# Proof Realization of Intuitionistic and Modal Logics

Sergei N. Artemov*

Technical Report MSI 96-06, Cornell University, 1996

**Abstract**

Logic of Proofs ($\mathcal{LP}$) has been introduced in [2] as a collection of all valid formulas in the propositional language with labeled logical connectives $[\![t]\!](\cdot)$ where $t$ is a *proof term* with the intended reading of $[\![t]\!]F$ as "*$t$ is a proof of $F$*". $\mathcal{LP}$ is supplied with a natural axiom system, completeness and decidability theorems. $\mathcal{LP}$ may express some constructions of logic which have been formulated or/and interpreted in an informal metalanguage involving the notion of proof, e.g. the intuitionistic logic and its Brauwer-Heyting-Kolmogorov semantics, classical modal logic $\mathcal{S}4$, *etc* (cf. [2]). In the current paper we demonstrate how the intuitionistic propositional logic $\mathcal{I}nt$ can be directily realized into the Logic of Proofs. It is shown, that the proof realizability gives a fair semantics for $\mathcal{I}nt$.

## 1 Introduction

Logic of Proofs ($\mathcal{LP}$) incorporates proof terms directly into the propositional language using new atomic formulas $[\![t]\!]F$ with the intended reading "*$t$ is a proof of $F$*" (cf. [2]). A functional completeness theorem from [2] (cf. Section 5 below) demonstrates, that three basic operations on proofs: *application, proof checker*, and *choice* constitute a basis for all operations on proofs, expressible in the propositional language with the labeled connectives of the type $[\![term]\!](formula)$. These three operations are explicitely incorporated into $\mathcal{LP}$. The language of $\mathcal{LP}$ has an exact intended semantics, where "$t$ is a proof of $F$" is interpreted as a corresponding arithmetical formula about the codes of $t$ and $F$. The decidability of $\mathcal{LP}$ was established in ([2]), along with the completeness of a natural axiom system for $\mathcal{LP}$.

The intuitionistic logic $\mathcal{I}nt$ was supplied ([7], [8], cf.[11], [5], [12]) with an informal Brouwer-Heyting-Kolmogorov (*BHK*) operational semantics , which was given in terms of logical conditions on the formulas and their proofs, *e.g.* the "implication" clause is "$p$ proves $A \to B$ iff $p$ is a construction transforming any proof $c$ of $A$ into a proof $p(c)$ of $B$". In 1933 Gödel made a step to formalize *BHK* semantics by introducing a faithfull embedding of $\mathcal{I}nt$

*Department of Mathematics, Cornell University , Ithaca NY, 14853 `email:sergei@artemov.mian.su`; Steklov Mathematical Institute, Russian Academy of Sciences, 42 Vavilova str., Moscow 117966, Russia

into a "natural born" provability logic $\mathcal{S}4$; this attempt has remained incomplete, since, in turn, $\mathcal{S}4$ has lacked the intended provability semantics.

$$\mathcal{I}nt \quad \hookrightarrow \quad \mathcal{S}4 \quad \hookrightarrow \; ?$$

As it was also established in [2], an immediate forgetful translation of $\mathcal{LP}$ gives exactly $\mathcal{S}4$; in particular, there is a realization algorithm recovering $\mathcal{LP}$-terms in any $\mathcal{S}4$-proof. So, $\mathcal{LP}$ provides an intended provability interpretation for the modal logic $\mathcal{S}4$

$$\mathcal{S}4 \quad \hookrightarrow \quad \mathcal{LP} \quad \hookrightarrow \quad \mathit{Arithmetic},$$

thus completing the Gödels embedding of $\mathcal{I}nt$ into $\mathcal{S}4$ to the fair arithmetical provability semantics for both $\mathcal{I}nt$ and $\mathcal{S}4$

$$\mathcal{I}nt \quad \hookrightarrow \quad \mathcal{S}4 \quad \hookrightarrow \quad \mathcal{LP} \quad \hookrightarrow \quad \mathit{Arithmetic}.$$

In the current paper we give a direct realization algorithm of $\mathcal{I}nt$ into $\mathcal{LP}$, Gödel style. This proof realizability provides a fair semantics for $\mathcal{I}nt$:

$$\mathcal{I}nt \vdash F \quad \Leftrightarrow \quad F \textit{ is proof realizable.}$$

## 2 Logic of Proofs

The language of $\mathcal{LP}$ contains

boolean constants $\top, \bot$, sentence variables $p_0, \dots, p_n, \dots$
proof variables $x_0, \dots, x_n, \dots$
proof axiom constants $a_0, \dots, a_n, \dots$
boolean connectives $\rightarrow, \dots$
functional symbols: monadic $!$, binary $+$ and $\times$
operator symbol $[\![\mathit{term}]\!]\ (\mathit{formula})$.

Terms and formulas are defined in a natural way: a proof variable and an axiom constant is a term; a sentence variable and a boolean constant is a formula; whenever $s, t$ are terms $!t, (s+t), (s\times t)$ are again terms, boolean connectives behave conventionally, and for $t$ a term and $F$ a formula $[\![t]\!]F$ is a formula. A term is *ground* if is does not contain variables.

We will write $s{\cdot}t$ or even $st$ instead of $(s \times t)$ and skip parentheses when convenient. If $\vec{x} = (x_1, \dots, x_n)$ and $\Gamma = (A_1, \dots, A_n)$, then we will write $[\![\vec{x}]\!]\Gamma$ for $[\![x_1]\!]A_1, \dots, [\![x_n]\!]A_n$.

