Reflective λ-calculus

Jesse Alt* Sergei Artemov†

August 2000

Abstract

The Curry-Howard isomorphism converting intuitionistic proofs into typed λ-terms is a simple instance of an internalization property of a system $\boldsymbol{\lambda}^{\infty}$ which unifies intuitionistic logic with λ-calculus and which is capable of internalizing its own derivations as λ-terms. We establish confluence and strong normalization of $\boldsymbol{\lambda}^{\infty}$. The system $\boldsymbol{\lambda}^{\infty}$ considerably extends the expressive power of each of its major components: typed λ-calculus, intuitionistic and modal logic. It may be regarded as the pure λ version of the Logic of Proofs **LP** from [1, 2, 3]. In particular, the standard model of $t\!:\!F$ in $\boldsymbol{\lambda}^{\infty}$ is "t is a proof of F".

1 Introduction

The Curry-Howard isomorphism maps a natural derivation from hypotheses $A_1, A_2, \ldots, A_n \vdash B$ in intuitionistic logic to well defined typed λ-term

$$t(x_1, x_2, \ldots, x_n)\!:\!B,$$

with a dual meaning of $t(x_1, x_2, \ldots, x_n)$ as

i) a term having type B provided its variables $x_1, x_2, \ldots, x_n$ have types $A_1, A_2, \ldots, A_n$ respectively,

ii) a proof of B provided $x_1, x_2, \ldots, x_n$ are proofs of $A_1, A_2, \ldots, A_n$ respectively.

*Cornell University `email: jma35@cornell.edu`

†Cornell University `email: artemov@cs.cornell.edu`

In this respect one can view λ-calculus as propositional level calculus of formal derivations (proofs).

The whole formation calculus of typed λ-terms can be shaped as an internalized version of intuitionistic derivations from hypotheses (cf. [7]), where a derivation of a well-defined λ-term

$$x_1\!:\!A_1, x_2\!:\!A_2, \ldots, x_n\!:\!A_n \Rightarrow t(x_1, x_2, \ldots, x_n)\!:\!B$$

is a step-by-step internal replica of a corresponding intuitionistic derivation of sort

$$A_1, A_2, \ldots, A_n \vdash B.$$

This format suggests that the Curry-Howard isomorphism is a special case of the *Internalization Property (IP)*: if

$$A_1, A_2, \ldots, A_n \vdash B$$

then for some term $t(x_1, x_2, \ldots, x_n)$

$$x_1\!:\!A_1, x_2\!:\!A_2, \ldots, x_n\!:\!A_n \vdash t(x_1, x_2, \ldots, x_n)\!:\!B$$

In our case the upper "$\vdash$" means derivation in intuitionistic logic whereas the second "$\vdash$" denotes derivation in a λ-term formation calculus.

The modal logic format provides an alternative way to represent provability where $\Box F$ is interpreted as "F is provable" or equivalently "there exists a proof of F" (cf. [2, 3, 5, 7]). In particular, the sequent $\Box A_1, \Box A_2, \ldots, \Box A_n \vdash \Box B$ can be read as

"if there exist proofs of $A_1, A_2, \ldots, A_n$, then there exists a proof of B".

The modal reincarnation of internalization is provided by the necessitation rule

$$\frac{A_1, A_2, \ldots, A_n \vdash B}{\Box A_1, \Box A_2, \ldots, \Box A_n \vdash \Box B},$$

which holds for any normal modal logic. Unlike λ-terms, the modal language does not represent proofs directly, but rather via provability, i.e. an assertion that proof exists. Since the formal provability falls into the scope of the Gödel Incompleteness Theorem, the modal format for representing proofs had had serious problems with its semantics ([2, 3]): it took more than 60 years to build an exact intended model for the Gödel provability calculus **S4**.

However, the modal format captures such fundamental property of proofs as *reflection*, i.e. an ability of a formal theory to argue about its own provability. Proper combinations of modality $\Box$ and boolean connectives can represent meaningfull principles which are not expressible by plain λ-terms:

$\Box F \to F$ - correctness of the background proof system

$\Box F \to \Box\Box F$ - existence of proof checking

$\vdash \Box F \vee \Box G \;\Rightarrow\; \vdash \Box F$ or $\vdash \Box G$ - the disjunctive property, etc.

The main motivation of this work is to find a unitied system of representing proofs that combines the explicit character of λ-terms and the reflective abilities of the modal language.

A good starting point is provided by the Logic of Proofs **LP** ([1, 2, 3]) where proofs are represented by advanced combinatory terms called proof polynomials. Proof polynomilas are built from constants and variables by three basic operations: "$\cdot$" (application), "!" (proof checker), and "+" (union). The usual set of type formation rules (propositional formulas formation rules) is extended by a new one: given a proof polynomial p and formula F build a new type (propositional formula) $p\!:\!F$. Valid identities in this language are given by propositional formulas in the extended language generated by the logic calculus having axioms

A0. Axiom schemes of propositional logic	
A1. $t\!:\!F \to F$	*"verification"*
A2. $t\!:\!(F \to G) \;\to (s\!:\!F \to (t\cdot s)\!:\!G)$	*"application"*
A3. $t\!:\!F \;\to\, !t\!:\!(t\!:\!F)$	*"proof checker"*
A4. $s\!:\!F \to (s{+}t)\!:\!F, \quad t\!:\!F \to (s{+}t)\!:\!F$	*"union"*

Rules of inference of **LP** are *modus ponens* and *axiom necessitation*: given a constant c and an axiom F from $A0 - A4$ infer $c\!:\!F$.

A *Constant Specification ($\mathcal{CS}$)* is a finite set of formulas $c_1\!:\!A_1, \ldots, c_n\!:\!A_n$ such that c_i is a constant, and A_i an axiom *A0 – A4*. Each derivation in **LP** naturally generates the $\mathcal{CS}$ consisting of all formulas introduced in this derivation by the *axiom necessitation* rule.

1.1 Remark. Proof constants may be regarded as atomic constant terms (*combinators*) of typed combinatory logic. A constant c_1 specified as $c_1\!:\!(A \to (B \to A))$ can be identified with the combinator $\mathbf{k}^{A,B}$ of the type $A \to (B \to A)$. A constant c_2 such that $c_2 : [(A \to (B \to C)) \to ((A \to B) \to (A \to C))]$ corresponds to the combinator $\mathbf{s}^{A,B,C}$ of the type $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$. The proof variables may be regarded as term variables of combinatory logic, the operation "$\cdot$" as the application of terms. In general an **LP**-formula $t{:}F$ can be read as a combinatory term t of the type F. Typed combinatory logic $\mathbf{CL}_{\rightarrow}$ thus corresponds to a fragment of **LP** consisting only of formulas of the sort $t:F$ where t contains no operations other than "$\cdot$" and F is a formula built from the propositional letters by "$\rightarrow$" only.

