数理解析研究所講究録
1021 巻 1997 年 28-47

Classical Brouwer-Heyting-Kolmogorov interpretation

Masahiko Sato (佐藤 雅彦)

Department of Information Science
Kyoto University
masahiko@kuis.kyoto-u.ac.jp

Abstract. The Brouwer-Heyting-Kolmogorov interpretation explains the meaning of logical operations as operators that construct proofs from proofs of the operands. The BHK interpretation is usually understood as giving intuitionistic interpretation for the logical operators, but, as pointed out by Troelstra and van Dalen [12], it is possible to understand the BHK interpretation classically. We elaborate this idea and develop a classical theory of proofs as abstract mathematical entities where the truth of a proposition becomes equivalent to the existence of proofs of the proposition.

We develop a first order theory of arithmetic, equivalent to **PA**, and give a classical BHK interpretation for the theory. We show the soundness of the interpretation by showing that if a derivation P of a formula A is given, then the interpretation of P is a proof of the interpretation of A. We also show that the interpreted value of derivations is preserved under reductions of derivations.

We also present a system of catch/throw calculus and develop a classical BHK interpretation for it. Since the calculus in non-deterministic, we interpret a derivation by a set of proofs. We show the soundness of the interpretation, and show that if a derivation reduces to another derivation, then the associated set of proofs for the latter derivation is smaller than that for the former derivation.

1 Introduction

The Tarskian interpretation of formulas interprets a mathematical formula either as *true* or *false*. The Brouwer-Heyting-Kolmogorov (BHK) interpretation, on the other hand, interprets a mathematical formula by assigning the set of *proofs* of the given formula. In the BHK interpretation, a formula is true if and only if it has a proof. Here, the notion of a proof is not understood as a formal derivation but as an informal mathematical object just like a natural number or a real number.

In the Tarskian interpretation, for example, $A \supset B$ is true if and only if either A is false or B is true. In the BHK interpretation, $A \supset B$ is true if and only if it has a proof f, i.e., if there is a function f such that for any proof p of A, $f(p)$ is a proof of B. A disjunction $A \vee B$ is true in the Tarskian interpreration if either

A or B is true, and it is true in the BHK interpretation if A or B has a proof p, and in the first case $(0, p)$ is a proof of $A \vee B$ and in the second case $(1, p)$ is a proof of $A \vee B$. So the BHK interpretation gives finer interpretation than the Tarskian interpretation. In addition to this, these two interpretations have the following essential differences.

A proposition is *classically* true if it is true under the Tarskian interpretation, and it is *intuitionistically* true if it is true under the BHK interpretation. Since the law of the excluded middle is always true classically but not always true intuitionistically, we know that these two interpretations are not equivalent.

However, as pointed out in Troelstra and van Dalen [12], by modifying the BHK interpretation appropriately it is possible to interpret propositions in terms of proofs in such a way that any proposition is classically true if and only if it has a proof. In this paper, we elaborate this idea, and show that this *classical* BHK interpretation is consistent with the intuitionistic BHK interpretation in the sense that any proof of a proposition under the intuitionistic BHK interpretation is also a proof of the proposition in the classical BHK interpreration.

We also present a formal system of arithmetic in a modified natural deduction style and show that it is possible to give a sound interpretation of the system in terms of the classical BHK interpretation.

We then consider an extension of the system by the catch/throw inference rules. The resulting system is inherently non-deterministic, and we will show that it is possible to extend the BHK interpretation and give a sound interpretation for this system.

By the well-known Curry-Howard isomorphism, it is possible to regard formal derivations in the intuitionistic fragment of our formal system as programs in a typed language. Then our BHK interpretation gives a natural denotational semantics to this programming language. It is therefore possible to use this framework as a basis for constructive programming where programming is replaced by proving.

2 The classical interpretation

To make our argument concrete, we will work on a first order language whose intended domain of interpretation is the set of natural numbers.

So our language contains the constant 0, a unary function symbol succ (for successor), binary function symbols plus and times. (We also have symbols for all the primitive recursive functions and associated defining axioms, however, we will not mention them explicitly for the sake of simplicity.) We use x, y, z etc. as meta variables for individual variables. We define terms as usual using these symbols. We use a, b, c etc. as meta variables for terms. We will write 0, 1, 2, ... for $0, \mathsf{succ}(0), \mathsf{succ}(\mathsf{succ}(0)), \ldots$ and call these terms *numerals*. We will identify each numeral with the natural number which corresponds to the numeral in an obvious way. We use k, l, m, n etc. as meta variables for numerals (and natural numbers). We will write N for the set of natural numbers.

The only primitive predicate symbol we use is $=$. We will use $\wedge$, $\vee$, $\supset$, $\forall$ and $\exists$ as primitive logical symbols. Atomic formulas are expressions of the form $a = b$, and formulas are constructed from atomic formulas by using the above logical symbols. We use A, B, C etc. as meta variables for formulas. We define $\bot$ and $\neg A$ as abbreviations of $\mathsf{succ}(0) = 0$ and $A \supset \bot$, respectively.

We will call an atomic formula of the form $x = k$ a *binding*. A binding $x = k$ binds the variable x to the natural number k. We define an *environment* as a finite set of bindings such that for any variable x if $x = k$ and $x = k'$ are both in the set then $k \equiv k'$ (that is, k and k' are the same natural number). We use ϵ, δ etc. as meta variables for environments. For an environment ϵ and a variable x, we define $\epsilon.x$ to be k if $x = k \in \epsilon$ and undefined if no such k exists. Let ϵ be an environment and $x = k$ be a binding. Then we define an environment $\epsilon[x = k]$ as follows.

$$\epsilon[x = k] \triangleq \begin{cases} (\epsilon - \{x = k'\}) \cup \{x = k\} & \text{if } x = k' \in \epsilon \text{ for some } k' \\ \epsilon \cup \{x = k\} & \text{otherwise.} \end{cases}$$

Let a be a term and ϵ be an environment. We say that ϵ *covers* a if $\mathrm{FV}(a) \subseteq \mathrm{dom}(\epsilon)$, where $\mathrm{FV}(a)$ stands for the set of free variables in a and $\mathrm{dom}(\epsilon)$ is the set of variables x such that $\epsilon.x$ is defined. We define the concept that an environment ϵ *covers* a formula A similarly.

Given an environment ϵ and a term a such that ϵ covers a, we can associate a natural number $\epsilon[\![a]\!]$ by induction on a as follows. We will say that a denotes $\epsilon[\![a]\!]$ in the environment ϵ.

1. $\epsilon[\![0]\!] \triangleq 0$.
2. $\epsilon[\![x]\!] \triangleq \epsilon.x$.
3. $\epsilon[\![\mathsf{succ}(a)]\!] \triangleq \mathsf{succ}(\epsilon[\![a]\!])$.
4. $\epsilon[\![\mathsf{plus}(a,b)]\!] \triangleq \epsilon[\![a]\!] + \epsilon[\![b]\!]$.
5. $\epsilon[\![\mathsf{times}(a,b)]\!] \triangleq \epsilon[\![a]\!] \cdot \epsilon[\![b]\!]$.

We have just seen that a term, which is a syntactic entity, denotes a natural number, which is an abstract mathematical object, under any environment that covers the term.

Similarly, we wish to define the denotation of a formula under an environment that covers the formula. We first define *propositions* as certain sets and then we define the classical BHK interpretation in such a way that each formula will denote a proposition.