**2.1 Definition.** System $\mathcal{LP}_{AS}$. The axioms are all formulas of the form

Ac. Axiom schemes of classical propositional logic in the language of $\mathcal{LP}$
A1. $[\![t]\!]F \rightarrow F$ "reflexivity"
A2. $[\![t]\!](F \rightarrow G) \ \rightarrow ([\![s]\!]F \rightarrow [\![ts]\!]G)$ "application"
A3. $[\![t]\!]F \ \rightarrow \ [\![!t]\!][\![t]\!]F$ "proof checker"
A4. $[\![s]\!]F \ \rightarrow \ [\![s{+}t]\!]F, \qquad [\![t]\!]F \ \rightarrow \ [\![s{+}t]\!]F$ "choice"

AS. A finite set of formulas of the form $[\![c]\!]A$,
where $c$ is an axiom constant, and $A$ is an axiom Ac-A4 "axiom specification"

Rule: *modus ponens.*

System $\mathcal{LP}$ is the union of $\mathcal{LP}_{AS}$'s for all axiom specifications AS.

The intended understanding of $\mathcal{LP}$ is as a logic of operations on proofs, where $[\![t]\!]F$ stands for

*"$t$ is a code for a proof of $F$".*

For the usual Gödel proof predicate $Proof(x,y)$ in $\mathcal{PA}$ there are primitive recursive functions from codes of proofs to codes of proofs corresponding to "$\times$" and "!": "$\times$" stands for a operation on proof sequences which realizes the *modus ponens* rule in arithmetic, and "!" is a "proof checker" operation, appearing in the proof of the second Gödel Incompleteness theorem. The usual proof predicate has a natural nondeterministic version $PROOF(x,y)$ called *standard nondeterministic proof predicate*

*"$x$ is a code of a derivation containing a formula with a code $y$".*

$PROOF$ already has all three operations of the $\mathcal{LP}$-language: the operation $s+t$ is now just a concatenation of (nondeterministic) proofs $s$ and $t$.

**2.2 Comment.** System $\mathcal{LP}$ is not a multimodal logic, since no single modality $[\![t]\!](\cdot)$ satisfies the property $[\![t]\!](p \to q) \to ([\![t]\!]p \to [\![t]\!]q)$ in $\mathcal{LP}$. This makes $\mathcal{LP}$ different from numerous multimodal logics. However, the entire variety of labeled modalities in $\mathcal{LP}$ can emulate $\mathcal{S}4$([2], cf. Theorem 3.4).

**2.3 Comment.** The usual deduction theorem holds for $\mathcal{LP}$:

$$\Gamma, A \vdash_{\mathcal{LP}} B \quad \Rightarrow \quad \Gamma \vdash_{\mathcal{LP}} A \to B.$$

**2.4 Lemma.** (Substitution lemma for $\mathcal{LP}$). *If $\Gamma(x,p) \vdash_{\mathcal{LP}} B(x,p)$ for a propositional varaiable $p$ and a proof variable $x$, then for any proof term $t$ and any formula $F$*

$$\Gamma(x/t, p/F) \vdash_{\mathcal{LP}} B(x/t, p/F).$$

**Proof** is trivial, since all axioms and rules of $\mathcal{LP}$ remain axioms and rules after a substitution.

**2.5 Lemma.** *The following rules are admissible in $\mathcal{LP}$. Here $A$,$B$ are $\mathcal{LP}$-formulas, $\Gamma, \Delta$ are finite sets of $\mathcal{LP}$-formulas, $y$ is a proof variable, $t, r$ are proof terms, $\vec{y}$ and $\vec{s}$ are vectors of proof variables and proof terms correspondingly, "$\vdash$" means "$\vdash_{\mathcal{LP}}$".*

*Dynamic Necessitation:* $$\frac{\vdash B}{\vdash [\![t]\!]B}$$ *for some ground $t$;*

*Lifting:* $$\frac{[\![\vec{s}]\!]\Gamma, \Delta \vdash B}{[\![\vec{s}]\!]\Gamma, [\![\vec{y}]\!]\Delta \vdash [\![t(\vec{y})]\!]B}$$ *for some $t(\vec{y})$;*

*Lowering:* $$\frac{\Gamma, [\![\vec{y}]\!]\Delta \vdash [\![t]\!]B}{\Gamma, \Delta \vdash B}\ ;$$
*($\vec{y}$ does not occur in the conclusion)*

*Abstraction:* $$\frac{[\![\vec{s}]\!]\Gamma, [\![y]\!]A \vdash [\![t(y)]\!]B}{[\![\vec{s}]\!]\Gamma \vdash [\![\lambda y.t(y)]\!](A \rightarrow B)}$$
*for some proof term denoted as $\lambda y.t(y)$*
*($y$ does not occur in the conclusion.)*

**Proof.** *Dynamic Necessitation* is a special case of *Lifting*.

*Lifting.* By induction on a proof of $B$ from the premises $[\![\vec{s}]\!]\Gamma, \Delta$. If $B \in [\![\vec{s}]\!]\Gamma$, then $[\![\vec{s}]\!]\Gamma, [\![\vec{y}]\!]\Delta \vdash [\![!s_i]\!]B$ for some $s_i \in \vec{s}$. If $B \in \Delta$, then $[\![y_j]\!]B \in [\![\vec{y}]\!]\Delta$. for some $y_j \in \vec{y}$. If $B$ is an axiom Ac $-$ A4, then $[\![c]\!]B$ is an axiom AS. If $B$ is from A5, *i.e.* $B$ is $[\![c]\!]A$ for some $A$ from Ac $-$ A4, then by A3, $\vdash [\![c]\!]A \rightarrow [\![!c]\!][\![c]\!]A$, and $[\![\vec{s}]\!]\Gamma, [\![\vec{y}]\!]\Delta \vdash [\![!c]\!]B$. Let $B$ be obtained from $C$,$C \rightarrow B$ by *modus ponens*. Then, by the induction hypothesis, $[\![\vec{s}]\!]\Gamma, [\![\vec{y}]\!]\Delta \vdash [\![t_1(\vec{y})]\!](C \rightarrow B)$ and $[\![\vec{s}]\!]\Gamma, [\![\vec{y}]\!]\Delta \vdash [\![t_2(\vec{y})]\!]C$. for some terms $t_1$ and $t_2$. By A2, $[\![\vec{x}]\!]\Gamma, [\![\vec{y}]\!]\Delta \vdash [\![t_1 \cdot t_2]\!]B$