1.2 Definition. By intuitionistic $\mathbf{LP}_{\rightarrow\wedge}$ fragment of the Logic of Proofs we mean the system obtained by restricting the propositional part of the **LP** language to $\{\rightarrow, \wedge\}$, functional symbols to $\{\cdot, !\}$, underlying logic (**A0**) to intuitionistic one, and not allowing axiom necessitation rule to introduce the same constant more than once in a given derivation.

2 $\boldsymbol{\lambda}^{\infty}$: confluence and strong normalization

The system $\boldsymbol{\lambda}^{\infty}$ is a joint calculus of propositions (types) and proofs (λ-terms). The language of $\boldsymbol{\lambda}^{\infty}$ contains[1]

propositional letters (atomic types) $p_1, p_2, p_3, \ldots$
variables $x_1, x_2, x_3, \ldots$
connectives $\rightarrow, \wedge$
functional symbols - unary $!$, $\Uparrow^n$, $\Downarrow^n$, $\boldsymbol{\pi}_0^n$, $\boldsymbol{\pi}_1^n$; binary $\circ^n$, $\mathbf{p}^n$[2],
operator symbols ':', λ^n, $n = 1, 2, 3, \ldots$.

Terms τ of $\boldsymbol{\lambda}^{\infty}$ are build from variables x by functional symbols and λ^n-operators in the usual manner:

$$\tau = x \mid\ !\ \tau \mid \Uparrow^n \tau \mid \Downarrow^n \tau \mid \boldsymbol{\pi}_0^n \tau \mid \boldsymbol{\pi}_1^n \tau \mid \tau \circ^n \tau \mid \mathbf{p}^n(\tau, \tau) \mid \lambda^n x.\tau.$$

Formulas (types) φ are built from propositional letters p and terms τ by connectives and operator ':'

$$\varphi = p \mid \varphi \rightarrow \varphi \mid \varphi \wedge \varphi \mid \tau : \varphi.$$

[1]For the sake of conciseness we will follow a well established tradition in the typed λ-calculus to first present the "noble" fragment based on the intuitionistic logic with $\rightarrow$ and $\wedge$ only. We foresee no serious problems in extending $\boldsymbol{\lambda}^{\infty}$ to the full intuitionistic logic with $\vee$ and $\perp$ as well as to the classical logic on the background.

[2]As usual, f^1 is read as f.

The informal semantics for $t{:}F$ is "t is a proof of F"; so a formula

$$t_n{:}t_{n-1}{:}\ldots t_1{:}A$$

should be read as "t_n is a proof that t_{n-1} is a proof that ... is a proof that t_1 proves A". For the sake of brevity we will refer to formulas as terms (of depth 0).

2.1 The $\boldsymbol{\lambda}^\infty$ calculus

In this section we introduce the calculus $\boldsymbol{\lambda}^\infty$ based on two clear principles

1. $\boldsymbol{\lambda}^\infty$ contains natural deductions of intuitionistic logic
2. $\boldsymbol{\lambda}^\infty$ enjoys the Internalization: *if* $A_1, A_2, \ldots, A_n \vdash B$ *then*

$$x_1{:}A_1, x_2{:}A_2, \ldots, x_n{:}A_n \vdash t(x_1, x_2, \ldots, x_n){:}B$$

for some $\boldsymbol{\lambda}^\infty$*-term* $t(x_1, x_2, \ldots, x_n)$.

In $\boldsymbol{\lambda}^\infty$ we use the natural deduction format, where derivations are represented by proof trees with assumptions, both open (charged) and closed (discharged). A discharged assumption A will be denoted by $[A]$. We will also use a concise sequent style notation for derivations in $\boldsymbol{\lambda}^\infty$ by reading $\Gamma \vdash F$ as an $\boldsymbol{\lambda}^\infty$-derivation of F with the set of open assumptions Γ.

2.1 Definition. We will identify *well-defined terms* with derivations in the calculus $\boldsymbol{\lambda}^\infty$ of the form

$$x_1{:}A_1, x_2{:}A_2, \ldots, x_n{:}A_n \vdash t(x_1, x_2, \ldots, x_n){:}B.$$

This may be also read in the usual λ-term manner: term $t(x_1, x_2, \ldots, x_n)$ has type B provided each variable x_i has type A_i for all $i = 1, 2, \ldots, n$.

Within the current definition below we assume that $n = 0, 1, 2, \ldots$ and $\vec{v} = (v_1, v_2, \ldots, v_n)$. In particular, if $n = 0$ then $\vec{v}$ is empty. We also agree on the following vector-style notations:

$\vec{t}{:}A$ denotes $t_n{:}t_{n-1}{:}\ldots t_1{:}A$ (e.g. $\vec{t}{:}A$ is A, when $n = 0$),

$\vec{t}{:}\{A_1, A_2, \ldots, A_n\}$ denotes $\{t_1{:}A_1, t_2{:}A_2, \ldots, t_n{:}A_n\}$,

$\lambda^n \vec{x}.\vec{t}{:}B$ denotes $\lambda^n x_n.t_n{:}\lambda^{n-1}x_{n-1}.t_{n-1}{:}\ldots \lambda x_1.t_1{:}B$,

$(\vec{t} \circ^n \vec{s}){:}B$ denotes $(t_n \circ^n s_n){:}(t_{n-1} \circ^{n-1} s_{n-1}){:}\ldots (t_1 \circ s_1){:}B$,

$\Uparrow^n \vec{t}\!:\!B$ denotes $\Uparrow^n t_n : \Uparrow^{n-1} t_{n-1} : \ldots \Uparrow t_1 : B$,

likewise for all other functional symbols of $\boldsymbol{\lambda}^\infty$.

Derivations (i.e. well-defined terms) are generated by the following clauses. Here A, B, C are formulas, Γ a finite set of formulas, $\vec{s}, \vec{t}, \vec{u}$ are n-vectors of terms, $\vec{x}$ are a n-vectors of variables, $n = 0, 1, 2, \ldots$.

