Международная конференция "Мальцевские чтения"

On Brouwer-Heyting-Kolmogorov provability semantics

Sergei Artemov
City University of New York, Graduate Center

Новосибирск, 13 ноября 2013 года

Constructive Semantics

The intended meaning of intuitionistic logic is given by the informal *Brouwer-Heyting-Kolmogorov (BHK) semantics* of constructive proofs:

1. a proof of $A \rightarrow B$ is a construction which, given a proof of A, returns a proof of B;
2. a proof of $A \wedge B$ consists of a proof of A and a proof of B;
3. a proof of $A \vee B$ is given by presenting either a proof of A or a proof of B;
4. a proof of $\forall x A(x)$ is a function converting any c into a proof of $A(c)$;
5. a proof of $\exists x A(x)$ is a pair (c, d) where d is a proof of $A(c)$.

Revealed the computational content of intuitionistic logic: realizability models, Martin-Lof Type Theory, etc. However, **the original provability BHK turned out to be elusive.**

Negation problem

BHK negation $\neg F$ is $F \rightarrow \bot$, where $\bot$ is a statement that does not have a proof. By BHK clause 1, a proof of $\neg$F is *a construction which brings to the contradiction any proof assertion concerning F.*

Suppose *F* does not have a proof, then ***any p is a `proof' of*** $\neg F$: the assumption *"c is a proof of F"* is provably false, hence yields that $p(c)$ is a proof of the contradiction.

This feature of BHK is rather disturbing:
no relevance between a proof and a sentence it proves:
no constructivity of intuitionistic truths: sentences $\neg F$ as above are de facto assumed to be true without having a meaningful witness.

Universal quantification problem

Consider BHK clause for universal quantifier:

A proof of $\forall u F(u)$ *is a function converting* d *into a proof of* $F(d)$

Here is a simple "constructive proof" S of the Fermat's Last Theorem (FLT): let u range over quadruples of integers (x, y, z, n), and $F(u)$ is the standard Fermat's condition that if $x, y, z > 0$ and $n > 2$, then $x^n + y^n \neq z^n$ which is clearly algorithmically verifiable for each specific u. Algorithm S takes any specific quadruple $u = d$, substitutes it to $F(u)$, and presents a straightforward PA-derivation of $F(d)$. Apparently, S satisfies the aforementioned BHK $\forall$-clause, but could not by any stretch of imagination be called a proof of FTL.

Similar reasoning provides a "constructive proof" for each true Pi-1 sentence $\forall x\, F(x)$: the required algorithm takes n and searches for a proof of $F(n)$.

Kreisel Second Clause correction

In 1962, Kreisel offered corrections to the problematic implication and universal quantifier clauses:

1´ a proof *p* of $A \rightarrow B$ is a construction which given a proof *u* of *A* returns a proof p(u) of *B*, ***plus a verification that p indeed satisfies these conditions.***

4´. a proof *p* of $\forall x A(x)$ is a function converting any *c* into a proof of $A(c)$, ***plus a verification that p indeed satisfies these conditions.***

Kreisel Second Clause correction

We suggest calling BHK augmented by Kresel's second clause

BHK*

and consider BHK* the ultimate (informal) formulation of BHK. This approach has been shared by Myhill, van Dalen, Troelstra, Goodman, Dummett, Feferman, Beeson, Sundholm, and others.

Prior to the Logic of Proofs solution which we present in this talk, despite considerable efforts, there were no formalization of the Brouwer-Heyting-Kolmogorov semantics BHK* given.

To make this point, we will consider four main efforts:

Kreisel-Goodman theory of constructions;

Realizability;

Curry-Howard isomorphism;

Intuitionistic Type Theory.

Kreisel's Theory of Constructions *C*

There were many features introduced there that were eventually used in the Logic of Proofs solution. However, the original variant of ***C*** was inconsistent. Goodman in 1970 fixed that gap but his solution involved a stratification of constructions into levels which ruined the *BHK* character of this semantics.

A comprehensive account of the Kreisel-Goodman theory could be found in S. Weinstein's paper of 1983, which concludes that "*The interpretation of intuitionistic theories in terms of the notions of proof and construction . . . has yet, however, failed to receive a definitive formulation.*"

Walter Dean and Hidenori Kurokawa are now making a serious effort to investigate how close ***C*** was to a BHK* formalization.

Realizability

Kleene realizability disclosed a *computational* content of formal intuitionistic derivations which is however quite different from the provability semantics. Kleene realizers are not proofs in a formal theory, the predicate "*r* realizes *F*" is not decidable. Realizability models not the original BHK, but BHK adjusted by adding a selector to the disjunction clause (BHKs). Finally, Realizability does not satisfy BHK* (Kreisel Second Clause):

- *a realizer for an implication is a computable function that takes a realizer for the hypothesis and produces a realizer for the conclusion*.
- *a realizer for a universal statement is a computable function that produces, for each m, a witness for the formula instantiated with m*.

