : A_1, \ldots, a_n : A_n$ *and* $\Gamma = b_1 : B_1, \ldots, b_1 : B_n$*. If* $\Delta; \Gamma \vdash t :: C$ *then* $\Delta; \cdot \vdash \langle \boldsymbol{b} \rangle t : C[\boldsymbol{B}]$ *and* $\Delta; \cdot \vdash !s'(\boldsymbol{u}) : s(\boldsymbol{a}, \boldsymbol{b}) : C$*, where* $s(\boldsymbol{a}, \boldsymbol{b}) = (a_1, \ldots, a_n).(b_1, \ldots, b_n) : \boldsymbol{B}$ *and* $s'(\boldsymbol{u}) = XTRT\ \boldsymbol{b} : \boldsymbol{B}\ AS\ u : C[\boldsymbol{B}]\ IN\ !(\Delta.s(\boldsymbol{a}, \boldsymbol{b}))$*.*

Proof. The proof is by induction on Γ, requiring a step by step weakening of Δ by each true assumption $B_i \in \Gamma$ for $C[\mathbf{B}_n]$, followed by an application to a fresh variable a to extract it as a valid variable in order to get an unconditional proof term s in C. An example of one such step with $\Gamma = \{b_i : B_i\}$:

$$\dfrac{\dfrac{\dfrac{\Delta; b_i : B_i \vdash t :: C}{\Delta; \vdash \langle b_i \rangle t : C[B_i]}\, DepProof \qquad \dfrac{\Delta; \vdash b_i : B_i}{\Delta; \cdot \vdash !b_i : (b_i : B_i)}\, !I}{\Delta; \Gamma \vdash XTRT\ !b_i AS\ v_i : B\ IN\ t : C^{v}_{b_i}}\, !E$$

The selection of elements in $\mathbf{B}_n$ proceeds from B_i to B_{i-1} for $\Gamma = \{b_1 : B_1, \ldots, b_n : B_n\}$. The additional step required by the $!I$ Rule allows the discharging of the dependent term, followed by the additional discharging by the ! operator.

□

Theorem 3 (Characterization)**.** *ILP_{dep} is the language of ILP enhanced with an operation on polynomials for the internalization property.*

Proof. Immediate by the construction presented for Lemma 16. □

8 Conclusions

We have shown how to extend the standard setting of the Intuitionistic Logic of Proofs with identity by means of a notion of dependent proof term. This provides an easy extension to the functional fragment of the theory and it suggests a ground comparison with theories of dependent types. Moreover, we have given an interpretation of such dependent terms by means of a natural deduction calculus by separating derivations from valid assumptions and derivations including true assumptions: the notion of dependent proof term is translated as term derivability from locally valid assumptions. Our claim is that the notion of true assumption derived from a dependent term is an appropriate interpretation of possibility for proof terms. Basic properties such as normalisation and confluence are shown for this calculus in terms of reductions to standard non-dependent terms. As last result, normalisation allows to present ILP_{dep} as the standard translation of intuitionistic logic with the internalization property into the Logic of Proofs. Further research shall focus on the use of dependent terms for epistemic purposes, especially in the study and definition of knowledge in a multi-agent and distributed setting.

Acknowledgements

This research started during a visiting research period at the Schools of Philosophy and of Computer Science a the Graduate Center of the City University of New York (USA), supported by the FWO - Research Foundation Flanders. The author wishes to thank Sergei Artemov for many helpful discussions, and Melvin Fitting and all the participants to the Computer Science Colloquium for comments and suggestions when a first version of this work was presented. This final version has been completed during a research period at the Isaac Newton Institute for Mathematical Sciences, Cambridge (UK), where the author was visiting fellow of the scientific programme "Semantics and Syntax: a Legacy of Alan Turing".

References

[1] Andreas Abel, Klaus Aehlig, and Peter Dybjer. Normalization by evaluation for Martin-Löf type theory with one universe. In Marcelo Fiore, editor, *Proceedings of the 23rd Conference on the Mathematical Foundations of Programming Semantics (MFPS XXIII), New Orleans, LA, USA, 11-14 April 2007*, volume 173 of *Electronic Notes in Theoretical Computer Science*, pages 17–39. Elsevier, 2007.