	Natural deduction rule	*Its sequent form*
(Ax)	$\vec{x}\!:\!A$	$\Gamma, \vec{x}\!:\!A \vdash \vec{x}\!:\!A.$

($\boldsymbol{\lambda}$) $\dfrac{\vec{t}\!:\!B}{\lambda^n \vec{x}.\vec{t}\!:\!(A \to B)}$ As usual, we discharge premises corresponding to variables bounded by this rule.

None of $\vec{x}$ should occur free in the derivation after all the premises corresponding to those variables are discharged. In the full sequent form this rule looks

$$\frac{\Gamma, x_n : x_{n-1} : \ldots x_1 : A \vdash t_n : t_{n-1} : \ldots t_1 : B}{\Gamma \vdash \lambda^n x_n.t_n : \lambda^{n-1} x_{n-1}.t_{n-1} : \ldots \lambda x_1.t_1 : (A \to B)},$$

where none of $\vec{x}$ occurs free in the conclusion sequent.

(App) $\dfrac{\vec{t}\!:\!(A \to B) \quad \vec{s}\!:\!A}{(\vec{t} \circ^n \vec{s})\!:\!B}$ $\qquad \dfrac{\Gamma \vdash \vec{t}\!:\!(A \to B) \quad \Gamma \vdash \vec{s}\!:\!A}{\Gamma \vdash (\vec{t} \circ^n \vec{s})\!:\!B}$

(p) $\dfrac{\vec{t}\!:\!A \quad \vec{s}\!:\!B}{\mathbf{p}^n(\vec{t}, \vec{s})\!:\!(A \wedge B)}$ $\qquad \dfrac{\Gamma \vdash \vec{t}\!:\!A \quad \Gamma \vdash \vec{s}\!:\!B}{\Gamma \vdash \mathbf{p}^n(\vec{t}, \vec{s})\!:\!(A \wedge B)}$

($\boldsymbol{\pi}$) $\dfrac{\vec{t}\!:\!(A_0 \wedge A_1)}{\boldsymbol{\pi}_i^n \vec{t}\!:\!A_i}\ (i = 0, 1)$ $\qquad \dfrac{\Gamma \vdash \vec{t}\!:\!(A_0 \wedge A_1)}{\Gamma \vdash \boldsymbol{\pi}_i^n \vec{t}\!:\!A_i}\ (i = 0, 1)$

($\Uparrow$) $\dfrac{\vec{t}\!:\!u\!:\!A}{\Uparrow^n \vec{t}\!:\,!u\!:\!u\!:\!A}$ $\qquad \dfrac{\Gamma \vdash \vec{t}\!:\!u\!:\!A}{\Gamma \vdash \Uparrow^n \vec{t}\!:\,!u\!:\!u\!:\!A}$

($\Downarrow$) $\dfrac{\vec{t}\!:\!u\!:\!A}{\Downarrow^n \vec{t}\!:\!A}$ $\qquad \dfrac{\Gamma \vdash \vec{t}\!:\!u\!:\!A}{\Gamma \vdash \Downarrow^n \vec{t}\!:\!A}$

2.2 Remark. It is clear that minimal propositional logic, with rules for implication and conjunction only, is contained in $\boldsymbol{\lambda}^\infty$. Indeed, if we require, for a derivation in $\boldsymbol{\lambda}^\infty$, that $n = 0$, and use the corresponding rules only, we have the system $\mathbf{Ni}_{\to\wedge}$ (cf. [7]). Similarly, the usual typed λ-calculus over $\to, \wedge$ corresponds to the level $n = 1$.

2.3 Example. Here are some examples of $\boldsymbol{\lambda}^\infty$-derivations in the sequent format (cf. 3.2). We skip the trivial axiom parts for brevity.

$$\frac{x : A \vdash\ !x : x : A}{\vdash x : A \to\ !x : x : A}$$

However, one can derive neither $\vdash A \to x : A$, nor $\vdash \lambda x.!x : (A \to x : A)$, since x occurs free there.

$$\frac{y : x : A \vdash \Uparrow y : !x : x : A}{\vdash \lambda y.\Uparrow y : (x : A \to\ !x : x : A)} \qquad \frac{x : A \vdash A}{\vdash x : A \to A} \qquad \frac{y : x : A \vdash \Downarrow y : A}{\vdash \lambda y.\Downarrow y : (x : A \to A)}$$

$$\cfrac{\cfrac{u : x : A, v : y : B \vdash \mathbf{p}^2(u, v) : \mathbf{p}(x, y) : (A \wedge B)}{u : x : A \vdash \lambda v.\mathbf{p}^2(u, v) : (y : B \to \mathbf{p}(x, y) : (A \wedge B))}}{\vdash \lambda uv.\mathbf{p}^2(u, v) : (x : A \to (y : B \to \mathbf{p}(x, y) : (A \wedge B)))}$$

$$\cfrac{\cfrac{u : x : A, v : y : B \vdash \mathbf{p}^2(u, v) : \mathbf{p}(x, y) : (A \wedge B)}{u : x : A \vdash \lambda^2 v.\mathbf{p}^2(u, v) : \lambda y.\mathbf{p}(x, y) : (B \to (A \wedge B))}}{\vdash \lambda^2 uv.\mathbf{p}^2(u, v) : \lambda xy.\mathbf{p}(x, y) : (A \to (B \to (A \wedge B)))}$$

$$\cfrac{\cfrac{u : x : A, v : y : B \vdash \mathbf{p}^2(u, v) : \mathbf{p}(x, y) : (A \wedge B)}{u : x : A, v : y : B \vdash \Uparrow \mathbf{p}^2(u, v) : !\mathbf{p}(x, y) : \mathbf{p}(x, y) : (A \wedge B)}}{\vdash \lambda uv.\Uparrow \mathbf{p}^2(u, v) : (x : A \to (y : B \to\ !\mathbf{p}(x, y) : \mathbf{p}(x, y) : (A \wedge B))}$$

Note that unlike in the previous example we cannot introduce λ^2 in place of λ at the last stage here since the resulting sequent would be

$$\vdash \lambda^2 uv.\Uparrow \mathbf{p}^2(u, v) : \lambda xy.!\mathbf{p}(x, y) : ((A \to (B \to \mathbf{p}(x, y) : (A \wedge B))$$

containing variables x, y free, which is illegal.