*Lowering.* By the arithmetical completeness theorem for $\mathcal{LP}$ ([2] or Theorem 4.1 of the current paper), it suffices to show that for any arithmetical interpretation $*$ if $\models \Gamma^*$ and $\models \Delta^*$, then $\models B^*$. Suppose $\models \Gamma^*$ and $\models \Delta^*$; without loss of generality we assume that for each $A_j \in \Delta$ the arithmetical formula $A_j^*$ is provably $\Delta_1$, any such $A_j^*$ is provable in Peano Arithmetic $\mathcal{PA}$, and $\models Prf(k_j, \ulcorner A_j^* \urcorner)$ for some natural number $k_j$. Since $\vec{y}$ does not occur in $\Gamma, \Delta, B$ we may restrict ourselves to the Axiom Specifications which do not contain $\vec{y}$ either. Indeed, if there were a proof $\Gamma, \Delta \vdash B$ from the Axiom Specification set AS containing $\vec{y}$, then after the substitution $[y_j/c]$ for any proof constant $c$ we would get a $\vec{y}$-free proof $\Gamma, \Delta \vdash B$.

We define a new arithmetical interpretation $*_1$ upgrading $*$ by the evaluation $y_j^{*_1} := k_j$. Now $\Gamma^*$ coincides with $\Gamma^{*_1}$, and thus $\models \Gamma^{*_1}$. By the definition of $*_1$, $\models ([\![\vec{y}]\!]\Delta)^{*_1}$. By the arithmetical correctness of $\mathcal{LP}$, from the premise $\Gamma, [\![\vec{y}]\!]\Delta \vdash [\![t]\!]B$ we conclude $\models ([\![t]\!]B)^{*_1}$, and thus $\models B^{*_1}$. Again, we notice that $B^{*_1}$ coincides with $B^*$, and get the desired $B^*$.

*Abstraction.* From $[\![\vec{s}]\!]\Gamma, [\![y]\!]A \vdash [\![t(y)]\!]B$ by Lowering, get $[\![\vec{s}]\!]\Gamma, A \vdash B$, then by Deduction, $[\![\vec{s}]\!]\Gamma \vdash A \rightarrow B$, and then use Lifting to get $[\![\vec{s}]\!]\Gamma \vdash [\![r]\!](A \rightarrow B)$ for some proof term $r$.
◀

**2.6 Comment.** A term $t(\vec{y})$ introduced by the Lifting rule is nothing but a protocol of a proof of $B$ from $[\![\vec{s}]\!]\Gamma, [\![\vec{y}]\!]\Delta$. The same holds for the rule of Abstraction, where $\lambda y.t(y)$ is a protocol of a proof of $A \to B$ from $[\![\vec{s}]\!]\Gamma$.

The Lowering rule is the only rule in this list which does not introduce a proof term. Also, the proof of this rule does not look constructive. However, since $\mathcal{LP}$ is decidable there is a (primitive recursive) procedure, which constructs a proof from the conclusion given a proof from the premise. A more direct algorithm could perhaps be found after developing some basic proof theory for $\mathcal{LP}$.

The Abstraction rule might not look like an operation on terms either, because in the the process of constructing $\lambda y.t(y)$ from $t(y)$ we get rid of the latter and seemingly construct $\lambda y.t(y)$ from the scratch. However, it is not the case. A term $t(y)$ is a protocol of a proof of $B$ from $[\![\vec{s}]\!]\Gamma, [\![y]\!]A$. From this proof we get a proof $[\![\vec{s}]\!]\Gamma, A \vdash B$, then a proof of $A \to B$ from $[\![\vec{s}]\!]\Gamma$. Finally, $\lambda y.t(y)$ is a protocol of the latter proof. All the procedures from this chain of transformations leading from $t(y)$ to $\lambda y.t(y)$ are constructive; not all of them can be formalized in $\mathcal{LP}$, but it is a small prise we pay for our intension to keep the basic language for the Logic of Proofs as simple a possible.

## 3 Realization of $\mathcal{S}4$ in $\mathcal{LP}$.

**3.1 Example.** $\mathcal{S}4 \vdash (\Box A \wedge \Box B) \to \Box(A \wedge B)$. In $\mathcal{LP}$ this can be reproduced by the following:

1. $A, B \vdash A \wedge B$
2. $[\![x]\!]A, [\![y]\!]B \vdash [\![t(x,y)]\!](A \wedge B)$, from 1. by Lifting
3. $[\![x]\!]A \wedge [\![y]\!]B \vdash [\![t(x,y)]\!](A \wedge B)$
4. $\vdash ([\![x]\!]A \wedge [\![y]\!]B) \to [\![t(x,y)]\!](A \wedge B)$

In fact, here $t(x,y)$ can be taken $(cx)y$, where $[\![c]\!](A \to (B \to (A \wedge B)))$ is an axiom AS.

**3.2 Example.** $\mathcal{S}4 \vdash (\Box A \vee \Box B) \to \Box(A \vee B)$. In $\mathcal{LP}$ the corresponding derivation is

1. $A \vdash A \vee B$
2. $B \vdash A \vee B$
3. $[\![x]\!]A \vdash [\![t(x)]\!](A \vee B)$ by Lifting from 1
4. $[\![y]\!]B \vdash [\![s(y)]\!](A \vee B)$ by Lifting from 2
5. $[\![x]\!]A \vdash [\![t(x)+s(y)]\!](A \vee B)$, $[\![y]\!]B \vdash [\![t(x)+s(y)]\!](A \vee B)$ by A4 from 3, 4.
6. $[\![x]\!]A \vee [\![y]\!]B \vdash [\![t(x)+s(y)]\!](A \vee B)$
7. $\vdash [\![x]\!]A \vee [\![y]\!]B) \to [\![t(x)+s(y)]\!](A \vee B)$

The fundamental fact about $\mathcal{S}4$ is that, **all** $\mathcal{S}4$-theorems have a corresponding operational reading in $\mathcal{LP}$.