In particular, **n** *realizes* $\neg F$ iff for no **m**, **m** *realizes F.*
No relevance between a witness **n** and the sentence $\neg F$.

Curry-Howard Isomorphism

Curry-Howard isomorphism transliterates natural derivations in intuitionistic logic to typed λ-terms hence providing a typed functional reading of logical derivations.

As proof objects Curry-Howard λ-terms do not have a semantic value since they denote the very intuitionistic derivations which semantics is supposed to justify and thus yield an immediately circular provability semantic. Above all, CH does not have a verification mechanism and hence does not support BHK*.

J.-Y. Girard: "*In both of these cases* [*CH and Realizability*], *the foundational pretensions have been removed. This allows us to make a good use of an idea which may have spectacular applications in the future.*" (Proofs and Types, 1989).

Intuitionistic Type Theory

Martin-Löf's ITT is a rich calculus of types and computational terms: it uses the proof/witness rhetoric to illustrate the rules, but is ultimate formal semantics of ITT is computational: *typed computational programs*.

ITT does not seem to support BHK*, but can serve as a *semi-formal foundation of intuitionistic reasoning*, a kind of refined BHK, with some formal system on the background, which, however, is supposed to be read with some informal "intuitionistic" semantics in mind to properly fill in the gaps in the formal exposition. This can be sufficient for a devoted intuitionist who does not seek a compete formalization of foundations, but the for the purposed of BHK formalization,

ITT does not formalize BHK*

Computational BHK is not BHK*

Computational BHK (Realizability, CH, ITT) does not address the negation problem: in realizability,

n *realizes* $\neg F$ iff for no **m**, **m** *realizes F.*

The predicate "for no **m**, **m** *realizes F*" is not constructive, its realizer **n** does not appear to qualify as its constructive 'witness' since it does not carry any information about the validity of "for no **m**, **m** *realizes F.*" All independent formulas are constructively false since their negations are 'baptized' as realizable by any witness **n**.

The principal conceptual failure of the computational BHK is its inability to resolve the Schwichtenberg paradox: in the computational model, the algorithm from the paradox description is a legitimate "proof."

BHKs = BHK + selector

In computational semantics there is a requirement of a bit selector that points at the proper disjunct:

> 3′. p proves $A \vee B$ iff $p = (p_0, p_1)$ with $p_0 \in \{0, 1\}$, and p_1 proves A if $p_0 = 0$ and p_1 proves B if $p_0 = 1$.

This adjustment illustrates a difference between proofs and computational programs in the BHK setting: the proof predicate

$$p \text{ } \textit{is a proof of } F \tag{1}$$

is **decidable**, whereas the realizability assertion

$$p \text{ } \textit{realizes } F \tag{2}$$

is **not decidable**. The ‘selector’ is needed for (2) but is redundant for (1) since given p, one can compute the right disjunct.

What does computational BHK formalize?

Computational BHK (Realizability, CH, ITT) does not actually formalize the original BHK, it rather formalizes BHKs which is a version of BHK in which the disjunction clause 3 is replaced by a stronger requirement 3´ with selector. If we compare the requirements to witnesses, then

$$\text{BHK} \supset \text{BHKs} \supset \text{BHK*}.$$

Inclusion BHKs ⊃ BHK* holds assuming another Kreisel’s suggestion to keep the relation witness/formula decidable.

BHK underspecifies constructive provability semantics;
BHKs - computability semantics: Realizability, CH, ITT;
BHK* - the intended semantics of constructive proofs, **is not** formalized by computability-based systems such as Realizability, CH, ITT.

Intuitionistic vs. Classical Perspective

Intuitionists normally base their formal systems on intuition of constructive, e.g., ***BHK-style informal semantics***, rather then on classical foundations. This approach will not be discussed here.

Classical mathematicians (such as Gödel, Kolmogorov, Kleene, Novikov, and others) seek a rigorous

classical definition of the constructive semantics.

In particular, BHK "proofs" and "constructions" should be defined using mathematical objects, e.g., classical proofs and programs, so that intuitionistic postulates become theorems of classical mathematics. This approach will be the object of our study.

Provability calculus, 1933

Kolmogorov and Gödel viewed BHK-proofs classically. Gödel endorsed classical modal logic S4 as the calculus of provability:

Axioms and rules of classical propositional logic,

$\Box(F \to G) \to (\Box F \to \Box G)$,

$\Box F \to F$ *reflexivity,*

$\Box F \to \Box\Box F$ *introspection,*

Rule of Necessitation: $\dfrac{\vdash F}{\vdash \Box F}$.