$$\frac{x\!:\!y\!:\!(A\!\to\! B), s\!:\!t\!:\!A \vdash (x \circ^2 s)\!:\!(y \circ t)\!:\!B}{s\!:\!t\!:\!A \vdash x\!:\!y\!:\!(A\!\to\! B)\!\to\!(x \circ^2 s)\!:\!(y \circ t)\!:\!B}$$

$$\frac{x\!:\!y\!:\!(A\!\to\! B), s\!:\!t\!:\!A \vdash (x \circ^2 s)\!:\!(y \circ t)\!:\!B}{s\!:\!t\!:\!A \vdash \lambda x.(x \circ^2 s)\!:\!(y\!:\!(A\!\to\! B)\!\to\!(y \circ t)\!:\!B)}$$

$$\frac{x\!:\!y\!:\!(A\!\to\! B), s\!:\!t\!:\!A \vdash (x \circ^2 s)\!:\!(y \circ t)\!:\!B}{s\!:\!t\!:\!A \vdash \lambda^2 x.(x \circ^2 s)\!:\!\lambda y.(y \circ t)\!:\!((A\!\to\! B)\!\to\! B)}$$

2.4 Theorem. (Internalization Property for $\boldsymbol{\lambda}^\infty$) *Let $\boldsymbol{\lambda}^\infty$ derive*

$$A_1, A_2, \ldots, A_m \vdash B.$$

Then one can build a well-defined term $t(x_1, x_2, \ldots, x_m)$ with fresh variables $\vec{x}$ such that $\boldsymbol{\lambda}^\infty$ also derives

$$x_1\!:\!A_1, x_2\!:\!A_2, \ldots, x_m\!:\!A_m \vdash t(x_1, x_2, \ldots, x_m)\!:\!B.$$

Proof. The idea is straightforward: increment n at every node of the derivation $A_1, A_2, \ldots, A_n \vdash B$. The base case is obvious. We will check the most principal step clause **($\boldsymbol{\lambda}$)** leaving the rest as an easy exercise. Let the last step of a derivation be

$$\frac{\Gamma, y_n\!:\!y_{n-1}\!:\!\ldots y_1\!:\!A \vdash t_n\!:\!t_{n-1}\!:\!\ldots t_1\!:\!B}{\Gamma \vdash \lambda^n y_n.t_n\!:\!\lambda^{n-1} y_{n-1}.t_{n-1}\!:\!\ldots \lambda y_1.t_1\!:\!(A\!\to\! B)}\,.$$

By the induction hypothesis, for some term $s(\vec{x}, x_{m+1})$ of fresh variables $\vec{x}, x_{m+1}$

$$\vec{x}\!:\!\Gamma, x_{m+1}\!:\!y_n\!:\!y_{n-1}\!:\!\ldots y_1\!:\!A \vdash s(\vec{x}, x_{m+1})\!:\!t_n\!:\!t_{n-1}\!:\!\ldots t_1\!:\!B.$$

Apply the rule **($\boldsymbol{\lambda}$)** for $n+1$ to obtain

$$\vec{x}\!:\!\Gamma \vdash \lambda^{n+1} x_{m+1}.s\!:\!\lambda^n y_n.t_n\!:\!\lambda^{n-1} y_{n-1}.t_{n-1}\!:\!\ldots \lambda y_1.t_1\!:\!(A\!\to\! B),$$

and put $t(x_1, x_2, \ldots, x_m) = \lambda^{n+1} x_{m+1}.s(\vec{x}, x_{m+1})$.
◄

2.5 Definition. For $n = 0, 1, 2, \ldots$, the redexes for $\boldsymbol{\lambda}^\infty$ and their contracta (written as $t \rhd t'$, t a redex, t' a contractum of t) are:

- $(\lambda^n x.t)\circ^n s : F \ \rhd\ t[s/x] : F$ $\quad$ (β-contraction)
- $\boldsymbol{\pi}_i^n \mathbf{p}^n(t_0,t_1) : F_i \ \rhd\ t_i : F_i,\ i \in \{0,1\}$
- $\Downarrow^n \Uparrow^n t : F \rhd\ t : F$

2.2 Confluence of the $\boldsymbol{\lambda}^\infty$calculus

2.6 Definition. **(a)** For any well-defined $\boldsymbol{\lambda}^\infty$-terms $t,\ t'$, t *one-step reduces* to t' ($t \succ_1 t'$) if t contains a subterm occurrence s of the form of one of the redexes above, $s \rhd s'$, and $t' \equiv t[s'/s]$.

(b) The reduction relation $\succeq$ is the reflexive and transitive closure of $\succ_1$.

2.7 Theorem. The reduction relation $\succeq$ is confluent.

Proof. We generalize the standard method of proving confluence (cf. [4]) For given n, we define three notions of *parallel reduction at level* n ($\propto_{a,n}$, $\propto_{b,n}$, and $\propto_{c,n}$), which correspond to simultaneously contracting all redexes of a certain type. We then prove confluence for each type of parallel reduction, and prove that the three types commute, and so the general notion of parallel reduction at level n ($\propto_n$) is confluent. It's clear that, for $j \neq k$, parallel reduction at level j commutes with parallel reduction at level k. Thus, the general notion of parallel reduction is confluent. The transitive closure of this relation equals $\succeq$, and it follows that $\succeq$ is confluent.

2.8 Definition. For given n, the *parallel reduction* relations $\propto_{a,n}$,$\propto_{b,n}$ and $\propto_{c,n}$ are generated by the following clauses:

1. $t \propto_n t$
2. If $t \ \propto_n\ t'$, then $\lambda^m x.t\ \propto_n\ \lambda^m x.t'$, $\Uparrow^m t\ \propto_n \Uparrow^m t'$, $\Downarrow^m t\ \propto_n \Downarrow^m t'$, $\pi_i^m t\ \propto_n\ \pi_i^m t'$, and $!t \propto_n !t'$, for $m = 1, 2, \ldots$
3. If $t \propto_n t'$, and $s \propto_n s'$, then $t\circ^m s \propto_n t'\circ^m s'$, and $p^m(t,s) \propto_n p^m(t',s')$ for $m = 1, 2, \ldots$

In addition, the following clauses are particular to the relations:

4a. If $t \propto_{a,n} t'$ and $s \propto_{a,n} s'$, then $(\lambda^n x.t) \circ^n s \propto_{a,n} t'[s'/x]$

4b. If $t \propto_{b,n} t'$ then $\Downarrow^n \Uparrow^n t \propto_{b,n} t'$

4c. If $t_i \propto_{c,n} t'_i$ then $\pi^n_i p^n(t_0, t_1) \propto_{c,n} t'_i$

2.9 Proposition. If $t \propto_{z,n} t'$ and $s \propto_{z,n} s'$ then $t[s/x] \propto_{z,n} t'[s'/x]$, for $z \in \{a, b, c\}$.