**3.3 Definition.** By an *$\mathcal{LP}$-realization* $r = r(AS)$ of a modal formula $F$ we mean

1. an assignment of $\mathcal{LP}$-terms to all occurrences of the modality in $F$,
2. a choice of an axiom specification AS;

Under $F^r$ we denote the image of $F$ under a realization $r$. Positive and negative occurrences of modality in a formula and a sequent are defined in the usual way. A realization $r$ is *normal* if all negative occurrences of $\Box$ are realized by proof variables.

**3.4 Theorem.** ([2]) *If $\mathcal{S}4 \vdash F$, then $\mathcal{LP}_{AS} \vdash F^r$ for some axiom specification AS and some normal realization $r = r(AS)$.*

The proof describes an algorithm which for a given cut-free derivation $\mathcal{T}$ in $\mathcal{S}4$ assigns $\mathcal{LP}$ terms to all occurrences of the modality in $\mathcal{T}$.

**3.5 Corollary.**

$$\mathcal{S}4 \vdash F \quad \Leftrightarrow \quad \mathcal{LP} \vdash F^r \textit{ for some realization } r.$$

# 4 Arithmetical Semantics

Let us agree to use a new functional symbol $\iota z\varphi(z)$ for any arithmetical formula $\varphi(z)$ and assume that $\iota$-terms could be eliminated in the usual way by using the small scope convention (cf. [4]). An arithmetical formula $\varphi$ is *provably* $\Delta_1$ iff both $\varphi$ and $\neg\varphi$ are provably $\Sigma_1$. A term $\iota z\varphi$ is *provably recursive* iff $\varphi$ is provably $\Sigma_1$. *Closed recursive term* is a provably total and provably recursive term $\iota z\varphi$ such that $\varphi$ contains no free variables other than $z$. Close recursive terms represent all provably recursive names for natural numbers. We have to use all of them as proof realizers, since some operations on proofs, *e.g.* the *proof checker* "!", depend on the name of the argument, not on its value. Indeed, if $PROOF(\overline{n}, \overline{k})$ holds, then $PROOF(\overline{n}+0, \overline{k})$ also holds, $!(\overline{n})$ is a proof of $PROOF(\overline{n}, \overline{k})$ and $!(\overline{n}+0)$ is a proof of $PROOF(\overline{n}+0, \overline{k})$. However, $!(\overline{n})$ and $!(\overline{n}+0)$ deliver proofs of different formulas, thus, generally speaking, $!(\overline{n}) \neq !(\overline{n}+0)$.

A *proof predicate* is a provably $\Delta_1$-formula $Prf(x, y)$ such that for all $\varphi$

$$\mathcal{PA} \vdash \varphi \quad \Leftrightarrow \quad \text{for some } n \in \omega \quad Prf(n, \ulcorner\varphi\urcorner) \text{ holds.}$$

A proof predicate *Prf(x,y)* is *normal* if

1) for every proof $k$ the set $T(k) = \{l \mid Prf(k, l)\}$ is finite and the function

$$\widetilde{T(k)} = \textit{the code of } T(k)$$

is provably recursive,

2) for every finite set $S$ of theorems of $\mathcal{PA}$, $S \subseteq T(k)$ for some proof $k$.

The nondeterministic proof predicate *PROOF* (above) is a normal proof predicate.

For every normal proof predicate *Prf* there are provably recursive terms $m(x, y)$, $a(x, y)$, $c(x)$ such that for all closed recursive terms $s, t$ and for all arithmetical formulas $\varphi, \psi$ the following formulas are valid:

$$Prf(s, \ulcorner \varphi \to \psi \urcorner) \wedge Prf(t, \ulcorner \varphi \urcorner) \to Prf(m(s, t), \ulcorner \psi \urcorner)$$

$$Prf(s, \ulcorner \varphi \urcorner) \to Prf(a(s, t), \ulcorner \varphi \urcorner), \qquad Prf(t, \ulcorner \varphi \urcorner) \to Prf(a(s, t), \ulcorner \varphi \urcorner)$$

$$Prf(t, \ulcorner \varphi \urcorner) \to Prf(c(\ulcorner t \urcorner), \ulcorner Prf(t, \ulcorner \varphi \urcorner) \urcorner).$$

Let AS be an axiom specification. An arithmetical *AS-interpretation* $*$ of $\mathcal{LP}$-language has the following parameters: AS, a normal proof predicate *Prf*, an evaluation of sentence letters by sentences of arithmetic, an evaluation of proof letters and axiom constants by closed recursive terms. We put $\top^* \equiv (0 = 0)$ and $\bot^* \equiv (0 = 1)$, $*$ commute with boolean connectives, $(t \cdot s)^* \equiv m(t^*, s^*)$, $\quad (t + s)^* \equiv a(t^*, s^*)$, $\quad (!t)^* \equiv c(\ulcorner t^* \urcorner)$, $([\![t]\!]F)^* \equiv Prf(t^*, \ulcorner F^* \urcorner)$. We also assume, that $\mathcal{PA} \vdash G^*$ for all $G \in AS$.

Under any *AS-interpretation* $*$ an $\mathcal{LP}$-term $t$ becomes a closed recursive term $t^*$ (*i.e.* a recursive name of a natural number), and an $\mathcal{LP}$-formula $F$ becomes an arithmetical sentence $F^*$. In what follows "arithmetically AS-valid" means either "provable in $\mathcal{PA}$" or "true in the standard modal" under any AS-interpretation.