In the classical BHK interpretation, we will use the term 'function' in the same way as we use it in classical mathematics based on set theory. Namely, by a *function* we understand a set f of pairs such that for any objects a, b, c, if (a, b) and (a, c) are in f then $b = c$. For a pair p, we write $\pi_0(p)$ ($\pi_1(p)$) for the left (right) component of p, respectively. If f is a function, we put $\mathrm{dom}(f) \triangleq \{\pi_0(p) \mid p \in f\}$, and if $a \in \mathrm{dom}(f)$, then we write $f(a)$ for the unique b such that $(a, b) \in f$. If S is a set and e is a mathematical expression (possibly) containing x such that e denotes a unique object for any x in S, then we write $\lambda x \in S.\ e$

for the function f such that $\mathrm{dom}(f) = S$ and $f(x) = e$ for any x in S. Also, we will write $f : S \to T$ if f is a function such that $\mathrm{dom}(f) = S$ and $f(p) \in T$ for any $p \in S$.

We introduce the following notation. Suppose that $p \in S$ and $f : \mathsf{N} \to S \to S$. Then we define a function $rec(p, f) : \mathsf{N} \to S$ by the following equations.

$$rec(p, f)(0) = p$$
$$rec(p, f)(\mathsf{succ}(n)) = f(n)(rec(p, f)(n))$$

Propositions are inductively defined as follows.

1. For each natural number k, the singleton set $\{k\}$ is a proposition.
2. The empty set $\emptyset$ is a proposition.
3. If S and T are propositions, then $S \times T = \{(p, q) \mid p \in S \text{ and } q \in T\}$ is a proposition.
4. If S and T are propositions, then $S + T = \{(0, p) \mid p \in S\} \cup \{(1, q) \mid q \in T\}$ is a proposition.
5. If S and T are propositions, then $S \to T$, the set of functions from S to T, is a proposition.
6. If S_k is a proposition for each natural number k, then $\prod_k S_k$, the set of functions f from N such that $f(k) \in S_k$ for each $k \in \mathsf{N}$, is a proposition.
7. If S_k is a proposition for each natural number k, then $\sum_k S_k = \{(k, p) \mid p \in S_k\}$, is a proposition.

We write Prop for the set of propositions. We are thus following the principle of *propositions-as-sets* since we defined propositions as sets.

Given an environment ϵ and a formula A such that ϵ covers A, we can associate a proposition $\epsilon[\![A]\!]$ by induction on A as follows. We will call this interpretation *the classical BHK interpretation.*

1. $\epsilon[\![a = b]\!] \triangleq \{\epsilon[\![a]\!]\} \cap \{\epsilon[\![b]\!]\}$.
2. $\epsilon[\![A \wedge B]\!] \triangleq \epsilon[\![A]\!] \times \epsilon[\![B]\!]$.
3. $\epsilon[\![A \vee B]\!] \triangleq \epsilon[\![A]\!] + \epsilon[\![B]\!]$.
4. $\epsilon[\![A \supset B]\!] \triangleq \epsilon[\![A]\!] \to \epsilon[\![B]\!]$.
5. $\epsilon[\![\forall x.A]\!] \triangleq \prod_k \epsilon[x = k][\![A]\!]$.
6. $\epsilon[\![\exists x.A]\!] \triangleq \sum_k \epsilon[x = k][\![A]\!]$.

We have the following theorem which establishes a logical equivalence between the classical BHK interpretation and the Tarskian interpretation.

Theorem 1. *If an environment ϵ covers a formula A, then A is true in ϵ under the Tarskian interpretation if and only if the proposition $\epsilon[\![A]\!]$ is a non-empty set.*

Proof. The proof is straightforward if we note the following equivalences.

- $\{k\} \cap \{l\} \neq \emptyset \iff k = l$.
- $S \times T \neq \emptyset \iff S \neq \emptyset$ and $T \neq \emptyset$.
- $S + T \neq \emptyset \iff S \neq \emptyset$ or $T \neq \emptyset$.
- $S \to T \neq \emptyset \iff S = \emptyset$ or $T \neq \emptyset$.
- $\prod_k S_k \neq \emptyset \iff S_k \neq \emptyset$ for all k.
- $\sum_k S_k \neq \emptyset \iff S_k \neq \emptyset$ for some k.

See also exercise 1.3.4 of Troelstra and van Dalen [12]. □

We will call elements of propositions *proofs*. Then, the above theorem says that a formula A is true under ϵ if and only if the proposition $\epsilon[\![A]\!]$ has a proof.

For any proposition S, we put $\pi(S)$ to be $\top$ (true) if S is non-empty and to be $\bot$ (false) if S is empty. We write $\epsilon[A]$ for the Tarskian truth value of A in ϵ. Then, the above theorem can be given as the equation:

$$\epsilon[A] = \pi(\epsilon[\![A]\!])$$

In this way, we can decompose the Tarskian interpretation as the composition of π and the BHK interpretation.

3 Interpretation of derivations

In the previous section, we introduced propositions and proofs as abstract mathematical entities (or semantical objects). In this section we define derivations as syntactic objects that are intended to denote propositions. So, after defining derivations, we will define an interpretation of a derivation in an environment by a proof. We also define reduction rules (or computation rules) for derivations and show that if a derivation reduces to another derivation, then they both denote the same proof (in any environment). In summary, we have the following table:

syntactic objects	semantic objects
term (a, b, c)	natural number (k, m, n)
formula (A, B, C)	proposition (S, T, U)
derivation (P, Q, R)	proof (p, q, r)
context (Γ, Δ)	environment (ϵ)

We define derivations in a natural deduction style. We will give inductive rules that are used to derive judgments. A *judgment* is either of the form $\Gamma \vdash a : \mathsf{N}$, $\Gamma \vdash A : \mathsf{Prop}$ or $\Gamma \vdash P : A$ where Γ is a *context*, a is a *term*, P is a *derivation* and A is a *formula*. In this way, we will define judgments, contexts and derivations simultaneously (as well as terms and formulas).

We will define a context as a finite set of declarations, where a *declaration* is either of the form x or of the form y^A (A, a formula). In the first case, we say that x is declared as a natural number and in the second case, we say that y is declared as a derivation of A. Each context will satisfy the condition that for

any variable x at most one declaration for x is in the context. Below, we display a context by listing its declarations as a sequence. So the empty econtext is displayed as an (invisible) empty sequence. If Γ and Δ are contexts, then Γ, Δ will stand for the union of the two contexts, and $\Gamma - x$ ($\Gamma - y^A$) will stand for the context obtained from Γ by removing the declaration x (y^A), respectively. Moreover, when we use the notation Γ, Δ we tacitly assume that Γ and Δ are *compatible*, that is $\Gamma \cup \Delta$ contains at most one declaration for each variable x. If $\Gamma \vdash P : A$ is derivable, then we will say that P *has type* A *under the context* Γ. We say that a context Γ is *smaller* than another context Δ if $\Gamma \subseteq \Delta$.

First, we give rules for terms.

1. If x is a variable, then $x \vdash x : \mathsf{N}$.
2. $\vdash 0 : \mathsf{N}$.
3. If $\Gamma \vdash a : \mathsf{N}$, then $\Gamma \vdash \mathsf{succ}(a) : \mathsf{N}$.
4. If $\Gamma \vdash a : \mathsf{N}$ and $\Delta \vdash b : \mathsf{N}$, then $\Gamma, \Delta \vdash \mathsf{plus}(a, b) : \mathsf{N}$.
5. If $\Gamma \vdash a : \mathsf{N}$ and $\Delta \vdash b : \mathsf{N}$, then $\Gamma, \Delta \vdash \mathsf{times}(a, b) : \mathsf{N}$.