Proof. The proof follows [4]. The new cases are easily handled.
◀

2.10 Proposition. $\propto_{a,n}$ is confluent.

Proof. First, note that the following statements hold:

(i) If $t \circ^n s \propto_{a,n} u$ then either $u \equiv t' \circ^n s'$ with $t \propto_{a,n} t'$ and $s \propto_{a,n} s'$, or $t \equiv \lambda^n x.t_0$ and $u \equiv t'_0[s'/x]$, with $t_0 \propto_{a,n} t'_0$ and $s \propto_{a,n} s'$

(ii) If $t \propto_{a,n} t'$ and t is not of the form $t_0 \circ^n s$, then t' is of the same form as t, and– if s_0 (and s_1) is the primary proper sub-term(s) of t, s'_0 (and s'_1) the largest proper sub-term(s) of t'– $s_0 \propto_{a,n} s'_0$ (and $s_1 \propto_{a,n} s'_1$).

By induction on $t \propto_{a,n} t'$, we show that, for all t'' such that $t \propto_{a,n} t''$, there exists a t''' such that $t', t'' \propto_{a,n} t'''$.

Case 1 $t \propto_{a,n} t'$ because $t \equiv t'$. Then take $t''' \equiv t''$.

Case 2 $t \propto_{a,n} t'$ by either 2 or 3, and t is not of the form $t_0 \circ^n t_1$. Then the induction hypothesis gives us t'''. For example, suppose t has the form $p^n(t_0, t_1)$. Then, from the observation above, $t' \equiv p^n(t'_0, t'_1)$, and $t_0 \propto_{a,n} t'_0$, $t_1 \propto_{a,n} t'_1$. And, for all t'' such that $t \propto_{a,n} t''$, t'' has the form $p^n(t''_0, t''_1)$, with $t_0 \propto_{a,n} t''_0$ and $t_1 \propto_{a,n} t''_1$. By induction, there exist t'''_0 and t'''_1 such that $t'_0, t''_0 \propto_{a,n} t'''_0$ and $t'_1, t''_1 \propto_{a,n} t'''_1$. Then take $t''' \equiv p^n(t'''_0, t'''_1)$.

Case 3 $t \propto_{a,n} t'$ by clause 3, and t has form $t_0 \circ^n t_1$. Then $t' \equiv t'_0 \circ^n t'_1$, $t_0 \propto_{a,n} t'_0$, $t_1 \propto_{a,n} t'_1$, and there are two sub-cases:

Sub-case 3.1 $t \propto_{a,n} t''$ by clause 3. Then $t'' \equiv t''_0 \circ^n t''_1$, with $t_0 \propto_{a,n} t''_0$, $t_1 \propto_{a,n} t''_1$. By induction, there exist t'''_0 and t'''_1 such that $t'_0, t''_0 \propto_{a,n} t'''_0$ and $t'_1, t''_1 \propto_{a,n} t'''_1$. Then take $t''' \equiv t'''_0 \circ^n t'''_1$.

Sub-case 3.2 $t \propto_{a,n} t''$ by clause 4a. Then $t \equiv (\lambda^n x.s) \circ^n t_1$ and $t'' \equiv s''[t_1''/x]$ with $s \propto_{a,n} s''$, $t_1 \propto_{a,n} t_1''$. Then $t_0' \equiv \lambda^n x.s'$ and $s \propto_{a,n} s'$. By induction, there are s''', t_1''' such that $s', s'' \propto_{a,n} s'''$, $t_1', t_1'' \propto_{a,n} t_1'''$. We take $t''' \equiv s'''[t_1'''/x]$.

Case 4 $t \propto_{a,n} t'$ by 4a. Then $t \equiv (\lambda^n x.t_0) \circ^n t_1$, and $t' \equiv t_0'[t_1'/x]$, with $t_0 \propto_{a,n} t_0'$, $t_1 \propto_{a,n} t_1'$; again there are two sub-cases:

Sub-case 4.1 $t \propto_{a,n} t''$ by 3. Then $t'' \equiv (\lambda^n x.t_0'') \circ^n t_0''$, $t_0 \propto_{a,n} t_0''$, $t_1 \propto_{a,n} t_1''$. By induction, there exist t_0''', t_1''' such that $t_0', t_0'' \propto_{a,n} t_0'''$ and $t_1', t_1'' \propto_{a,n} t_1'''$. We take $t''' \equiv t_0'''[t_1'''/x]$.

Sub-case 4.2 $t \propto_{a,n} t''$ by 4a. Then $t'' \equiv t_0''[t_1''/x]$, with $t_0 \propto_{a,n} t_0''$, $t_1 \propto_{a,n} t_1''$. By induction, there exist t_0''', t_1''' such that $t_0', t_0'' \propto_{a,n} t_0'''$ and $t_1', t_1'' \propto_{a,n} t_1'''$. We take $t''' \equiv t_0'''[t_1'''/x]$.

The confluence of $\propto_{b,n}$ and $\propto_{c,n}$ are established in a similar manner. Below we describe one of the interesting cases in the proof for $\propto_{b,n}$.

Case 4 $t \propto_{b,n} t'$ by 4b. Then $t \equiv \Downarrow^n \Uparrow^n t_0$ and $t' \equiv t_0'$, with $t_0 \propto_{b,n} t_0'$. There are two sub-cases.

Sub-case 4.1 $t \propto_{b,n} t''$ by 2. Then $t'' \equiv \Downarrow^n \Uparrow^n t_0''$, with $t_0 \propto_{b,n} t_0''$. By induction, there exists t_0''' such that $t_0', t_0'' \propto_{b,n} t_0'''$. We take $t''' \equiv t_0'''$.

Sub-case 4.2 $t \propto_{b,n} t''$ by 4b. Then $t'' \equiv t_0''$, with $t_0 \propto_{b,n} t_0''$. By induction, there exists t_0''' such that $t_0', t_0'' \propto_{b,n} t_0'''$. We take $t''' \equiv t_0'''$.

◀

In establishing the commutativity of $\propto_{a,n}, \propto_{b,n}$, and $\propto_{c,n}$, it suffices to notice that, for $t \propto_{x,n} t'$, $t \propto_{y,n} t''$, $x, y \in \{a, b, c\}$, and $x \not\equiv y$, either the two reductions are by clauses 1.-3., in which case they're easily handled by the induction hypothesis, or they're by one of the clauses 4a,b,c., in which case the reductions are disjoint.