[2] J. Alt and S. Artemov. Reflective λ-calculus. In *Lectures Notes in Computer Science*, volume 2183 of *Proceedings of the Dagstuhl Seminar on Proof Theory in Computer Science*. Springer Verlag, 2001.

[3] Jesse Alt and Sergei Artemov. Reflective λ-calculus. In Reinhard Kahle, Peter Schroeder-Heister, and Robert F. Stärk, editors, *Proof Theory in Computer Science*, volume 2183 of *Lecture Notes in Computer Science*, pages 22–37. Springer, 2001.

[4] S. Artemov. Logic of proofs. *Annals of Pure and Applied Logic*, 67(2):29–59, 1994.

[5] S. Artemov. Explicit provability and constructive semantics. *Bulletin of Symbolic Logic*, 7(1):1–36, 2001.

[6] S. Artemov. The logic of justification. *Review of Symbolic Logic*, 1, no. 4:477–513, 2008.

[7] S. Artemov and R. Iemhoff. The basic intuitionistic logic of proofs. *Journal of Symbolic Logic*, 72(2):439–451, 2007.

[8] S. Artemov and T. Yavorskaya. First-order logic of proofs. Technical Report TR-2011005, CUNY, PhD Program in Computer Science, 2011.

[9] Sergei N. Artëmov and Eduardo Bonelli. The intensional lambda calculus. In Sergei N. Artëmov and Anil Nerode, editors, *LFCS*, volume 4514 of *Lecture Notes in Computer Science*, pages 12–25. Springer, 2007.

[10] Eduardo Bonelli and Federico Feller. The logic of proofs as a foundation for certifying mobile computation. In Sergei N. Artëmov and Anil Nerode,

editors, *LFCS*, volume 5407 of *Lecture Notes in Computer Science*, pages 76–91. Springer, 2009.

[11] Evgenij Dashkov. Arithmetical completeness of the intuitionistic logic of proofs. *Journal of Logic and Computation*, 21(4):665–682, 2011.

[12] N. G. de Bruijn. The mathematical language AUTOMATH, its usage, and some of its extensions. In M. Laudet, D. Lacombe, L. Nolin, and M. Schützenberger, editors, *Proc. of Symposium on Automatic Demonstration, Versailles, France, Dec. 1968*, volume 125, pages 29–61. Berlin, 1970.

[13] V. de Paiva. *Natural Deduction and Context as (Constructive) Modality*, volume 2680 of *Lecture Notes in Artificial Intelligence*, pages 116–129. Springer Verlag, 2003.

[14] M. Fitting. A Quantified Logic of Evidence. *Annals of Pure and Applied Logic*, 152:67–83, 2008.

[15] N. Ghani, V. de Paiva, and E. Ritter. *Explicit Substitutions for Constructive Necessity*, volume Automata, Languages and Programming of *Lecture Notes In Computer Science*, page 743. Springer, 1998.

[16] Neil Ghani. Eta-expansions in dependent type theory - the calculus of constructions. In *Proceedings of the Third International Conference on Typed Lambda Calculus and Applications (TLCA'97*, pages 164–180. Springer-Verlag LNCS, 1997.

[17] K. Gödel. Eine interpretation des intuitionistischen aussagenkalküls. *Ergebnisse des Mathematischen Colloquium*, 4:39–40, 1933.

[18] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. *Journal of the ACM*, 40(1):143–184, January 1993.

[19] S.C. Kleene. On the interpretation of intuitionistic number theory. *Journal of Symbolic Logic*, 10(4), 1945.

[20] P. Martin-Löf. *Intuitionistic Type Theory*. Bibliopolis, 1984.

[21] M. Mendler and V. de Paiva. Constructive CK for Contexts. In *Proceedings of the first Workshop on Context Representation and Reasoning - CONTEXT05*, 2005.

[22] F. Pfenning. Logical frameworks. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning, Vol. 2*. Amsterdam, 2001.

[23] F. Pfenning. Logical frameworks - a brief introduction. In H. Schwichtenberg and R. Steinbrüggen, editors, *Proof and System-Reliability*, volume 62 of *NATO Science Series II*, pages 137–166. Kluwer Academic Publisher, 2002.

[24] G. Primiero. A contextual type theory with judgemental modalities for reasoning from open assumptions. *Logique & Analyse*, 220, 2012.