We say that a is a *term* if a judgment of the form $\Gamma \vdash a : \mathsf{N}$ is derivable by using the above rules. We also define $\mathrm{FV}(a)$ as the set of variables x such that x occurs in Γ.

Next, we have rules for formulas.

1. If $\Gamma \vdash a : \mathsf{N}$ and $\Delta \vdash b : \mathsf{N}$, then $\Gamma, \Delta \vdash a = b : \mathsf{Prop}$.
2. If $\Gamma \vdash A : \mathsf{Prop}$ and $\Delta \vdash B : \mathsf{Prop}$, then $\Gamma, \Delta \vdash A \wedge B : \mathsf{Prop}$.
3. If $\Gamma \vdash A : \mathsf{Prop}$ and $\Delta \vdash B : \mathsf{Prop}$, then $\Gamma, \Delta \vdash A \vee B : \mathsf{Prop}$.
4. If $\Gamma \vdash A : \mathsf{Prop}$ and $\Delta \vdash B : \mathsf{Prop}$, then $\Gamma, \Delta \vdash A \supset B : \mathsf{Prop}$.
5. If x is a variable and $\Gamma \vdash A : \mathsf{Prop}$, then $\Gamma - x \vdash \forall x.A : \mathsf{Prop}$.
6. If x is a variable and $\Gamma \vdash A : \mathsf{Prop}$, then $\Gamma - x \vdash \exists x.A : \mathsf{Prop}$.

We say that A is a *formula* if a judgment of the form $\Gamma \vdash A : \mathsf{Prop}$ is derivable by using the above rules. We also define $\mathrm{FV}(A)$ as the set of variables x such that x occurs in Γ. We define substitution operation on formulas as usual, and we write $A[x = a]$ for the result of substituting a for all the free occurrences of x in A (after appropriately renaming bound variables in A if necessary).

Finally we give rules for derivations. We give these rules as inference rules. We first give general rules for equality.

$$\frac{\Gamma \vdash a : \mathsf{N}}{\Gamma \vdash \mathsf{id}(a) : a = a}\ (\mathsf{id}) \qquad \frac{\Gamma \vdash P : a = b \quad \Delta \vdash Q : A[x = a]}{\Gamma, \Delta \vdash \mathsf{repl}_{A[x=b]}(P, Q) : A[x = b]}\ (\mathsf{repl})$$

The rules specific to natural numbers are as follows.

$$\frac{\Gamma \vdash P : a = b}{\Gamma \vdash \mathsf{succ}(P) : \mathsf{succ}(a) = \mathsf{succ}(b)}\ (\mathsf{succ})$$

$$\frac{\Gamma \vdash P : \mathsf{succ}(a) = 0 \quad \Delta \vdash A : \mathsf{Prop}}{\Gamma, \Delta \vdash \mathsf{abort}_A(P) : A}\ (\bot E)$$

$$\frac{\Gamma \vdash b : \mathsf{N}}{\Gamma \vdash \mathsf{plus}(0,b) : \mathsf{plus}(0,b) = b}\ (\mathsf{plus}_0)$$

$$\frac{\Gamma \vdash a : \mathsf{N} \quad \Delta \vdash b : \mathsf{N}}{\Gamma,\Delta \vdash \mathsf{plus}(\mathsf{succ}(a),b) : \mathsf{plus}(\mathsf{succ}(a),b) = \mathsf{succ}(\mathsf{plus}(a,b))}\ (\mathsf{plus}_1)$$

$$\frac{\Gamma \vdash b : \mathsf{N}}{\Gamma \vdash \mathsf{times}(0,b) : \mathsf{times}(0,b) = 0}\ (\mathsf{times}_0)$$

$$\frac{\Gamma \vdash a : \mathsf{N} \quad \Delta \vdash b : \mathsf{N}}{\Gamma,\Delta \vdash \mathsf{times}(\mathsf{succ}(a),b) : \mathsf{times}(\mathsf{succ}(a),b) = \mathsf{plus}(b,\mathsf{times}(a,b))}\ (\mathsf{times}_1)$$

$$\frac{\Gamma \vdash P : A[x=0] \quad \Delta \vdash Q : \forall x.(A \supset A[x = \mathsf{succ}(x)])}{\Gamma,\Delta \vdash \mathsf{rec}(P,Q) : \forall x.A}\ (\mathsf{ind})$$

The rules for logical symbols are as follows.

$$\frac{\Gamma \vdash A : \mathsf{Prop}}{x^A, \Gamma \vdash x^A : A}\ (\mathsf{assume}) \qquad \frac{\Gamma \vdash P : A \quad \Delta \vdash Q : B}{\Gamma,\Delta \vdash \mathsf{pair}(P,Q) : A \wedge B}\ (\wedge I)$$

$$\frac{\Gamma \vdash P : A \wedge B}{\Gamma \vdash \mathsf{car}(P) : A}\ (\wedge E_1) \qquad \frac{\Gamma \vdash P : A \wedge B}{\Gamma \vdash \mathsf{cdr}(P) : B}\ (\wedge E_2)$$

$$\frac{\Gamma \vdash P : A \quad \Delta \vdash B : \mathsf{Prop}}{\Gamma,\Delta \vdash \mathsf{inl}_B(P) : A \vee B}\ (\vee I_1) \qquad \frac{\Gamma \vdash P : B \quad \Delta \vdash A : \mathsf{Prop}}{\Gamma,\Delta \vdash \mathsf{inr}_A(P) : A \vee B}\ (\vee I_2)$$

$$\frac{\Gamma \vdash P : A \vee B \quad \Delta \vdash Q : C \quad \Pi \vdash R : C}{\Gamma,\Delta - x^A,\Pi - y^B \vdash \mathsf{case}(P,x^A.Q,y^B.R) : C}\ (\vee E)$$

$$\frac{\Gamma \vdash P : B \quad \Delta \vdash A : \mathsf{Prop}}{\Gamma - x^A,\Delta \vdash \lambda x^A.P : A \supset B}\ (\supset I) \qquad \frac{\Gamma \vdash P : A \supset B \quad \Delta \vdash Q : A}{\Gamma,\Delta \vdash \mathsf{apply}(P,Q) : B}\ (\supset E)$$

$$\frac{\Gamma \vdash P : A}{\Gamma - x \vdash \lambda x.P : \forall x.A}\ (\forall I) \qquad \frac{\Gamma \vdash P : \forall x.A \quad \Delta \vdash a : \mathsf{N}}{\Gamma,\Delta \vdash \mathsf{apply}(P,a) : A[x=a]}\ (\forall E)$$

$$\frac{\Gamma \vdash a : \mathsf{N} \quad \Delta \vdash P : A[x=a]}{\Gamma,\Delta \vdash \mathsf{pair}_{\exists x.A}(a,P) : \exists x.A}\ (\exists I) \qquad \frac{\Gamma \vdash P : \exists x.A \quad \Delta \vdash Q : C}{\Gamma,\Delta - x - y^A \vdash \mathsf{split}(P,(x,y^A).Q) : C}\ (\exists E)$$

We say that P is an *intuitionistic derivation* if a judgment of the form $\Gamma \vdash P : C$ is derivable by using the above rules. If x is in Γ, we say that the derivation P depends on the *assumption* x^A.