Gödel offered connecting classical provability with intuitionistic logic in a way that respects the provability reading of IPC:

$$\mathsf{IPC} \vdash F \quad \textit{iff} \quad \mathsf{S4} \vdash tr(F),$$

where $tr(F)$ is obtained by 'boxing' each subformula of F.

Provability embedding

When parsing Gödel's translation $tr(F)$ of some formula F, we encounter a provability modality before each subformula, which forces us to read said subformula as provable rather than true. Therefore, Gödel's translation reflects the fundamental intuitionistic paradigm that intuitionistic truth is provability. Moreover, the classical version of BHK is that which provides a non-circular semantics for intuitionistic logic.

At that stage, the problem of finding a provability semantics for IPC seemed to reduce to developing such a semantics for S4. There was an immediate problem and a subtle problem along this pass.

Provable reflexivity issue

The immediate problem, noticed by Gödel, was that reflexivity is not compatible with a suggestive reading of $\Box$ as a formal provability in PA. Indeed, from $\Box\bot \to \bot$ one can derive

$$\Box(\neg\Box\bot)$$

in S4, which states the provability of consistency in PA. Later in 1938, Gödel suggested an elegant way around this problem, which was to return to the original BHK language of explicit proofs: if $p\!:\!F$ stands for *p is a proof of F*, then the explicit reflection

$$p{:}F \to F$$

is internally provable since $p{:}F$ is decidable.

Impredicativity issue

The second, more subtle problem was impredicativity of BHK in its implication clause. In realizability, it manifested itself in undecidability of realizers. We will soon see how this problem played out in provability BHK.

Perhaps, due to these difficulties, provability BHK proved to be quite elusive. In this context, Gödel in one of his lectures in 1938 discussed the possibility of a classical logic of proofs which could provide a provability semantics of S4, hence for IPC. However, this Gödel's lecture remained unpublished till the third volume of Kurt Gödel Collected work appeared in 1995, when the Logic of Proofs LP has already been independently developed.

First steps

The idea of the logic of proofs LP was to make provability operators in S4 explicit by using the proof assertions $t{:}F$. The first steps were straightforward, almost trivial: a direct inspection of the Hilbert-Bernays derivability conditions yielded two computable operations on proofs: *application*

$$t{:}(A \rightarrow B) \rightarrow (s{:}A \rightarrow [t \cdot s]{:}B)$$

and *proof checker*:

$$t{:}A \rightarrow\ !t{:}t{:}A.$$

whose forgetful projections correspond to basic S4 axioms

$$\Box(F \rightarrow G) \rightarrow (\Box F \rightarrow \Box G) \text{ and } \Box F \rightarrow \Box\Box F.$$

New operation +

Further analysis showed that one more operation on proofs is needed to capture the S4 reasoning. Provability operators do not distinguish between different proofs of the same fact; in the explicit setting, we need an operation that reconciles such proofs. We call such operation '+' and assume the identities

$$t{:}A \rightarrow [t+s]{:}A \qquad \text{and} \qquad s{:}A \rightarrow [t+s]{:}A.$$

As motivation, one might think of s and t as two volumes of an encyclopedia, and $s+t$ as the set of those two volumes. Imagine that one of the volumes, say s, contains a sufficient justification for a proposition F, i.e., $s{:}F$ is the case. Then the larger set $s+t$ also contains a sufficient justification for F, $[s+t]{:}F$. In the context of Hilbert-style derivations, '$s+t$' can be interpreted as a concatenation of proofs s and t.

The basic Logic of Proofs

In the language of LP, proofs are represented by *proof terms* constructed from *proof variables* and *proof constants* by means of functional symbols for elementary computable operations on proofs, binary $\cdot$, $+$, and unary $!$. The formulas of LP are built by Boolean connectives from propositional atoms and those of the form $t{:}F$ where t is a *proof term* and F is a formula.

The basic system of the Logic of Proofs LP_0 has the axioms and rules of classical logic along with the schemas:

$t{:}A \rightarrow A$	*reflexivity.*
$t{:}(A \rightarrow B) \rightarrow (s{:}A \rightarrow (t \cdot s){:}B)$	*application*
$t{:}A \rightarrow (t+s){:}A, \quad s{:}A \rightarrow (t+s){:}A$	*sum*
$t{:}A \rightarrow !t{:}t{:}A$	*proof checker.*

Arithmetical semantics

$t{:}A \to A$ *reflexivity.*

$t{:}(A \to B) \to (s{:}A \to (t \cdot s){:}B)$ *application*

$t{:}A \to (t+s){:}A, \quad s{:}A \to (t+s){:}A$ *sum*

$t{:}A \to\, !t{:}t{:}A$ *proof checker.*

The intended semantics for LP_0 is provided by proof predicates in Peano Arithmetic PA. The proof terms are interpreted by codes of arithmetical derivations. Operations $\cdot$, $+$, and unary ! become total recursive functions on such codes. Formulas of LP are interpreted by closed arithmetical formulas; interpretations commute with Boolean connectives and $t{:}F$ is interpreted by an arithmetical proof predicate that numerates theorems of PA.