Note that the reflexivity principle is back, since $[\![t]\!]F \to F$ is provable in $\mathcal{PA}$ under any interpretation $*$. Indeed, let $n$ be the value of $t^*$. If $Prf(n, \ulcorner F^* \urcorner)$ is true, then $\mathcal{PA} \vdash F^*$, thus $\mathcal{PA} \vdash Prf(n, \ulcorner F^* \urcorner) \to F^*$. If $Prf(n, \ulcorner F^* \urcorner)$ is false, then $\mathcal{PA} \vdash \neg Prf(n, \ulcorner F^* \urcorner)$, and again $\mathcal{PA} \vdash Prf(n, \ulcorner F^* \urcorner) \to F^*$.

**4.1 Theorem.** ([2], Arithmetical completeness of $\mathcal{LP}$)

$$\mathcal{LP}_{AS} \vdash F \quad \Leftrightarrow \quad F^* \textit{ is arithmetically AS-valid .}$$

Combining 3.4 and 4.1, we obtain the arithmetical completeness of $\mathcal{S}4$:

$$\mathcal{S}4 \vdash F \quad \Leftrightarrow \quad F^r \textit{ is arithmetically AS-valid for some realization } r \textit{ and some axiom specification AS.}$$

Gödel in [6] defined a translation *tr* of intuitionistic formulas, into $\mathcal{S}4$-formulas where *tr(F)* is obtained from $F$ by boxing all atoms and all implications in $F$. This Gödel translation is shown ([6], [9]) to provide a faithful embedding of $\mathcal{I}nt$ into $\mathcal{S}4$. The proof interpretation of $\mathcal{LP}$-terms above provides a faithful proof arithmetical realization of $\mathcal{I}nt$:

$$\mathcal{I}nt \vdash F \quad \Leftrightarrow \quad [tr(F)]^r \textit{ is arithmetically AS-valid for some normal realization } r \textit{ and some axiom specification AS.}$$

## 5 Functional completeness

It is proven in [2] that $\mathcal{LP}$ describes *all* possible propositional operations on proofs. The basic operations $\times, !, +$ thus play for proofs a role similar to that boolean connectives play for classical logic.

Consider an arbitrary scheme of a specification of an operation of proofs in arithmetic. Such a specification is an arithmetical formula

$$\forall \vec{x} \in C\ \exists y \text{ “}y \textit{ is a proof of } G(\vec{x})\text{”},$$

or, equivalently

$$\forall \vec{x}(C(\vec{x}) \ \rightarrow\ \exists y \text{ “}y \textit{ is a proof of } G(\vec{x})\text{”}),$$

true in the standard model of arithmetic, where $C$ and $G$ are arbitrary arithmetical conditions. Now we restrict $C$ and $G$ by an $\mathcal{LP}$-language without functions, which will play a role of a *specification language*. The only proof terms in the specification language are the proof variables. Now it is a reasonable question to ask

*what operations on proofs can be specified in the logic of proofs?*

Now we can make $C(\vec{x})$ and $G(\vec{x})$ conditions in the specification language. Also, we express the existential quantifier $\exists y$ *“y is a proof of* $G(\vec{x})$*”* by the usual provability modality $\Box$, extending the definition of $F^*$ by one more item : $(\Box F)^*$ is $Pr(\ulcorner F^* \urcorner)$.

Finally we restrict $C$ to a “*proof positive*” condition, *i.e.* one where the outermost q-atomic subformulas are positive in $C$.

**5.1 Comment.** Indeed, a condition of the sort

$$\neg [\![x]\!]\ P \ \rightarrow\ \Box\neg [\![x]\!] P,$$

although valid for any proof predicate, may hardly be accepted as a specification of an operation on proofs equally as good as $\times, !, +$, because it derives conclusions from *negative* information about proofs; here, from “*x IS NOT a proof*”. The presentation of the negative information in the logic of proofs remains a challenging open problem.

It seems that now we have found a balanced definition of an operation on proofs. The regular case

$$[\![x_1]\!] C_1 \wedge \ldots \wedge [\![x_n]\!] C_n \rightarrow \Box G,$$

which comes from the straightforward formalization of the notion of an admissible inference rule

$$\frac{C_1, \ldots, C_n}{G}$$

is covered. Further shrinking of $C$ to say conjunctions of q-atomic formulas would eliminate natural and useful nondeterministic proof systems.

**5.2 Definition.** We may define now an *abstract propositional operation on proofs* as a formula $C \rightarrow \Box G$, valid under all arithmetical interpretations, where $C, G$ are formulas in the specification language and $C$ is proof positive.

**5.3 Comment.** Operations $\times, !, +$ can be identified as abstract propositional operations on proofs. Indeed, formulas

$[\![x_1]\!](F \rightarrow G) \wedge [\![x_2]\!]\ F \rightarrow \Box\ G$

$[\![x]\!]F \rightarrow \Box[\![x]\!]\ F$

$[\![x_1]\!]F \vee [\![x_2]\!]F \rightarrow \Box F$

are valid under every arithmetical translation and Skolem functions for the existential quantifiers on proofs in $\Box$'s here can be realized by $m(x_1, x_2)$, $c(x)$, $a(x_1, x_2)$ correspondingly.

The following theorem ([2]) demonstrates that $\mathcal{LP}$-terms suffice to realize any propositional operation on proofs.