We now add the following rule which formalizes *the law of the excluded middle*. A derivation which is obtained by possibly using this rule in addition to the other rules is called a *classical derivation.*

$$\frac{\Gamma \vdash A : \mathsf{Prop}}{\Gamma \vdash \mathsf{lem}_A : A \vee \neg A}\ (\mathsf{lem})$$

We can verify that any formula A is derivable in **HA** (Heyting arithmetic) iff $\vdash P : A$ is intuitionistically derivable, and A is derivable in **PA** (Peano arithmetic) iff $\vdash P : A$ is classically derivable.

We have thus defined three kinds of judgments and defined how to derive these judgments. It is easy to see that if a judgment of the first kind $\Gamma \vdash a : \mathsf{N}$ or a judgment of the second kind $\Gamma \vdash A : \mathsf{Prop}$ is derivable, then Γ is of the form $x_1 \ldots, x_n$. This means that a term a or a formula A depends on the variables $x_1, \ldots, x_n$. Therefore, it is in general necessary that ϵ covers these variables in order that $\epsilon[\![a]\!]$ and $\epsilon[\![A]\!]$ are definable.

We can easily verify the following lemma by induction on the construction of derivations.

Lemma 2. *If* $\Gamma \vdash a : \mathsf{N}$ $(\Gamma \vdash A : \mathsf{Prop})$ *is derivable and* ϵ *covers* Γ, *then* $\epsilon[\![a]\!] \in \mathsf{N}$ $(\epsilon[\![A]\!] \in \mathsf{Prop})$, *respectively.*

If a judgment of the third kind $\Gamma \vdash P : C$ is derivable, then each element of the sequence Γ is either of the form x or of the form x^A where A is a formula and we write $\mathrm{dom}(\Gamma)$ for the set of such x's. In order to interpret such a context, we need to extend the definition of binding and environment as follows. A *binding* is an expression of the form $x = k$ where k is a natural number or of the form $x = p$ where p is a proof. An *environment* is a finite set of bindings such that for each variable x there exists at most one binding of the form $x = k$ or $x = p$ in the set. For an environment ϵ and a variable x, $\epsilon.x$ is defined in the same way as before and we write $\mathrm{dom}(\epsilon)$ for the set of variables x such that $\epsilon.x$ is defined. Let ϵ be an environment and Γ be a context. We write $\epsilon \models \Gamma$ if $\mathrm{dom}(\Gamma) \subseteq \mathrm{dom}(\epsilon)$, $\epsilon.x \in \mathsf{N}$ for each x in Γ, and $\epsilon.x \in \epsilon[\![A]\!]$ for each x^A in Γ.

Suppose that $\Gamma \vdash P : A$ is derivable and $\epsilon \models \Gamma$, then we can define $\epsilon[\![P]\!]$ and show that $\epsilon[\![P]\!] \in \epsilon[\![A]\!]$ as follows. We first define *pre-derivations* by the following grammar.

$$\begin{aligned} P ::= {} & x \mid \mathsf{id}(a) \mid \mathsf{repl}(P, Q) \mid \mathsf{succ}(P) \mid \mathsf{abort}(P) \mid \mathsf{rec}(P, Q) \\ & \mid \mathsf{plus}(0, b) \mid \mathsf{plus}(\mathsf{succ}(a), b) \mid \mathsf{times}(0, b) \mid \mathsf{times}(\mathsf{succ}(a), b) \\ & \mid \mathsf{pair}(P, Q) \mid \mathsf{pair}(a, P) \mid \mathsf{car}(P) \mid \mathsf{cdr}(P) \mid \mathsf{split}(P, (x, y).Q) \\ & \mid \mathsf{inl}(P) \mid \mathsf{inr}(P) \mid \mathsf{case}(P, x.Q, y.R) \\ & \mid \lambda x^A.P \mid \lambda x.P \mid \mathsf{apply}(P, Q) \mid \mathsf{apply}(P, a) \\ & \mid \mathsf{lem}_A \end{aligned}$$

For each pre-derivation P we can define the set $\mathrm{FV}(P)$ of free variables in P as expected. For each pre-derivation P and environment ϵ such that $\mathrm{FV}(P) \subseteq \mathrm{dom}(\epsilon)$ we define a proof $\epsilon[\![P]\!]$ inductively as follows.

1. $\epsilon[\![x]\!] \triangleq \epsilon.x$.
2. $\epsilon[\![\mathsf{id}(a)]\!] \triangleq \epsilon[\![a]\!]$.
3. $\epsilon[\![\mathsf{repl}(P,Q)]\!] \triangleq \epsilon[\![Q]\!]$.
4. $\epsilon[\![\mathsf{succ}(P)]\!] \triangleq \epsilon[\![P]\!]$.
5. $\epsilon[\![\mathsf{abort}(P)]\!] \triangleq$ undefined.
6. $\epsilon[\![\mathsf{rec}(P,Q)]\!] \triangleq rec(\epsilon[\![P]\!], \epsilon[\![Q]\!])$.
7. $\epsilon[\![\mathsf{plus}(0,b)]\!] \triangleq \epsilon[\![b]\!]$.
8. $\epsilon[\![\mathsf{plus}(\mathsf{succ}(a),b)]\!] \triangleq \epsilon[\![a]\!] + 1 + \epsilon[\![b]\!]$.
9. $\epsilon[\![\mathsf{times}(0,b)]\!] \triangleq 0$.
10. $\epsilon[\![\mathsf{times}(\mathsf{succ}(a),b)]\!] \triangleq (\epsilon[\![a]\!] + 1)\epsilon[\![b]\!]$.
11. $\epsilon[\![\mathsf{pair}(P,Q)]\!] \triangleq (\epsilon[\![P]\!], \epsilon[\![Q]\!])$.
12. $\epsilon[\![\mathsf{pair}(a,P)]\!] \triangleq (\epsilon[\![a]\!], \epsilon[\![P]\!])$.
13. $\epsilon[\![\mathsf{car}(P)]\!] \triangleq \pi_0(\epsilon[\![P]\!])$.
14. $\epsilon[\![\mathsf{cdr}(P)]\!] \triangleq \pi_1(\epsilon[\![P]\!])$.
15. $\epsilon[\![\mathsf{split}(P,(x,y).Q)]\!] \triangleq \epsilon[x=k][y=p][\![Q]\!]$, if $\epsilon[\![P]\!] = (k,p)$.
16. $\epsilon[\![\mathsf{inl}(P)]\!] \triangleq (0, \epsilon[\![P]\!])$.
17. $\epsilon[\![\mathsf{inr}(P)]\!] \triangleq (1, \epsilon[\![P]\!])$.
18. $\epsilon[\![\mathsf{case}(P,x.Q,y.R)]\!] \triangleq \begin{cases} \epsilon[x=p][\![Q]\!] \text{ if } \epsilon[\![P]\!] = (0,p), \\ \epsilon[y=p][\![R]\!] \text{ if } \epsilon[\![P]\!] = (1,p). \end{cases}$
19. $\epsilon[\![\lambda x^A.P]\!] \triangleq \lambda p \in \epsilon[\![A]\!].\epsilon[x=p][\![P]\!]$.
20. $\epsilon[\![\lambda x.P]\!] \triangleq \lambda k \in \mathsf{N}.\epsilon[x=k][\![P]\!]$.
21. $\epsilon[\![\mathsf{apply}(P,Q)]\!] \triangleq \epsilon[\![P]\!](\epsilon[\![Q]\!])$.
22. $\epsilon[\![\mathsf{apply}(P,a)]\!] \triangleq \epsilon[\![P]\!](\epsilon[\![a]\!])$.
23. $\epsilon[\![\mathsf{lem}_A]\!] \triangleq \begin{cases} (0,p) \text{ if } p \in \epsilon[\![A]\!], \\ (1,\emptyset) \text{ if } \epsilon[\![A]\!] = \emptyset. \end{cases}$

We note that $\epsilon[\![P]\!]$ is sometimes undefined even if $\mathrm{FV}(P) \subseteq \mathrm{dom}(\epsilon)$. For example, $\epsilon[\![\mathsf{car}(\mathsf{id}(0))]\!]$ is undefined since $\pi_0(0)$ is undefined. We also note that $\epsilon[\![\mathsf{lem}_A]\!]$ may not be unique if $\epsilon[\![A]\!]$ is non-empty. Even in such a case we can make the definition unique by using a choice function that selects an element from the set $\epsilon[\![A]\!]$.