Invariant principles

It is not sufficient to consider only the standard proof predicate $\mathit{Proof}(x, y)$

$$\text{`}x \textit{ is a proof of } y.\text{'}$$

A fixed proof predicate could support some sporadic identities. For example, consider the question of whether

$$x{:}(\top \wedge \top) \rightarrow x{:}\top$$

is a sound principle? We don't know the answer for $\mathit{Proof}(x, y)$ and **we don't want to know this answer**. By re-arranging the set of proofs insignificantly, one can falsify this 'law' without compromising the ability of the proof system to effectively numerate the same set of theorems. So, we have to consider only *principles that hold for all proof systems*. This sometimes is called super-evaluation.

Soundness and completeness in PA

A *proof system* is a provably decidable predicate $\mathit{Proof}(x, y)$ that enumerates all theorems of PA, i.e.,

$$\mathsf{PA} \vdash \varphi \quad \text{iff} \quad \mathit{Proof}(n, \varphi) \ \text{holds for some } n\ ,$$

together with computable functions which satisfy identities for '$\cdot$,' '$+$,' and '!' respectively.

An *arithmetical interpretation* $*$ consists of a proof system, interpretation of proof variables and constants by codes of proofs, and propositional variables by arithmetical sentences. Boolean connectives do not change and

$$(p{:}F)^* = \mathit{Proof}(p, F).$$

Theorem: LP_0 *is sound and complete.*

Kripke-style semantics

Kripke-style models for LP_0 are built from the usual $\mathsf{S4}$-models $(W, R, \models)$. We retain a classical interpretation $*$ formulas Fm as propositions, i.e., as *subsets of the set* W *of possible worlds*,

$$*: Fm \mapsto 2^W$$

and $u \models F$ means $u \in F^*$.

We interpret proof terms Tm at each world as *sets of formulas*,

$$*: W \times Tm \mapsto 2^{Fm}$$

and $u \models t{:}F$ means that $F \in *(u,t)$. Some natural closure conditions are assumed.

Interpreting proofs syntactically, as sets of formulas rather than propositions is crucial: theory of justification cannot be built on the traditional 'propositions' paradigm.

Constant specifications

To capture S4, we have to explicitly represent the internalization property:

$$\vdash F \quad \Rightarrow \quad \vdash \Box F.$$

Constant Specification CS is a set of formulas of the form $c{:}A$ where c is a proof constant and A is an axiom of LP_0.

Some special CS's:

Empty: $CS = \emptyset$.

Finite: CS is a finite set of formulas.

Total: for each axiom A and any constants c, $c{:}A$ is in CS.

Logic of Proofs with given Constant Specification CS:

$$\mathsf{LP}_{CS} = \mathsf{LP}_0 + CS$$

Logic of Proofs LP is LP_{CS} for the total CS.

Internalization

One of the basic properties of LP is its capability of internalizing its own derivations. The weak form:

$$\textit{if} \vdash F,\ \textit{then} \vdash p{:}F\ \textit{for some proof term}\ p\ .$$

The following more general *internalization rule* holds for LP: *if*

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

then there is a proof term $t(x_1, \ldots, x_n)$ *such that*

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

The Curry-Howard isomorphism covers only a simple instance of the proof internalization property where all of $A_1, \ldots, A_n, B$ are purely propositional formulas containing no proof assertions.

Realization

The principal feature of LP is its ability to realize all S4 theorems by restoring corresponding proof terms inside occurrences of modality.

A *forgetful projection* of an LP-formula F is a modal formula obtained by replacing all assertions $t{:}(\cdot)$ in F by $\Box(\cdot)$.

Realization Theorem: S4 *is the forgetful projection of* LP.

That the forgetful projection of LP is S4-compliant is a straightforward observation. The converse has been established by presenting an algorithm which substitutes proof terms for all occurrences of modalities in a cut-free Gentzen-style S4-derivation of a formula F, thereby producing a formula F^r derivable in LP.