**5.4 Theorem.** ([2]) *For any abstract propositional operation on proofs $C \rightarrow \Box G$ there exists an $\mathcal{LP}$-term $t$ such that*

$$\mathcal{LP} \vdash C \rightarrow [\![t]\!]G.$$

# 6 Logic of Proofs *vrs* Provability Logic.

The Logic of Proofs gives a formalization of the arithmetical provability operator different from the one of the Provability Logic. In a certain sense, the Logic of Proofs introduces a new propositional language which is taylored to get rid of the hidden quantifiers on proofs. The indended interpretation of a formula of the $\mathcal{LP}$-language gives provably decidable arithmetical sentence, provided the evaluations of the propositions are. As a result, there is do direct way to interpret the Second Gödel Incompletness theorem into $\mathcal{LP}$. The fixed point construction which establishes the arithmetical completeness of $\mathcal{LP}$([2]) is totally different from the one used by R. Solovay is his proof of the arithmetical completeness of the provability logic (cf. [3]). However, the arithmetical interpretations of the Logic of Proofs and the Provability Logic are clearly compatible; in [1] in the proof of the arithmetical completenes of the system $\mathcal{B}$ it was shown how to build the arithmetical fixed point for the Logic of Proofs (without operations) in the top of the Solovay fixed point. The fixed point construction from [2] is a dynamic version of the one from [1]. This gives a pretty clear idea how to prove the arithmetical completeness of the logic in the language containing both the provability operator $\Box$, and the proof operators $[\![t]\!]$'s with the operations on the proof terms. However, there is a creative portion of work to be done here: one has to find a decent set of new operation on proofs which handle the modality $\Box$ properly.

# 7 Logic of Proofs *vrs* Modal Logic.

By 3.4, $\mathcal{LP}$ is a version of $\mathcal{S}4$ presented in a more rich operational language, with no information being lost, since $\mathcal{S}4$ is the the exact term-forgetting projection of $\mathcal{LP}$. An easy inspection of the realizing algorithm shows that

$$\mathcal{LP}\text{-fomula} \quad = \quad \mathcal{S}4\text{-formula} + \text{its } \mathcal{S}4\text{-proof.}$$

A transliretating of an $\mathcal{S}4$-theorem into $\mathcal{LP}$-language may result in an exponential growth of its length. However, this increase looks much less dramatic if we calculate the complexity of the input $\mathcal{S}4$-theorem $F$ in an "honest" way as the length of a proof of $F$ in $\mathcal{S}4$: the proof terms appearing in the realization algorithm have a size linear of the length of the proof, so, the total length of an $\mathcal{LP}$-realization of an $\mathcal{S}4$-formula $F$ is bounded by the quadratic function of the length of a given $\mathcal{S}4$-proof of $F$.

The decomposition of the $\mathcal{S}4$-modality into a finitely generated set of terms in $\mathcal{LP}$ above is a general fact, which may be used in other applications of the modal logic.

# 8 Logic of Proofs *vrs* Intuitionistic logic.

Kleene realizability (cf. [11]) of the intuitionistic language does not use the logical provability constraints from the original *BHK* formulation and referes to all recursive functions, not just operations on proofs. As a result, too many formulas become realizable, more than $\mathcal{I}nt$ can derive:

$$\mathcal{I}nt \quad \subsetneq \quad \textit{Kleene realizable formulas}^1.$$

Proof realizability of $\mathcal{I}nt$ can be defined as a superposition of the realizations of $\mathcal{S}4$ in $\mathcal{LP}$ and $\mathcal{LP}$ in the arithmetic (above); $\mathcal{I}nt$ turns out to be complete with respect to the proof realizability

$$\mathcal{I}nt \quad = \quad \textit{proof realizable formulas.}$$

In addition to the general algorithm of realization of $\mathcal{S}4$ in $\mathcal{LP}$ (3.4), we describe now its "light" version, which realizes $\mathcal{I}nt$ in $\mathcal{LP}$ directly.

We assume, that $\mathcal{I}nt$ is presented in the language with $\{\wedge, \vee, \rightarrow, \perp\}$ and recall, that the Gödel translation of an $\mathcal{I}nt$-formula $F$ into a $\mathcal{S}4$-formula *tr(F)* consists in prefixing all subformulas in $F$ by $\Box$ (we agree to skip $\Box$ prefixes of $\perp$). Our realization algorithm extends this Gödel translation to $\mathcal{LP}$-formulas.

Step 1. Take a sequential cut-free derivation of $F$ in $\mathcal{I}nt$ with the axioms $p \Rightarrow p$, where $p$ is a propositional variable, and $\perp \Rightarrow$ . Replace every formula $G$ in this derivation by its Gödel translation *tr(G)*. The resulting tree $\mathcal{T}$ is an "almost" $\mathcal{S}4$-derivation of *tr(F)* with the axioms of the form $\Box p \Rightarrow \Box p$ with $p$ a propositional letter. More precisely, every $\mathcal{S}4$-sequent in $\mathcal{T}$ is provable in $\mathcal{S}4$; moreover, each step down in $\mathcal{T}$ can be regarded as a corresponding

[1] Unless a metatheory is restricted, *e.g.* by $\mathcal{HA}$ like in the Troelstra - Plisko theorem [10].

standard combination of $\mathcal{S}4$-rules, excluding *Cut*. All the following steps are an adoptation of the general realizing algorithm of $\mathcal{S}4$ into $\mathcal{LP}$ for $\mathcal{T}$.

Occurrences of $\Box$ in $\mathcal{T}$ are related if they occur in related formulas in premises and conclusions of nodes in $\mathcal{T}$; we extend this relationship by transitivity. All occurrences of $\Box$ in $\mathcal{T}$ are now naturally split into disjoint *families* of related ones. Since polarities of the $\Box$'s are respected in $\mathcal{T}$ we may speak about negative and positive families of related $\Box$'s. Two families are *close* if they contain $\Box$'s from an axiom $\Box p \Rightarrow \Box p$. We call a positive family *essential* if it contains at least one $\Box$ introduced by the $(\Rightarrow \Box)$ rule. In the tree $\mathcal{T}$, essential $\Box$'s appear only at the nodes, corresponding to the rules of introduction to the succedent. Since all $\Box$'s at the axiom nodes of $\mathcal{T}$ correspond to the atomic formulas, there are no essential $\Box$'s at leaves (axiom nodes). The basic observation here is that *no negative family is close to an essential positive family.*

Step 2. Realize each negative family and each nonessential positive family by a fresh proof variable, realize close families by the same proof variable.