Theorem 3. *If $\Gamma \vdash P : A$ is derivable and $\epsilon \models \Gamma$, then $\epsilon[\![P]\!] \in \epsilon[\![A]\!]$.*

Proof. We first remark that the derivation P may not be a legitimate pre-derivation, but by forgetting its subscripts and/or superscripts appropriately, we can uniquely translate P into a pre-derivation, say, Q. We therefore understand that $\epsilon[\![P]\!]$ in the statement of the theorem stands for $\epsilon[\![Q]\!]$.

We prove the theorem by induction on the derivation of $\Gamma \vdash P : A$. Here we check only the following case. Suppose that $\Gamma, \Delta \vdash \mathsf{abort}_A(P) : A$ is derived

from $\Gamma \vdash P : \mathsf{succ}(a) = 0$ and $\Delta \vdash A : \mathsf{Prop}$ and that $\epsilon \models \Gamma, \Delta$. Then, since $\epsilon \models \Gamma$, we have by induction hypothesis that $\epsilon[\![P]\!] \in \epsilon[\![\mathsf{succ}(a) = 0]\!]$. Here we have $\epsilon[\![\mathsf{succ}(a) = 0]\!] = \emptyset$ since $\mathsf{succ}(\epsilon[\![a]\!]) \neq 0$. This is a contradiction, and we see that there can be no ϵ such that $\epsilon \models \Gamma, \Delta$. So the theorem holds in this case. □

We now consider reduction rules for pre-derivations.

1. $\mathsf{apply}(\mathsf{rec}(P, Q), 0) \mapsto P$.
2. $\mathsf{apply}(\mathsf{rec}(P, Q), \mathsf{succ}(a)) \mapsto \mathsf{apply}(\mathsf{apply}(Q, a), \mathsf{apply}(\mathsf{rec}(P, Q), a))$.
3. $\mathsf{car}(\mathsf{pair}(P, Q)) \mapsto P$.
4. $\mathsf{cdr}(\mathsf{pair}(P, Q)) \mapsto Q$.
5. $\mathsf{case}(\mathsf{inl}(P), x.Q, y.R) \mapsto Q[x = P]$.
6. $\mathsf{case}(\mathsf{inr}(P), x.Q, y.R) \mapsto R[y = P]$.
7. $\mathsf{apply}(\lambda x^A.P, Q) \mapsto P[x = Q]$.
8. $\mathsf{apply}(\lambda x.P, a) \mapsto P[x = a]$.
9. $\mathsf{split}(\mathsf{pair}(P, Q), (x, y).R) \mapsto R[x = P][y = Q]$.

These reductions enjoy the following subject reduction property.

Theorem 4. *If P has type A under Γ and $P \mapsto Q$, then Q has type A under some Δ smaller than Γ.*

Moreover, we have the following theorem which shows that BHK interpretation is preserved by reductions. We may read this theorem as saying that if a derivation is reducible to another derivation, then although they are syntactically distinct, they both denote the same proof.

Theorem 5. *Suppose that $\Gamma \vdash P : A$ is derivable, $\epsilon \models \Gamma$ and $P \mapsto Q$. Then we have $\epsilon[\![P]\!] = \epsilon[\![Q]\!]$.*

Proof. We check here only the following case.

$$\begin{aligned}
\epsilon[\![\mathsf{apply}(\lambda x^A.P, Q)]\!] &= \epsilon[\![\lambda x^A.P]\!](\epsilon[\![Q]\!]) \\
&= (\lambda q \in \epsilon[\![A]\!].\epsilon[x = q][\![P]\!])(\epsilon[\![Q]\!]) \\
&= \epsilon[x = \epsilon[\![Q]\!]][\![P]\!] \\
&= \epsilon[\![P[x = Q]]\!]
\end{aligned}$$

The last equality above comes from Lemma 8, which in turn is based on Lemma 6 and Lemma 7. □

We need the following lemmas to prove the above theorem.

Lemma 6. *If $\Gamma \vdash a : \mathsf{N}$ and $\Delta \vdash b : \mathsf{N}$ are derivable, then $\Gamma - x, \Delta \vdash a[x = b] : \mathsf{N}$ is also derivable, and for any ϵ such that $\epsilon \models \Gamma - x, \Delta$, we have $\epsilon[\![a[x = b]]\!] = \epsilon[x = \epsilon[\![b]\!]][\![a]\!]$.*

Lemma 7. *If $\Gamma \vdash P : A$ and $\Delta \vdash a : \mathsf{N}$ are derivable, then $\Gamma - x, \Delta \vdash P[x = a] : A[x = a]$ is also derivable, and for any ϵ such that $\epsilon \models \Gamma - x, \Delta$, we have $\epsilon[\![P[x = a]]\!] = \epsilon[x = \epsilon[\![a]\!]][\![P]\!]$ and $\epsilon[\![A[x = a]]\!] = \epsilon[x = \epsilon[\![a]\!]][\![A]\!]$.*

Lemma 8. *If $\Gamma \vdash P : A$ and $\Delta \vdash Q : B$ are derivable, then $\Gamma - x : B, \Delta \vdash P[x = Q] : A$ is also derivable, and for any ϵ such that $\epsilon \models \Gamma - x, \Delta$, we have $\epsilon[\![P[x = Q]]\!] = \epsilon[x = \epsilon[\![Q]\!]][\![P]\!]$.*

4 The catch-throw calculus

In this section we extend **HA** by adding inference rules which correspond to the catch and throw mechanism used in programming language like Lisp. We will write $\mathbf{PA}_{c/t}$ for the extended calculus. $\mathbf{PA}_{c/t}$ is logically equivalent to **PA**. Such logical calculi were first proposed by Nakano [5, 6] and later modified by Sato [11] and Kameyama [3]. Here we use the inference rules introduced in [11], and we refer the reader to [11, 5] for detailed explanations of the motivations behind these inference rules.

We first extend the language by assuming that there are denumerably many *tag variables* that are used as tags for derivations. We will use u, v, w etc. as meta variables for tag variables. We define a *tag context* as a finite set of declarations of the form u^E where u is a tag variable and E is a formula. If Γ is a context and Δ is a tag context, then the pair (Γ, Δ), which we write $\Gamma; \Delta$ is said to be an *e-context* (extended context).