◀

2.3 Strong normalization of $\boldsymbol{\lambda}^\infty$-terms

2.11 Definition. **(a)** A *reduction sequence* is any sequence of terms $t_0, t_1, \ldots, t_n$ such that, for any $j \in \{0, \ldots, n-1\}$, $t_j \succ_1 t_{j+1}$. **(b)** A term t is *strongly normalizing* ($t \in \mathbf{SN}$) if every possible reduction sequence for t is finite. **(c)** If t is strongly normalizing, $\boldsymbol{\nu(t)}$= the least n such that every reduction sequence for t terminates in fewer than n steps.

We now show that all terms of $\boldsymbol{\lambda}^\infty$ are strongly normalizing. The proof follows the method of Tait (cf. [6]). For t a term, we say t is *of level* n if the primary rule in the derivation of t is a level n rule. For example, $\lambda^2 x.(x \circ^2 s)$ is a term of level 2. A case in which this rule of identifying a term's level may be ambiguous, is for an application of the rule $\Uparrow^0$, giving, from the term $u : A$, $!u : u : A$. The difficulty lies in a confusion over which term one has in mind. In this example, there is both the 0-level term t_0 (since it's a 0-level, or empty, term we don't bother writing it down) which has type $!u : u : A$. Alternatively, one has the level 1 term $!u$, of type $u : A$.

For the proof, we'll define a set $\mathtt{RED}^n_T$ of reducible terms for n level terms, by induction on type T. We then prove some properties of these sets of terms, including the property that all terms in the sets are strongly normalizing. Finally, strong normalization of the calculus is established by showing that all λ^∞-terms are in a set $\mathtt{RED}^n_T$. To begin with, we need a definition of a type's depth to induct on.

If T is a type, we define the *depth* of T, $|T|$, inductively:

$|p| = 0$, for p an atomic proposition

$|A \wedge B| = |A| + |B|$

$|A \rightarrow B| = max(|A|, |B|) + 1$

$|u : A| = |A| + 1$

2.3.1 Reducibility predicate

2.12 Definition. The sets $\mathtt{RED}^n_T$ of reducible terms of level n and type T are defined by induction on $|T|$.

1. For t of atomic type P, $t \in \mathtt{RED}^n_P$ if t is strongly normalizing.

2. For t of type $A \to B$, $t \in \mathtt{RED}^n_{A\to B}$ if, for all $s \in \mathtt{RED}^n_A$, $t \circ^n s \in \mathtt{RED}^n_B$.

3. For t of type $A_0 \wedge A_1$, $t \in \mathtt{RED}^n_{A_0 \wedge A_1}$ if $\pi^n_i t \in \mathtt{RED}^n_{A_i}$, for $i = 0, 1$.

4. For t of type $u : A$, $t \in \mathtt{RED}^n_{u:A}$ if $\Downarrow^n t \in \mathtt{RED}^n_A$

2.3.2 Properties of reducibility

2.13 Definition. A term t is *neutral* if t is not of the form $\lambda^n x.t_0$, $p^n(t_0, t_1)$, or $\Uparrow^n t_0$.

2.14 Proposition. For T any type, all terms $t \in \mathtt{RED}^n_T$ have the following properties:

(CR 1) If $t \in \mathtt{RED}^n_T$, then t is strongly normalizing.

(CR 2) If $t \in \mathtt{RED}^n_T$ and $t \succeq t'$, then $t' \in \mathtt{RED}^n_T$.

(CR 3) If t is neutral, and $t' \in \mathtt{RED}^n_T$ for all t' such that $t \succ_1 t'$, then $t \in \mathtt{RED}^n_T$.

As a special case of **(CR 3)**:

(CR 4) If t is neutral and normal, then $t \in \mathtt{RED}^n_T$.

Proof. Proof by induction on the type T.

Step 1 T is atomic.

(CR 1) Trivial.

(CR 2) If t is **SN**, and $t \succeq t'$, then t' is **SN**, so t' is reducible.

(CR 3) Let $M = max(\nu(t') \mid t \succ_1 t')$. There are finitely many t' such that $t \succ_1 t'$, so M is finite. And, $\nu(t) \leq M + 1$.

Step 2 $T \equiv (A \to B)$.

(CR 1) Let x be a variable of type A. Then $t \circ^n x \in \mathtt{RED}^n_B$, and by induction, $t \circ^n x$ is **SN**. Since every reduction sequence for t is embedded in a reduction sequence of $t \circ^n x$, t is **SN**.

(CR 2) Let $t \succeq t'$, and take $s \in \mathtt{RED}^n_A$. Then $t \circ^n s \in \mathtt{RED}^n_B$, and $t \circ^n s \succeq t' \circ^n s$. By induction, $t' \circ^n s$ is reducible. Since s was arbitrary, t' is reducible.

(CR 3) Let t be neutral, and suppose that t' is reducible, for all t' such that $t \succ_1 t'$. Let $s \in \mathtt{RED}^n_A$. By induction, s is **SN**. We prove, by induction on $\nu(s)$ that if $t \circ^n s \succ_1 (t \circ^n s)'$, $(t \circ^n s)'$ is reducible. There are two possibilities for $(t \circ^n s)'$:

1. $(t \circ^n s)'$ is $t' \circ^n s$, with $t \succ_1 t'$. Then t' is reducible, so $t' \circ^n s$ is.
2. $(t \circ^n s)'$ is $t \circ^n s'$, with $s \succ_1 s'$. Then s' is reducible, so $t \circ^n s'$ is.

By the induction hypothesis, $t \circ^n s$ is reducible, because it's neutral. Thus, t is reducible.

Step 3 $T \equiv (A \wedge B)$.

(CR 1) If t of type $(A \wedge B)$ is reducible, then $\pi^n_1 t$ is reducible. By induction, $\pi^n_1 t$ is **SN**. Since $\nu(t) \leq \nu(\pi^n_1 t)$, t is **SN**.

(CR 2) If $t \succeq t'$ then $\pi^n_i t \succeq \pi^n_i t'$, for $i \in \{0, 1\}$. By induction, $\pi^n_i t'$ is reducible, so t' is.

(CR 3) Let t be neutral, and t' reducible, for all t' such that $t \succ_1 t'$. If $\pi^n_i t \succ_1 s$, then $s \equiv \pi^n_i t'$, with $t \succ_1 t'$, because t is neutral. Since t' is reducible, for all such t', all the $\pi^n_i t'$ are reducible. By induction, $\pi^n_i t$ is reducible, so t is reducible.