Step 3. For every essential positive family $f$ enumerate all the nodes where the principal $\Box$ has been introduced, and let $n_f$ be the total number of such nodes for a family $f$. Realize all $\Box$'s in an essential positive family $f$ by the term

$$(u_1 + \ldots + u_{n_f}),$$

where $u_i$'s are fresh proof variables, which we call *provisional* variables. The resulting tree is called an *evaluated tree.*

Step 4. Perform the following leaves-root procedure of replacing provisional variables by proof terms, which will result in the desired realization $r$. After this procedure passes a node and perform corresponding changes of the labelling sequent $\Gamma \Rightarrow \Delta$, we will have

$$\Gamma \vdash_{\mathcal{LP}} \Delta. \qquad (\dagger)$$

The case of an axiom node $\Box p \Rightarrow \Box p$ in $\mathcal{T}$, is trivial, since the corresponding $\mathcal{LP}$-realization is $[\![x]\!]p \Rightarrow [\![x]\!]p$ for some proof variable $x$.

At the nodes of the evaluated tree, corresponding to the introduction to the antecedent rules, we don't perform any substitutions. It is an easy exercise in a propositional logic to verify, that the property $(\dagger)$ is respected.

A $(\Rightarrow\rightarrow)$ node in the evaluated tree looks like

$$\frac{[\![y]\!]A, [\![\vec{x}]\!]\Gamma \Rightarrow [\![s]\!]B}{[\![\vec{x}]\!]\Gamma \Rightarrow [\![t_1 + \ldots + u_i + \ldots + t_{n_f}]\!]([\![y]\!]A \rightarrow [\![s]\!]B)},$$

where $u_i$ is a provisional variable, corresponding to this particular node. By the induction hypothesis,

$$[\![y]\!]A, [\![\vec{x}]\!]\Gamma \vdash_{\mathcal{LP}} [\![s]\!]B.$$

By the Deduction rule for $\mathcal{LP}$,

$$[\![\vec{x}]\!]\Gamma \vdash_{\mathcal{LP}} [\![y]\!]A \to [\![s]\!]B,$$

and by Lifting, there exists a proof term $t(\vec{x})$ such that

$$[\![\vec{x}]\!]\Gamma \vdash_{\mathcal{LP}} [\![t(\vec{x})]\!]([\![y]\!]A \to [\![s]\!]B).$$

Substitute everywhere in the tree $t(\vec{x})$ for $u_i$. Since $t(\vec{x})$ does not contain provisional variables, $u_i$ is no longer present in the evaluated tree. By the substitution lemma the property (†) survives for all sequents in the tree. Clearly,

$$[\![\vec{x}]\!]\Gamma \vdash_{\mathcal{LP}} [\![t_1{+}\ldots{+}t{+}\ldots{+}t_{n_f}]\!]([\![y]\!]A \to [\![s]\!]B).$$

The remaining cases of $(\Rightarrow \wedge)$-nodes and $(\Rightarrow \vee)$-nodes are treated similarly. After the process reaches the root, no provisional variables remain in the tree, the assignment of proof terms to the $\Box$'s in the root sequent is the desired realization of this sequent in $\mathcal{LP}$.

Since an $\mathcal{I}nt$-formula $\varphi$ may be identified with the sequent $\Rightarrow \varphi$, we may define a realizer of $\varphi$ as a ground proof term $r$ realizing the sequent $\Rightarrow \varphi$; the resulting evaluated tree will then have the root $\Rightarrow [\![r]\!]\tilde{\varphi}$ for some $\mathcal{LP}$-formula $\tilde{\varphi}$. This $r$ is a protocol of the derivation of $\Rightarrow \tilde{\varphi}$.

The completeness theorem for proof realizations

$$\varphi \ \textit{ is provable in } \mathcal{I}nt \quad \Leftrightarrow \quad \varphi \ \textit{ is proof realizable}$$

follows now from the fairness of the embeddings

$$\mathcal{I}nt \ \hookrightarrow \ \mathcal{S}4 \ \hookrightarrow \ \mathcal{LP} \ \hookrightarrow \ \textit{Arithmetic}.$$

**8.1 Example.** The $\mathcal{I}nt$-derivation

$$\dfrac{\dfrac{\dfrac{\dfrac{A \Rightarrow A}{A \Rightarrow A\vee B} \quad \bot \Rightarrow \bot}{\neg(A\vee B), A \Rightarrow \bot}}{\neg(A\vee B) \Rightarrow \neg A} \qquad \dfrac{\dfrac{\dfrac{B \Rightarrow B}{B \Rightarrow A\vee B} \quad \bot \Rightarrow \bot}{\neg(A\vee B), B \Rightarrow \bot}}{\neg(A\vee B) \Rightarrow \neg B}}{\neg(A\vee B) \Rightarrow \neg A\wedge\neg B}$$

produces the following evaluated tree (we use $t{:}F$ instead $[\![t]\!]F$ to simplify the picture):

$$\dfrac{\dfrac{\dfrac{\dfrac{x{:}A \Rightarrow x{:}A}{x{:}A \Rightarrow u_1{+}u_2{:}(x{:}A\vee y{:}B)} \quad \bot \Rightarrow \bot}{z{:}\neg u_1{+}u_2{:}(x{:}A\vee y{:}B), x{:}A \Rightarrow \bot}}{z{:}\neg u_1{+}u_2{:}(x{:}A\vee y{:}B) \Rightarrow v{:}\neg x{:}A} \qquad \dfrac{\dfrac{\dfrac{y{:}B \Rightarrow y{:}B}{y{:}B \Rightarrow u_1{+}u_2{:}(x{:}A\vee y{:}B)} \quad \bot \Rightarrow \bot}{z{:}\neg u_1{+}u_2{:}(x{:}A\vee y{:}B), y{:}B \Rightarrow \bot}}{z{:}\neg u_1{+}u_2{:}(x{:}A\vee y{:}B) \Rightarrow w{:}\neg y{:}B}}{z{:}\neg u_1{+}u_2{:}(x{:}A\vee y{:}B) \Rightarrow p{:}(v{:}\neg x{:}A\wedge w{:}\neg y{:}B)}$$

Here $u_1, u_2, v, w, p$ are provisional proof variables corresponding to all four essential positive families in the tree. According to the algorithm, all of provisional variables will be evaluated by term appearing from the lifting lemma used at the corresponding nodes.