In $\mathbf{PA}_{c/t}$, we will derive judgments of the form $\Gamma \vdash P : A; \Delta$ where Γ is a context and Δ is a tag context. If $\Gamma \vdash P : A; \Delta$ is derivable, then we will say that P *has type* A *under* $\Gamma; \Delta$. We say that an e-context $\Gamma; \Delta$ is *smaller* than another context $\Pi; \Sigma$ if $\Gamma \subseteq \Pi$ and $\Delta \subseteq \Sigma$. If Δ is empty, then we will write $\Gamma \vdash P : A$ for $\Gamma \vdash P : A; \Delta$.

As we will see later, the reduction rules of derivations are not confluent. To cope with this situation, we will interpret a derivation in an environment not by a proof but by a set of proofs.

Now, $\mathbf{PA}_{c/t}$ is obtained from **HA** by adding the following two rules.

$$\frac{\Gamma \vdash P : E; \Delta \quad \Pi \vdash A : \mathsf{Prop}}{\Gamma, \Pi \vdash !_A u^E(P) : A; \Delta, u^E}\ (\mathsf{throw}) \qquad \frac{\Gamma \vdash P : A; \Delta}{\Gamma \vdash ?u^E.\ P : A \vee E; \Delta - u^E}\ (\mathsf{catch})$$

Since a judgment of $\mathbf{PA}_{c/t}$ contains a tag context in general, we have to modify inference rules of **HA** as well. Tag contexts in the premises are always inherited in the conclusion of any inference rule. So, for example, the $(\wedge I)$ rule becomes:

$$\frac{\Gamma \vdash P : A; \Delta \quad \Pi \vdash Q : B; \Sigma}{\Gamma, \Delta \vdash \mathsf{pair}(P, Q) : A \wedge B; \Delta, \Sigma}\ (\wedge I)$$

and we understand that other rules of **HA** are similarly modified to those of $\mathbf{PA}_{c/t}$, except the following rules which we write down explicitly. These rules are also extensions of the corresponding rules in **HA**, but we restrict tag contexts in some judgments to be empty.

$$\frac{\Gamma \vdash P : a = b \quad \Pi \vdash Q : A[x = a]; \Sigma}{\Gamma, \Pi \vdash \mathsf{repl}_{A[x=b]}(P, Q) : A[x = b]; \Sigma}\ (\mathsf{repl})$$

$$\frac{\Gamma \vdash P : A[x = 0] \quad \Delta \vdash Q : \forall x.(A \supset A[x = \mathsf{succ}(x)])}{\Gamma, \Delta \vdash \mathsf{rec}(P, Q) : \forall x.A} \ (\mathsf{ind})$$

$$\frac{\Gamma \vdash P : A \supset B; \Delta \quad \Pi \vdash Q : A}{\Gamma, \Pi \vdash \mathsf{apply}(P, Q) : B; \Delta} \ (\supset E)$$

$$\frac{\Gamma \vdash P : A \vee B \quad \Pi \vdash Q : C; \Sigma \quad \Phi \vdash R : C; \Psi}{\Gamma, \Pi - x^A, \Phi - y^B \vdash \mathsf{case}(P, x^A.Q, y^B.R) : C; \Sigma, \Psi} \ (\vee E)$$

$$\frac{\Gamma \vdash P : \exists x.A \quad \Pi \vdash Q : C; \Sigma}{\Gamma, \Pi - x - y^A \vdash \mathsf{split}(P, (x, y^A).Q) : C; \Sigma} \ (\exists E)$$

We can show that $\mathbf{PA}_{c/t}$ is logically equivalent to $\mathbf{PA}$ by using Theorem 4 in [11]. In particular, we can derive the law of the excluded middle in $\mathbf{PA}_{c/t}$ as follows.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dots}{\Gamma \vdash A : \mathsf{Prop}}\ (\mathsf{assume})}{x^A, \Gamma \vdash x^A : A}\ (\mathsf{throw})}{x^A, \Gamma \vdash !_\perp u^A(x^A) : \perp; u^A}\ (\supset I)}{\Gamma \vdash \lambda x^A.!_\perp u^A(x^A) : \neg A; u^A}\ (\mathsf{catch})}{\Gamma \vdash ?u^A.\ \lambda x^A.!_\perp u^A(x^A) : \neg A \vee A;}$$

It is possible to consider a more liberal system $\mathbf{PA}^+_{c/t}$ which is the same as $\mathbf{PA}_{c/t}$ except that we have the following implication elimination rule instead of the $(\supset E)$ rule above.

$$\frac{\Gamma \vdash P : A \supset B; \Delta \quad \Pi \vdash Q : A; \Sigma}{\Gamma, \Pi \vdash \mathsf{apply}(P, Q) : B; \Delta, \Sigma} \ (\supset E)^+$$

The difference between these two implication elimination rules is that in $(\supset E)^+$ it is possible to apply a function to an argument that has free tag variables, while in $(\supset E)$ a function can be applied only to an argument that is tag variable free. We can also show that $\mathbf{PA}^+_{c/t}$ is logically equivalent to $\mathbf{PA}$ by the same argument. $\mathbf{PA}^+_{c/t}$ is therefore consistent, but we could not find a sound BHK interpretation for $\mathbf{PA}^+_{c/t}$.

We now wish to give an interpretation of judgments derivable in $\mathbf{PA}_{c/t}$. Suppose that $\Gamma \vdash P : A; \Delta$ is derivable in $\mathbf{PA}_{c/t}$. We will call such P an *e-derivation* (extended derivation) and $A; \Delta$ an *e-formula*. In order to interpret these syntactic entities, we will extend the notions of proposition and proof to those of e-proposition and e-proof. In summary, we have the following table.

syntactic objects	semantic objects
term (a)	natural number (k)
tag context (Δ)	proposition environment (θ)
e-formula ($A;\Delta$)	e-proposition ($S;\theta$)
e-derivation (P)	e-proof (p)

We begin with a preliminary discussion that is necessary to give our interpretation. First, we prepare a notation. Let u be a tag variable, p be a proof and U be a proposition, then we write up for the pair (u,p) and uU for the set $\{up \mid p \in U\}$.

Let S be a proposition and X be an arbitrary set. We define a set $S;X$ by induction on the construction of S as follows.

1. If $S = \emptyset$ or $S = \{k\}$, then $S;X \triangleq S \cup X$.
2. $S \times T;X \triangleq (S;X \times T;X) \cup X$.
3. $S + T;X \triangleq (S;X + T;X) \cup X$.
4. $S \to T;X \triangleq (S \to (T;X)) \cup X$.
5. $\prod_k S_k;X \triangleq (\prod_k (S_k;X)) \cup X$.
6. $\sum_k S_k;X \triangleq (\sum_k (S_k;X)) \cup X$.

We have the following lemma which we can verify by induction on the construction of U.

Lemma 9. *If U is a proposition and $X \subseteq Y$, then $U;X \subseteq U;Y$.*

Let θ be a function such that $\mathrm{dom}(\theta)$ is a finite set of tag variables and $\theta(u)$ is a proposition for any $u \in \mathrm{dom}(\theta)$. We call such θ a *proposition environment.* If θ is a proposition environment and u is a tag variable, then $\theta - u$ denotes the restriction of θ to $\mathrm{dom}(\theta) - \{u\}$.