Step 4 $T \equiv (u : A)$.

(CR 1) If t of type $(u : A)$ is reducible, then $\Downarrow^n t$ is reducible. By induction, $\Downarrow^n t$ is **SN**, which implies t is, too.

(CR 2) Let $t \succeq t'$. Then also $\Downarrow^n t \succeq \Downarrow^n t'$. By induction, $\Downarrow^n t'$ is reducible. Thus, t' is reducible.

(CR 3) Let t be neutral, and t' reducible, for all t' such that $t \succ_1 t'$. Since t is neutral, if $\Downarrow^n t \succ_1 s$, then $s \equiv \Downarrow^n t'$, with $t \succ_1 t'$, and all such terms are reducible. By induction, $\Downarrow^n t$ is reducible, which means t is reducible.

◀

2.3.3 All well-defined terms of $\boldsymbol{\lambda}^\infty$ are reducible

Before proving the main theorem, we need the following lemmas.

2.15 Lemma. If, for all $u \in \mathtt{RED}^n_A$, $t[u/x] \in \mathtt{RED}^n_B$, then $\lambda^n x.t \in \mathtt{RED}^n_{A\to B}$.

Let $u \in \mathtt{RED}^n_A$. We'll show that $(\lambda^n x.t) \circ^n u$ is reducible, by induction on $\nu(u) + \nu(t)$. $(\lambda^n x.t) \circ^n u \succ_1$:

- $t[u/x]$, which is reducible by hypothesis.
- $(\lambda^n x.t') \circ^n u$ with $t \succ_1 t'$. By induction, this is reducible.
- $(\lambda^n x.t) \circ^n u'$ with $u \succ_1 u'$. By induction, this is reducible.

2.16 Lemma. If t_0, t_1 are reducible, then $p^n(t_0, t_1)$ is reducible.

Proof by induction on $\nu(t_0) + \nu(t_1)$. $\pi^n_i p^n(t_0, t_1) \succ_1$:

- t_i, which is reducible by hypothesis.
- $\pi^n_i p^n(t'_0, t_1)$ with $t_0 \succ_1 t'_0$, which is reducible by induction.
- $\pi^n_i p^n(t_0, t'_1)$ with $t_1 \succ_1 t'_1$, which is reducible by induction.

2.17 Lemma. If $t \in \mathtt{RED}^n_{u:A}$, then $\Uparrow^n t \in \mathtt{RED}^n_{!u:u:A}$.

By induction on $\nu(t)$. $\Downarrow^n \Uparrow^n t \succ_1$:

- t, which is reducible by hypothesis.
- $\Downarrow^n \Uparrow^n t'$ with $t \succ_1 t'$, which is reducible by induction.

2.18 Lemma. If $t \in \mathtt{RED}_A$, then $!t \in \mathrm{RED}_{t:A}$.

By induction on $\nu(t)$.

2.19 Proposition. Let $t \in \boldsymbol{\Lambda}^\infty$. If the free variables of t are among $x_1, x_2, \ldots, x_k$ having types $U_1, U_1, \ldots, U_k$, then for any terms $u_1, u_2, \ldots, u_k$ reducible of types $U_1, U_2, \ldots, U_k$, the term $t[u_1/x_1, u_2/x_2, \ldots, u_k/x_k]$ (which we will abbreviate as $t[\vec{u}/\vec{x}]$), is reducible.

Proof. By induction on the depth of t.

Step 1 If t is entered by **(Ax)**, then it's trivial, since t is a variable.

Step 2 If t is $\lambda^n y.t_0$, then by induction, $t_0[v/y, \vec{u}/\vec{x}]$ is reducible, for all reducible v. By Lemma 2.15, $\lambda^n y.(t_0[\vec{u}/\vec{x}])$ is reducible.

Step 3 If t is $t_0 \circ^n t_1$, then by induction, $t_0[\vec{u}/\vec{x}]$ and $t_1[\vec{u}/\vec{x}]$ are reducible. Then, by definition, $(t_0[\vec{u}/\vec{x}] \circ^n t_1[\vec{u}/\vec{x}])$ is reducible.

Step 4 If t is $p^n(t_0, t_1)$, then by induction, $t_0[\vec{u}/\vec{x}]$ and $t_1[\vec{u}/\vec{x}]$ are reducible. By Lemma 2.16, $p^n(t_0[\vec{u}/\vec{x}], t_1[\vec{u}/\vec{x}])$ is reducible.

Step 5 If t is $\pi_i^n t_0$, then by induction, $t_0[\vec{u}/\vec{x}]$ is reducible. By definition, $\pi_i^n(t_0[\vec{u}/\vec{x}])$ is reducible.

Step 6 If t is $\Uparrow^n t_0$, then by induction, $t_0[\vec{u}/\vec{x}]$ is reducible. By Lemma [**?**], $\Uparrow^n(t_0[\vec{u}/\vec{x}])$ is reducible.

Step 7 If t is $\Downarrow^n t_0$, then by induction, $t_0[\vec{u}/\vec{x}]$ is reducible. By definition, $\Downarrow^n(t_0[\vec{u}/\vec{x}])$ is reducible.

Step 8 If t is $!t_0$, then by induction $t_0[\vec{u}/\vec{x}]$ is reducible. By Lemma [**?**], $t[\vec{u}/\vec{x}]$ is reducible.

◀

As a corollary, we have:

2.20 Theorem. All well-defined terms of $\boldsymbol{\lambda}^\infty$ are strongly normalizable.

3 The $\boldsymbol{\lambda}^+$ fragment vs. LP

3.1 Definition. **(a)** The $\boldsymbol{\lambda}^+$ fragment of the $\boldsymbol{\lambda}^\infty$ calculus is obtained by restricting to the rules:

$$\{\boldsymbol{\lambda},\ \mathbf{p},\ \boldsymbol{\pi_i},\ \Uparrow,\ \Downarrow \mid n = 0, 1\}, \quad \{\mathbf{App} \mid n = 0, 1, 2\}$$

(b) The redexes of $\boldsymbol{\lambda}^+$ are the redexes of $\boldsymbol{\lambda}^\infty$, restricted to the rules above.