The variable $u_1$ should be specified at the node labeled by the sequent

$$[\![x]\!]A \Rightarrow [\![u_1{+}u_2]\!]([\![x]\!]A \vee [\![y]\!]B).$$

For that we apply Lifting to

$$[\![x]\!]A \vdash_{\mathcal{LP}} [\![x]\!]A \vee [\![y]\!]B.$$

In this particular case it is easy to write down a term for $u_1$ explicitly. Let $a$ be a proof constant satisfying

$$\vdash_{\mathcal{LP}} [\![a]\!]([\![x]\!]A \rightarrow ([\![x]\!]A \vee [\![y]\!]B)).$$

Since also

$$[\![x]\!]A \vdash_{\mathcal{LP}} [\![!x]\!][\![x]\!]A,$$

we have

$$[\![x]\!]A \Rightarrow [\![a{\cdot}!x]\!]([\![x]\!]A \vee [\![y]\!]B),$$

and $u_1$ should be evaluated by $a{\cdot}!x$. Similarly, $u_2$ should be evaluated by $b{\cdot}!y$, where $b$ is a proof constant specified by the condition

$$\vdash_{\mathcal{LP}} [\![b]\!]([\![y]\!]B \rightarrow ([\![x]\!]A \vee [\![y]\!]B)).$$

To find a term $s(z)$ for $v$ consider a node labeled by

$$[\![z]\!]\neg[\![a!x{+}b!y]\!]([\![x]\!]A \vee [\![y]\!]B) \Rightarrow [\![v]\!]\neg[\![x]\!]A.$$

From its preceeding sequent we have

$$[\![z]\!]\neg[\![a!x{+}b!y]\!]([\![x]\!]A \vee [\![y]\!]B), [\![x]\!]A \vdash_{\mathcal{LP}} \perp,$$

by Deduction, we get

$$[\![z]\!]\neg[\![a!x{+}b!y]\!]([\![x]\!]A \vee [\![y]\!]B) \vdash_{\mathcal{LP}} \neg[\![x]\!]A,$$

and by Lifting, we get $s(z)$ such that

$$[\![z]\!]\neg[\![a!x{+}b!y]\!]([\![x]\!]A \vee [\![y]\!]B) \vdash_{\mathcal{LP}} [\![s(z)]\!]\neg[\![x]\!]A.$$

Similarly, we evaluate $w$ by term $r(z)$ such that

$$[\![z]\!]\neg[\![a!x{+}b!y]\!]([\![x]\!]A \vee [\![y]\!]B) \vdash_{\mathcal{LP}} [\![r(z)]\!]\neg[\![y]\!]B.$$

Finally, the provisional variable $p$ is evaluated by a term $t(z)$ such that

$$[\![z]\!]\neg[\![a!x{+}b!y]\!]([\![x]\!]A \vee [\![y]\!]B) \vdash_{\mathcal{LP}} [\![t(z)]\!]([\![v]\!]\neg[\![x]\!]A \wedge [\![w]\!]\neg[\![y]\!]B).$$

## 9 Logic of Proofs *vrs* Typed Lambda Calculus.

The rule of the $\lambda$ abstraction can be realized as an admissible rule of inference in the Logic of Proofs (lemma 2.5). This shows a way to realize the entire Typed $\lambda$-calculus in $\mathcal{LP}$ by emulating the formation rules for $\lambda$-terms by the corresponding admissible rules in $\mathcal{LP}$. This realization gives a direct arithmetical provability semantics for the Typed $\lambda$-calculus.

## Acknowledgements.

The research was supported by ARO under the MURI program "Integrated Approach to Intelligent Systems", grant number DAAH04-96-1-0341.

## References

[1] S. Artëmov, "Logic of Proofs," *Annals of Pure and Applied Logic*, v. 67 (1994), pp. 29-59.

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

[3] G. Boolos, *Logic of Provability.*, CUP, 1993.

[4] D. van Dalen, *Logic and Structure*, Springer-Verlag, 1994.

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

[6] K. Gödel, "Eine Interpretation des intuitionistischen Aussagenkalkuls", *Ergebnisse Math. Colloq.*, Bd. 4 (1933), S. 39-40.

[7] A. Heyting, "Die intuitionistische Grundlegung der Mathematik", *Erkenntnis*, Bd. 2 (1931), S. 106-115.

[8] A. Kolmogoroff, "Zur Deutung der intuitionistischen Logik," *Math. Ztschr.*, Bd. 35 (1932), S.58-65.

[9] J.C.C. McKinsey and A. Tarski, "Some theorems about the sentential calculi of Lewis and Heyting", Journ.Symb. Logic, v. 13 (1948), pp. 1-15.

[10] V.E. Plisko, "On arithmetic complexity of certain constructive logics", transl from *Mat Zametki*,v. 52, pp. 701-709, 1992.

[11] A.S. Troelstra and D. van Dalen, *Constructivism in Mathematics. An Introduction*, v. 1, Amsterdam; North Holland, 1988.

[12] A.S. Troelstra and H. Schwiftenberg, *Basic Proof Theory*, Cambrudge University Press, 1996.