Realization: example

	Derivation in S4	Derivation in LP
1.	$\Box A \to \Box A \vee B$	$x{:}A \to x{:}A \vee B$
2.	$\Box(\Box A \to \Box A \vee B)$	$a{:}(x{:}A \to x{:}A \vee B)$
3.	$\Box\Box A \to \Box(\Box A \vee B)$	$!x{:}x{:}A \to (a\cdot !x){:}(x{:}A \vee B)$
4.	$\Box A \to \Box\Box A$	$x{:}A \to !x{:}x{:}A$
5.	$\Box A \to \Box(\Box A \vee B)$	$x{:}A \to (a\cdot !x){:}(x{:}A \vee B)$
5′.		$(a\cdot !x){:}(x{:}A \vee B) \to (a\cdot !x + b\cdot y){:}(x{:}A \vee B)$
5″.		$x{:}A \to (a\cdot !x + b\cdot y){:}(x{:}A \vee B)$
6.	$B \to \Box A \vee B$	$B \to x{:}A \vee B$
7.	$\Box(B \to \Box A \vee B)$	$b{:}(B \to x{:}A \vee B)$
8.	$\Box B \to \Box(\Box A \vee B)$	$y{:}B \to (b\cdot y){:}(x{:}A \vee B)$
8′.		$(b\cdot y){:}(x{:}A \vee B) \to (a\cdot !x + b\cdot y){:}(x{:}A \vee B)$
8″.		$y{:}B \to (a\cdot !x + b\cdot y){:}(x{:}A \vee B)$
9.	$\Box A \vee \Box B \to \Box(\Box A \vee B)$	$x{:}A \vee y{:}B \to (a\cdot !x + b\cdot y){:}(x{:}A \vee B)$

Extra steps 5′, 5″, 8′, and 8″ are needed to reconcile different internalized proofs of the same formula.

Realization via cut-elimination

There are several proofs of the realization theorem already known (S.A., Fitting, Wang). The first proof was constructive and uses cut-free proofs in S4. The resulting realization respects Skolem's idea that negative occurrences of existential quantifiers over proofs (hidden in the modality of provability) are realized by free proof variables whereas positive occurrences are realized by functions of those variables.

The Realization Theorem provides S4, and therefore intuitionistic logic IPC, with an exact semantics via LP proof terms. To complete building a provability BHK semantics for IPC it is now sufficient to note that LP has a natural interpretation in a system of formal proofs in Peano arithmetic PA or a similar system capable of encoding its own proofs.

Self-referentiality of proofs is needed.

LP admits *self-referential types* of the sort $t{:}F(t)$ stating that t is a proof of a sentence F which explicitly contains t. This self-referentiality is supported by the provability semantics that includes an arithmetical fixed-point argument. But is self-referentiality actually needed for the provability BHK semantics?

Consider so-called Moore sentence: *It rains but I don't know it.* If p stands for *it rains* and $\Box$ denotes 'knowledge' then a modal formalization of Moore sentence is

$$M = p \wedge \neg\Box p.$$

M is easily satisfiable, hence consistent, e.g., when p is true but not known. However, it is impossible to know Moore's sentence. Indeed, S4 proves $\neg\Box M = \neg\Box(p \wedge \neg\Box p)$.

Kuznets' Theorem

Here is a derivation of $\neg\Box M$ is S4:

1. $(p \wedge \neg\Box p) \to p$, *logical axiom*
2. $\Box((p \wedge \neg\Box p) \to p)$, *Necessitation*
3. $\Box(p \wedge \neg\Box p) \to \Box p$, *from 2, by Distribution*
4. $\Box(p \wedge \neg\Box p) \to (p \wedge \neg\Box p)$, *Reflexivity*
5. $\neg\Box(p \wedge \neg\Box p)$, *from 3 and 4, in Boolean logic*

Its natural realization in LP is self-referential:

1. $(p \wedge \neg[c\cdot x]{:}p) \to p$, *logical axiom*
2. $c:((p \wedge \neg[c\cdot x]{:}p) \to p)$, *self-referential CS*
3. $x{:}(p \wedge \neg[c\cdot x]{:}p) \to [c\cdot x]{:}p$, *from 2, by Application*
4. $x{:}(p \wedge \neg[c\cdot x]{:}p) \to (p \wedge \neg[c\cdot x]{:}p)$, *Reflexivity*
5. $\neg x{:}(p \wedge \neg[c\cdot x]{:}p)$, *from 3 and 4, in Boolean logic*

Kuznets's Theorem: *Any realization of* $\neg\Box M$ *in* LP *requires self-referential constant specifications.*

The impredicativity of BHK manifests itself!

Yu's Theorem

The question of self-referentiality of BHK-semantics for IPC has been answered by Yu (my Ph.D. student from New York). Extending Kuznets' method, he established

Yu's Theorem: *Each* LP *realization of the intuitionistic law of double negation* $\neg\neg(\neg\neg p \rightarrow p)$ *requires self-referential constant specifications.*

These results indicate that provability BHK semantics for S4 and IPC is intrinsically self-referential and needs a fixed-point construction for connect it to formal proofs in PA or similar systems. This might explain, in part, why any attempt to build provability BHK semantics in a direct inductive manner without self-referentiality was doomed to fail.

First order LP

FOLP was developed in 2011 in a joint work with Yavorsyaya.