There is a natural embedding $*$ of intuitionistic logic of proofs $\mathbf{LP}_{\rightarrow\wedge}$ with $\rightarrow, \wedge$ only into $\boldsymbol{\lambda}^+$. This embedding consists of systematic replacing constants c introduced by constant necessitations rule *R2*: $A \vdash c : A$ with closed $\boldsymbol{\lambda}^{+}$-terms g such that $\boldsymbol{\lambda}^{+}$ proves $A^* \vdash g : A^*$. Here are the key observation that gives a clear idea of how to construct standard closed $\boldsymbol{\lambda}^{+}$-terms that realize axiom necessitation for axioms **A1 - A3** of $\mathbf{LP}_{\to\wedge}$.

A1. Here $A = t{:}F \to F$. Take the $\boldsymbol{\lambda}^{+}$-derivation

$$\frac{x{:}t{:}F \vdash \Downarrow x{:}F}{\vdash \lambda x.\Downarrow x{:}(t{:}F \to F)}$$

Put $c = \lambda x.\Downarrow x$.

A2. Here $A = s{:}(F \to G) \to (t{:}F \to (s \circ t){:}G)$. Take the $\boldsymbol{\lambda}^{+}$-derivation

$$\cfrac{\cfrac{x{:}s{:}(F \to G), y{:}t{:}F \vdash (x \circ^2 y){:}(s \circ t){:}G}{x{:}s{:}(F \to G) \vdash \lambda y.(x \circ^2 y){:}(t{:}F \to (y \circ t){:}G)}}{\vdash \lambda xy.(x \circ^2 y){:}(s{:}(F \to G) \to (t{:}F \to (s \circ t){:}G))}$$

Put $c = \lambda xy.(x \circ^2 y)$.

A3. Here $A = t{:}F \to !t{:}t{:}F$. Take the $\boldsymbol{\lambda}^{+}$-derivation

$$\frac{x{:}t{:}F \vdash \Uparrow x{:}!t{:}t{:}F}{\vdash \lambda x.\Uparrow x{:}(t{:}F \to !t{:}t{:}F)}$$

Put $c = \lambda x.\Uparrow x$.

The exact relationships between $\boldsymbol{\lambda}^{\infty}$ and intuitionistic $\mathbf{LP}_{\to\wedge}$, $\boldsymbol{\lambda}^{\infty}$ and $\boldsymbol{\lambda}^{+}$ is an interesting separate issue. It is clear though that $\mathbf{LP}_{\to\wedge}$ is able to emulate any derivation rule of $\boldsymbol{\lambda}^{\infty}$ and therefore, in a certain sense, both $\boldsymbol{\lambda}^{\infty}$ and $\mathbf{LP}_{\to\wedge}$ can be interpreted in $\boldsymbol{\lambda}^{+}$.

We consider the case of **App** introduction

$$\frac{\Gamma \vdash \vec{s}{:}(A \to B) \quad \Gamma \vdash \vec{t}{:}A}{\Gamma \vdash (s_n \circ^n t_n){:}\ldots(s_1 \circ t_1){:}B}$$

Build the following derivation in $\mathbf{LP}_{\to\wedge}$ (will will use '$\cdot$' for application in $\mathbf{LP}_{\to\wedge}$):

1. $s_1{:}(A \to B) \to (t_1{:}A \to (s_1 \cdot t_1){:}B)$ (axiom **A1**).
2. $c{:}(s_1{:}(A \to B) \to (t_1{:}A \to (s_1 \cdot t_1){:}B))$ (axiom necessitation **R2**)
3. $s_2 : s_1 : (A \to B) \to (t_2 : t_1 : A \to (c \cdot s_2 \cdot t_2) : (s_1 \cdot t_1) : B)$ (from 2., by a standard reasoning in $\mathbf{LP}_{\to\wedge}$)

4. $g_3 : [s_2 : s_1 : (A \to B) \to (t_2 : t_1 : A \to c \cdot s_2 \cdot t_2 : (s_1 \cdot t_1) : B)]$ for some ground proof polynomial g (from 3., by constructive necessitation)

5. $s_3 : s_2 : s_1 : (A \to B) \to (t_3 : t_2 : t_1 : A \to (g_3 \cdot s_3 \cdot t_3) : (c \cdot s_2 \cdot t_2) : (s_1 \cdot t_1) : B)$ (from 4., by a standard reasoning in $\mathbf{LP}_{\to\wedge}$)

6. etc.

This format suggests that
$s_1 \cdot t_1$ corresponds to $s_1 \circ t_1$,
$c \cdot s_2 \cdot t_2$ corresponds to $s_2 \circ^2 t_2$,
$g_3 \cdot s_2 \cdot t_2$ corresponds to $s_3 \circ^3 t_3$,
etc.

$\boldsymbol{\lambda}^+$ does not satisfy the Lifting Lemma, but a slightly weaker form of the Lifting Lemma. If $\boldsymbol{\lambda}^+ \vdash F$, then $\boldsymbol{\lambda}^+ \vdash c : F^{\triangleright}$, for some closed $\boldsymbol{\lambda}^+$-term c, and for $F^{\triangleright} = F$, modulo the equivalence relation induced on $\boldsymbol{\lambda}^+$-terms by the $\boldsymbol{\lambda}^+$-conversions.

4 Acknowledgment

Sergei Artemov suggested the systems $\boldsymbol{\lambda}^\infty$ and $\boldsymbol{\lambda}^+$ and wrote sections 1, 2.1, 3. Jesse Alt made a major contribution to finding the proofs of confluence and strong normalization theorem for $\boldsymbol{\lambda}^\infty$ and wrote sections 2.2, 2.3.

References

[1] S. Artemov, "Operational Modal Logic," *Tech. Rep. MSI 95-29*, Cornell University, December 1995.

[2] S. Artemov, "Uniform provability realization of intuitionistic logic, modality and lambda-terms", Electronic Notes on Theoretical Computer Science, vol. 23, No. 1. 1999. http://www.elsevier.nl/entcs/

[3] S. Artemov, "Explicit provability and constructive semantics", The *Bulletin for Symbolic Logic*, to appear in 2001, v.6, No. 1. `http://www.math.cornell.edu/~artemov/BSL.ps`.

[4] H. Barendregt, "Lambda calculi with types". In *Handbook of Logic in Computer Science, vol.2*, Abramsky, Gabbay, and Maibaum, eds., Oxford University Press, pp. 118-309, 1992

[5] D. de Jongh and G. Japaridze, "Logic of Provability", in S. Buss, ed., *Handbook of Proof Theory*, Elsevier, 1998

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

[7] A.S. Troelstra and H. Schwichtenberg, *Basic Proof Theory*, Cambridge University Press, 1996.