Suppose that θ is a proposition environment such that $\mathrm{dom}(\theta) = \{u_1, \ldots, u_n\}$ where u_i are distinct and $U_i = \theta(u_i)$ $(1 \leq i \leq n)$. Then for any set X we associate another set $\varphi(X)$ by putting:

$$\varphi(X) \triangleq u_1(U_1;X) \cup \cdots \cup u_n(U_n;X),$$

and we define $|\theta|$ as the smallest fixed point of φ. Therefore, $|\theta|$ satisfies the following set equation.

$$|\theta| \triangleq u_1(U_1;|\theta|) \cup \cdots \cup u_n(U_n;|\theta|).$$

If S is a proposition and θ is a proposition environment, then we simply write $S;\theta$ for $S;|\theta|$ and call such a set an *e-proposition* (extended proposition). Elements of e-propositions will be called *e-proofs.* Let $p \in S;\theta$ be an e-proof. p is *exceptional* if $p \in |\theta|$ and *proper* otherwise. If $V \subseteq S;\theta$, we put $V_{\mathsf{e}} \triangleq \{p \in V \mid p \text{ is exceptional}\}$ and $V_{\mathsf{p}} \triangleq \{p \in V \mid p \text{ is proper}\}$.

Let r be an e-proof in $S;\theta$. We define a set $!(r) \subseteq |\theta|$ and a set $\mathrm{TV}(r)$ of tag variables simultaneously as follows. If $r \in |\theta|$, then r is of the form up where $u \in \mathrm{dom}(\theta)$ and $p \in \theta(u);\theta$, and in this case we put

$$!(r) \triangleq \begin{cases} \{up\} \cup !(p) & \text{if } u \notin \mathrm{TV}(p), \\ !(p) & \text{if } u \in \mathrm{TV}(p). \end{cases}$$

If $r \notin |\theta|$, then we define $!(r)$ inductively as follows.

1. If $S = \emptyset$ or $S = \{k\}$, then $r = k$ and we put $!(r) \triangleq \emptyset$.
2. If $r \in S;\theta \times T;\theta$, then $r = (p, q)$ and we put $!(r) \triangleq !(p) \cup !(q)$.
3. If $r \in S;\theta + T;\theta$, then $r = (0, p)$ or $r = (1, p)$ and we put $!(r) \triangleq !(p)$.
4. If $r \in S \to (T;\theta)$, then $!(r) \triangleq \bigcup_{p \in S} !(r(p))$.
5. If $r \in \prod_k (S_k;\theta)$, then $!(r) \triangleq \bigcup_k !(r(k))$.
6. If $r \in \sum_k (S_k;\theta)$, then $r = (k, p)$ and we put $!(r) \triangleq !(p)$.

We define $\mathrm{TV}(r) \subseteq \mathrm{dom}(\theta)$ by putting:

$$\mathrm{TV}(r) \triangleq \{u \in \mathrm{dom}(\theta) \mid up \in !(r) \text{ for some } p\}$$

If $\mathrm{TV}(r) = \emptyset$, then we say that r is *tag variable free* and if $u \in \mathrm{TV}(r)$, then we say that u *occurs in* $\mathrm{TV}(r)$. We have the following lemma.

Lemma 10. *If $r \in S;\theta$ and $u \notin \mathrm{TV}(r)$, then $r \in S;\theta - u$.*

Next let $r \in S;\theta$ and $u \in \mathrm{dom}(\theta)$. We define a set $r/u \subseteq \theta(u);\theta$ as follows. r/u will be said to be the u *component* of r.

$$r/u \triangleq \{p \mid up \in !(r)\}$$

It is easy to check that $u \in \mathrm{TV}(r)$ if and only if r/u is non-empty. We also have the following lemma which we can prove by using Lemma 10.

Lemma 11. $r/u \subseteq \theta(u);\theta - u$.

If an e-proof p is in the u component of an e-proof r, then it means that up is a possible exceptional value of r. So, unlike ordinary proofs we discussed in section 3, an e-proof denotes a set of its possible values.

We now define the catch and the throw operations on e-proofs. Let $r \in S;\theta$ be an e-proof and u be a tag variable. Then, we put

$$!(u, r) \triangleq !(ur)$$

and

$$?(u, r) \triangleq \begin{cases} \{(0, r)\} & \text{if } u \notin \mathrm{TV}(r), \\ \{(1, p) \mid p \in r/u\} & \text{if } u \in \mathrm{TV}(r). \end{cases}$$

By Lemma 11, we can see that $?(u,r) \subseteq S + U;\theta - u$ and if $r \in U;\theta$, then $!(u,r) \subseteq |\theta|$ where $U = \theta(u)$, For a set V of e-proofs, we put $!(u,V) \triangleq \bigcup_{p\in V} !(u,p)$ and $?(u,V) \triangleq \bigcup_{p\in V} ?(u,p)$.

Let Γ be a context, Δ be a tag context and ϵ be an environment. We define a proposition environment θ whose domain is $\mathrm{dom}(\Delta)$ by putting $\theta(u) \triangleq \epsilon[\![E]\!]$, where $u^E \in \Delta$. We will write $\epsilon[\![\Delta]\!]$ for this θ.

Now, suppose that the judgment $\Gamma \vdash P : A; \Delta$ is derivable in $\mathbf{PA}_{c/t}$. Then for any environment ϵ such that $\epsilon \models \Gamma$, we will define a set $\epsilon[\![P]\!]$ of e-proofs and will show that $\epsilon[\![P]\!]$ is non-empty and $\epsilon[\![P]\!] \subseteq \epsilon[\![A]\!];\epsilon[\![\Delta]\!]$.