Let $A(x)$ be $x = 0$. Then formula $\Box A(x)$ stating, in the provability setting, that $x = 0$ *is provable*, has x free and holds when x is instanced by 0. In such situations, i.e., when x is available for substitutions in $\Box A(x)$, we call such x a *global variable*. This is the modal logic reading.

However, there is another natural meaning of $\Box A(x)$, namely, that '$x = 0$' is provable as a syntactical object. Under this meaning, $\Box A(x)$ does not depend on a specific value of x, and is just false as a statement about provability in PA, since '$x = 0$' is not provable. In such situations, we call a variable x *local* since its scope does not extend beyond the provability operator. This 'local' reading of variables is not allowed in the traditional modal languages, but is principally needed for the first-order logic of proofs.

Proof assertions in FOLP

In the language of FOLP, the proof predicate is represented by formulas of the form

$$t{:}_X A$$

where X is a finite set of individual variables that are considered global parameters and free variables of this formula. All occurrences of variables from X that are free in A are also free in $t{:}_X A$. All other free variables of A are considered local and hence bound in $t{:}_X A$. For example, if $A(x, y)$ is an atomic formula, then in $p{:}_{\{x\}} A(x, y)$, variable x is free and variable y is bound. Likewise, in $p{:}_{\{x,y\}} A(x, y)$ both variables are free and in $p{:}_{\emptyset} A(x, y)$, neither x nor y is free.

Proofs are represented by proof terms which do not contain individual variables.

Language of FOLP - proof terms

FOLP is the extension of the first-order logic by means to represent proofs and proof assertions:

- proof variables $p_0, p_1, p_2, \ldots$ and constants $c_0, c_1, c_2, \ldots$;
- functional symbols for operations on proofs:
  - those of LP: binary $+$, $\cdot$, and unary $!$,
  - unary gen_x for each individual variable x;
- an operational symbol $(\cdot){:}_X(\cdot)$ for each finite set X of individual variables.

Proof terms are constructed from proof variables and constants by operations $+$, $\cdot$, $!$, and gen_x. Proof terms do not contain individual variables. Note also that in gen_x, variable x is merely a syntactic label of this operation and is not considered an occurrence of a variable.

Language of FOLP - formulas

Formulas are defined in the standard way with an additional clause for the proof operator. Namely,

- If t is a proof term, X a finite set of individual variables, and A is a formula, then

$$t{:}_X A$$

  is a formula. In this formula, all variables from X, and only from X, are free. All free occurrences of variables from X in A are also free.

The set of free variables of a formula A is denoted by $FVar(A)$. We use the abbreviation $t{:}A$ for $t{:}_\emptyset A$.

FOLP - axioms and rules

The basic first-order logic of proofs FOLP_0 has axioms and rules:

A1 *classical axioms of first-order logic*

A2 $t{:}_{Xy}A \rightarrow t{:}_X A, \quad y \notin FVar(A)$

A3 $t{:}_X A \rightarrow t{:}_{Xy}A$

B1 $t{:}_X A \rightarrow A$

B2 $s{:}_X(A \rightarrow B) \wedge t{:}_X A \rightarrow (s \cdot t){:}_X B$

B3 $t{:}_X A \rightarrow (t+s){:}_X A, \quad s{:}_X A \rightarrow (t+s){:}_X A$

B4 $t{:}_X A \rightarrow\, !t{:}_X t{:}_X A$

B5 $t{:}_X A \rightarrow \mathsf{gen}_x(t){:}_X \forall x A,\ x \notin X$

R1 $\vdash A, A \rightarrow B \quad \Rightarrow \quad \vdash B$ *modus ponens*

R2 $\vdash A \quad \Rightarrow \quad \vdash \forall x A$ *generalization*

As before, FOLP_{CS} denotes FOLP with a constant specification CS, FOLP corresponds to the total constant specification.

FOLP - axioms and rules

Here is the same set of postulates for FOLP_0 with subscripts X suppressed for better readability:

A1 *classical axioms of first-order logic*

A2 $t{:}_y A \rightarrow t{:}A, \quad y \notin FVar(A)$

A3 $t{:}A \rightarrow t{:}_y A$

B1 $t{:}A \rightarrow A$

B2 $s{:}(A \rightarrow B) \wedge t{:}A \rightarrow (s \cdot t){:}B$

B3 $t{:}A \rightarrow (t+s){:}A, \quad s{:}A \rightarrow (t+s){:}A$

B4 $t{:}A \rightarrow\, !t{:}t{:}A$

B5 $t{:}A \rightarrow \mathsf{gen}_x(t){:}\forall x A,\ x \notin X$

R1 $\vdash A, A \rightarrow B \;\Rightarrow\; \vdash B$ *modus ponens*

R2 $\vdash A \;\Rightarrow\; \vdash \forall x A$ *generalization*

Derivation example

Let us derive in FOLP an explicit counterpart of the converse Barcan Formula $\Box\forall xA \rightarrow \forall x\Box A$.

1. $\forall xA \rightarrow A$ - logical axiom;
2. $c{:}(\forall xA \rightarrow A)$ - axiom necessitation;
3. $c{:}_{\{x\}}(\forall xA \rightarrow A)$ - from 2, by axiom A3;
4. $c{:}_{\{x\}}(\forall xA \rightarrow A) \rightarrow (u{:}_{\{x\}}\forall xA \rightarrow (c\cdot u){:}_{\{x\}}A)$ - axiom B2;
5. $u{:}_{\{x\}}\forall xA \rightarrow (c\cdot u){:}_{\{x\}}A$ - from 3, 4, by Modus Ponens;
6. $u{:}\forall xA \rightarrow u{:}_{\{x\}}\forall xA$ - by axiom A3;
7. $u{:}\forall xA \rightarrow (c\cdot u){:}_{\{x\}}A$ - from 5, 6;
8. $\forall x[u{:}\forall xA \rightarrow (c\cdot u){:}_{\{x\}}A]$ - from 7, by generalization;
9. $u{:}\forall xA \rightarrow \forall x(c\cdot u){:}_{\{x\}}A$ - from 8, since the antecedent of 8 does not contain x free.

Internalization

Internalization: *Let* $p_0, \ldots, p_k$ *be proof variables,* $X_0, \ldots, X_k$ *be sets of individual variables, and* $X = X_0 \cup X_1 \cup \ldots \cup X_k$. *Suppose that in* FOLP

$$p_0{:}_{X_0} A_0, \ldots, p_k{:}_{X_k} A_k \vdash F.$$

Then there exists a proof term $t(p_0, p_1, \ldots, p_k)$ *such that*

$$p_0{:}_{X_0} A_0, \ldots, p_k{:}_{X_k} A_k \vdash t{:}_X F.$$

Realization of FOS4 and HPC

Let A be a first-order modal formula. By *realization* of a formula A we mean a formula A^r of the language of FOLP that is obtained from A by replacing all occurrences of subformulas of A of the form $\Box B$ by $t{:}_X B$ for some proof terms t and such that $X = FVar(B)$. A realization is *normal* if all negative occurrences of $\Box$ are assigned proof variables.

Realization Theorem *If* FOS4 $\vdash A$*, then there is a normal realization* A^r *such that* FOLP $\vdash A^r$.

Corollary F *is derivable in* HPC *if and only if its Gödel translation is realizable in* FOLP.

Generic provability semantics

A *generic proof predicate* is a provably Δ_1-formula $Prf(x,y)$ such that for every arithmetical formula φ,

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

along with some general effectiveness conditions, e.g., open variables in derivations are emulated by appropriate provably recursive functions.

This is the most general and abstract provability semantics of the three and the closest to the informal understanding of first-order provability logic.

Soundness Theorem:

If $\mathsf{FOLP} \vdash A$ *with a constant specification CS, then for every generic interpretation* $*$ *respecting CS*, $\mathsf{PA} \vdash A^*$.

To what extent FOLP is BHK*?

We show that the Gödel embedding + Logic of Proofs realization provides an exact semantics of first-order logic language which satisfies the BHK* requirements to constructive proofs.

It is very instructive to watch how exactly Gödel/LP formalization straightens known omissions of the original BHK and delivers a provability sound realization of BHK* including all Kreisel's additions.

Conjunction

A proof of $A \wedge B$ consists of a proof of A and a proof of B.

An intuitionistic conjunction $A \wedge B$ is realized in FOLP as

$$t{:}(\widetilde{A} \wedge \widetilde{B})$$

where $\widetilde{A}$ and $\widetilde{B}$ are, as before, FOLP-versions of A and B. This t contains sufficient information to recover both a proof of $\widetilde{A}$ and a proof of $\widetilde{B}$. Indeed, given such t and commonly known proofs a and b such that

$$a{:}((\widetilde{A} \wedge \widetilde{B}) \rightarrow \widetilde{A}) \quad \text{and} \quad b{:}((\widetilde{A} \wedge \widetilde{B}) \rightarrow \widetilde{B}),$$

one can find a proof of $\widetilde{A}$, $a \cdot t$, and a proof of $\widetilde{B}$, $b \cdot t$. Likewise, having a proof of $\widetilde{A}$ and a proof of $\widetilde{B}$, one can construct a proof of $\widetilde{A} \wedge \widetilde{B}$ within FOLP.

Disjunction

A proof of $A \vee B$ is given by presenting either a proof of A or a proof of B.

Argue in FOLP. Suppose $u{:}A$ or $u{:}B$. We have to construct a proof term $t(u)$ such that $t(u){:}(A \vee B)$. Consider the internalized disjunction principles

$$a{:}(A \rightarrow (A \vee B)) \text{ and } b{:}(B \rightarrow (A \vee B)),$$

both obviously provable in FOLP. Using application axiom B2, we conclude that either $[a \cdot u]{:}(A \vee B)$ or $[b \cdot u]{:}(A \vee B)$. In either case,

$$[a \cdot u + b \cdot u]{:}(A \vee B),$$

and we can set $t(u)$ to $[a \cdot u + b \cdot u]$.

Universal quantifier

A proof of $\forall x A(x)$ *is a function converting* c *into a proof of* $A(c)$

We known that it is not sufficiently precise and admits unacceptable 'constructive proofs.' The provability BHK offers a natural fix. To simplify the notations, assume that $A(x)$ is atomic. An intuitionistic statement $\forall x A(x)$ is represented in FOS4 by

$$\Box \forall x \Box A(x).$$

Its realization in FOLP is

$$u{:}\forall x[v{:}_{\{x\}}A(x)].$$

Its arithmetical interpretation states that there is a uniform proof u that for each c, the substitution of c for x produces a proof $v(c)$ of $A(c)$.

Corrected formulation

Here is the corrected reading of the BHK clause for $\forall$ suggested by Gödel's embedding and its subsequent realization in FOLP:

'*a constructive proof of* $\forall x A(x)$ *is a pair* (f, d) *where* f *is a function and* d *is a classical proof that for each* x, $f(x)$ *is a classical proof of* $A(x)$.'

This new version is non-circular, and is the one which naturally comes to mind after inspecting the Schwichtenberg Paradox: algorithm S fails this new BHK test since is does not provide a required proof that for all u, $S(u)$ is indeed a proof of $F(u)$.

Existential quantifier

A proof of $\exists x A(x)$ is a pair (c, d) where d is a proof of $A(c)$.

Consider an intuitionistic statement $\exists x A(x)$ for an atomic $A(x)$. It is represented in FOS4 as

$$\Box \exists x \Box A(x)$$

Its realization in FOLP is

$$u{:}\exists x[v{:}_{\{x\}}A(x)].$$

Note that under any arithmetical interpretation, $v:_{\{x\}}A(x)$ is provably decidable, hence $\exists x[v{:}_{\{x\}}A(x)]$ is a provably Σ-formula. From a proof u in PA of a Σ-formula $\exists x F(x)$ we can effectively find c and a PA proof d of $F(c)$. Applying this method here, we can obtain c and d such that d is a (classical) proof of $A(c)$.

Negation

The original BHK admitted anything as a 'constructive proof' of $\neg A$ for an unprovable statement A. This is not right since anything passes as a constructive proof of, say, $\neg \mathit{Consis}\ PA$.

Gödel's transtation of $\neg A$ (assume that A is atomic, to keep notations simpler) is $\Box(\neg\Box A)$. Its realization in the Logic of Proofs is

$$t(x){:}[\neg x{:}A]$$

where $t(x)$ is a proof term depending on x. In particular, a 'constructive proof' of $\neg \mathit{Consis}\ PA$ provides a proof term t such that for each x, $t(x)$ is a PA-proof of $\neg x{:}\mathit{Consis}\ PA$. Using joint logics of proofs and provability developed by T. Yavorskaya and E. Nogina, one can show that in the Logic of Proofs there is no such term t. Hence, in provability BHK, not every independent sentence is automatically false and negation is not trivial.

Implication, revisited

Again, consider atomic A and B. The original BHK required a constructive proof of $A \rightarrow B$ to be a construction (computable function) $f(x)$ such that for any proof x of A, $f(x)$ is a proof of B. Symbolically, $x{:}A \rightarrow f(x){:}B$.

As we have seen, this led to problems with constructive semantics of negation and needed a refinement. Provability BHK offers a natural fix (we preserve the language of the original):

a constructive proof of $A \rightarrow B$ *is a pair of constructions* (f, g) *such that for each* x*,* $g(x)$ *is a classical proof that* $x{:}A$ *implies* $f(x){:}B$.

Symbolically this can be written as

$$g(x){:}[x{:}A \rightarrow f(x){:}B].$$

Three facets of BHK semantics

Computational BHK (Realizability, CH, ITT) does not actually formalize the original BHK, it rather formalizes BHKs which is a version of BHK in which the disjunction clause 3 is replaced by a stronger requirement 3´ with selector. If we compare the requirements to witnesses, then

$$\text{BHK} \supset \text{BHKs} \supset \text{BHK*}.$$

Inclusion BHKs ⊃ BHK* holds assuming another Kreisel's suggestion to keep the relation witness/formula decidable.

BHK underspecifies constructive provability semantics;

BHKs - computability semantics: Realizability, CH, ITT;

BHK* - the intended semantics of constructive proofs, is not formalized by Realizability, CH, ITT.

Formalizing 'constructive proofs' requires both programs and verifications, programs only are not enough!