1. $\epsilon[\![!u(P)]\!] \triangleq !(u, \epsilon[\![P]\!])$.
2. $\epsilon[\![?u.\ P]\!] \triangleq ?(u, \epsilon[\![P]\!])$.
3. $\epsilon[\![x]\!] \triangleq \{\epsilon.x\}$.
4. $\epsilon[\![\mathsf{id}(a)]\!] \triangleq \{\epsilon[\![a]\!]\}$.
5. $\epsilon[\![\mathsf{repl}(P,Q)]\!] \triangleq \epsilon[\![Q]\!]$
6. $\epsilon[\![\mathsf{succ}(P)]\!] \triangleq \{k+1 \mid k \in \epsilon[\![P]\!]_\mathsf{p}\} \cup \epsilon[\![P]\!]_\mathsf{e}$.
7. $\epsilon[\![\mathsf{abort}(P)]\!] \triangleq \epsilon[\![P]\!]$.
8. $\epsilon[\![\mathsf{rec}(P,Q)]\!] \triangleq \{rec(p,q) \mid p \in \epsilon[\![P]\!] \text{ and } q \in \epsilon[\![Q]\!]\}$.
9. $\epsilon[\![\mathsf{plus}(0,b)]\!] \triangleq \{\epsilon[\![b]\!]\}$.
10. $\epsilon[\![\mathsf{plus}(\mathsf{succ}(a),b)]\!] \triangleq \{\epsilon[\![a]\!] + 1 + \epsilon[\![b]\!]\}$.
11. $\epsilon[\![\mathsf{times}(0,b)]\!] \triangleq \{0\}$.
12. $\epsilon[\![\mathsf{times}(\mathsf{succ}(a),b)]\!] \triangleq \{(\epsilon[\![a]\!] + 1)\epsilon[\![b]\!]\}$.
13. $\epsilon[\![\mathsf{pair}(P,Q)]\!] \triangleq \{(p,q) \mid p \in \epsilon[\![P]\!] \text{ and } q \in \epsilon[\![Q]\!]\} \cup \epsilon[\![P]\!]_\mathsf{e} \cup \epsilon[\![Q]\!]_\mathsf{e}$.
14. $\epsilon[\![\mathsf{pair}(a,P)]\!] \triangleq \{(\epsilon[\![a]\!], p) \mid p \in \epsilon[\![P]\!]\} \cup \epsilon[\![P]\!]_\mathsf{e}$.
15. $\epsilon[\![\mathsf{car}(P)]\!] \triangleq \{\pi_0(p) \mid p \in \epsilon[\![P]\!]_\mathsf{p}\} \cup \epsilon[\![P]\!]_\mathsf{e}$.
16. $\epsilon[\![\mathsf{cdr}(P)]\!] \triangleq \{\pi_1(p) \mid p \in \epsilon[\![P]\!]_\mathsf{p}\} \cup \epsilon[\![P]\!]_\mathsf{e}$.
17. $\epsilon[\![\mathsf{split}(P,(x,y).Q)]\!] \triangleq \bigcup_{(p,q)\in\epsilon[\![P]\!]} \epsilon[x=p][y=q][\![Q]\!]$.
18. $\epsilon[\![\mathsf{inl}(P)]\!] \triangleq \{(0,p) \mid p \in \epsilon[\![P]\!]\} \cup \epsilon[\![P]\!]_\mathsf{e}$.
19. $\epsilon[\![\mathsf{inr}(P)]\!] \triangleq \{(1,p) \mid p \in \epsilon[\![P]\!]\} \cup \epsilon[\![P]\!]_\mathsf{e}$.
20. $\epsilon[\![\mathsf{case}(P,x.Q,y.R)]\!] \triangleq \bigcup_{(0,p)\in\epsilon[\![P]\!]} \epsilon[x=p][\![Q]\!] \cup \bigcup_{(1,p)\in\epsilon[\![P]\!]} \epsilon[y=p][\![R]\!]$.
21. $\epsilon[\![\lambda x^A.P]\!] \triangleq \{f \mid \mathrm{dom}(f) = \epsilon[\![A]\!] \text{ and } f(p) \in \epsilon[x=p][\![P]\!] \text{ for all } p \in \epsilon[\![A]\!]\} \cup \bigcup_{p\in\epsilon[\![A]\!]} \epsilon[x=p][\![P]\!]_\mathsf{e}$.
22. $\epsilon[\![\lambda x.P]\!] \triangleq \{f \mid \mathrm{dom}(f) = \mathsf{N} \text{ and } f(k) \in \epsilon[x=k][\![P]\!] \text{ for all } k \in \mathsf{N}\} \cup \bigcup_k \epsilon[x=k][\![P]\!]_\mathsf{e}$.
23. $\epsilon[\![\mathsf{apply}(P,Q)]\!] \triangleq \{p(q) \mid p \in \epsilon[\![P]\!]_\mathsf{p} \text{ and } q \in \epsilon[\![Q]\!]\} \cup \epsilon[\![P]\!]_\mathsf{e}$.
24. $\epsilon[\![\mathsf{apply}(P,a)]\!] \triangleq \{p(k) \mid p \in \epsilon[\![P]\!]_\mathsf{p} \text{ and } k \in \epsilon[\![a]\!]\} \cup \epsilon[\![P]\!]_\mathsf{e}$.

We have the following soundness theorem for $\mathbf{PA}_{c/t}$.

Theorem 12. *If $\Gamma \vdash P : A; \Delta$ is derivable in $\mathbf{PA}_{c/t}$, and $\epsilon \models \Gamma$, then $\epsilon[\![P]\!] \neq \emptyset$ and $\epsilon[\![P]\!] \subseteq \epsilon[\![A]\!]; \epsilon[\![\Delta]\!]$.*

Proof. By induction on the construction of derivations. We check typical cases only. Since the verification of the non-emptiness of $\epsilon[\![P]\!]$ is easy, we only verify that $\epsilon[\![P]\!] \subseteq \epsilon[\![A]\!]; \epsilon[\![\Delta]\!]$.

Case (throw): The rule we consider is as follows.

$$\frac{\Gamma \vdash P : E; \Delta \quad \Pi \vdash A : \mathsf{Prop}}{\Gamma, \Pi \vdash !_A u^E(P) : A; \Delta, u^E} \ (\mathsf{throw})$$

Suppose that $\epsilon \models \Gamma, \Pi$. Then $\epsilon \models \Gamma$ and by induction hypothesis, we have $\epsilon[\![P]\!] \subseteq \epsilon[\![E]\!]; \epsilon[\![\Delta]\!]$. So, if $p \in \epsilon[\![P]\!]$, then we have $p \in \epsilon[\![E]\!]; \epsilon[\![\Delta]\!] \subseteq \epsilon[\![E]\!]; \epsilon[\![\Delta, u^E]\!]$, so that $!(u,p) \subseteq \epsilon[\![\Delta, u^E]\!] \subseteq \epsilon[\![A]\!]; \epsilon[\![\Delta, u^E]\!]$.

Case (catch): We consider the following rule:

$$\frac{\Gamma \vdash P : A; \Delta}{\Gamma \vdash ?u^E.\ P : A \vee E; \Delta - u^E} \ (\mathsf{catch})$$

Suppose that $\epsilon \models \Gamma$. Then, by induction hypothesis, we have$\epsilon[\![P]\!] \subseteq \epsilon[\![A]\!]; \epsilon[\![\Delta]\!]$. Suppose that $r \in \epsilon[\![P]\!]$. We have two cases to consider.

- *Case: u does not occur in r.* In this case, we have $r \in \epsilon[\![A]\!]; \epsilon[\![\Delta - u^E]\!]$ by Lemma 10. So, we have $?(u,r) = \{(0,r)\}$ where $(0,r) \in \epsilon[\![A \vee E]\!]; \epsilon[\![\Delta - u^E]\!]$.
- *Case: u occurs in r.* By Lemma 11, we have $r/u \subseteq \epsilon[\![E]\!]; \epsilon[\![\Delta - u^E]\!]$. Hence, we have $(1,p) \in \epsilon[\![A \vee E]\!]; \epsilon[\![\Delta - u^E]\!]$ for any $p \in r/u$.

So, the theorem holds in this case.

Case ($\vee E$): The ($\vee E$) rule in $\mathbf{PA}_{c/t}$ takes the following form.

$$\frac{\Gamma \vdash P : A \vee B \quad \Pi \vdash Q : C; \Sigma \quad \Phi \vdash R : C; \Psi}{\Gamma, \Pi - x^A, \Phi - y^B \vdash \mathsf{case}(P, x^A.Q, y^B.R) : C; \Sigma, \Psi} \ (\vee E)$$

Suppose that $\epsilon \models \Gamma, \Pi - x^A, \Phi - y^B$. Then $\epsilon \models \Gamma$. So, by induction hypothesis, we have $\epsilon[\![P]\!] \subseteq \epsilon[\![A \vee B]\!]$. Take any $r \in \epsilon[\![P]\!]$. Then we have either $r = (0,p)$ and $p \in \epsilon[\![A]\!]$ or $r = (1,p)$ and $p \in \epsilon[\![B]\!]$. In the first case, we have $\epsilon[x = p] \models \Pi$ and by induction hypothesis we have

$$\epsilon[x = p][\![Q]\!] \subseteq \epsilon[x = p][\![C]\!]; \epsilon[x = p][\![\Sigma]\!] = \epsilon[\![C]\!]; \epsilon[\![\Sigma]\!] \subseteq \epsilon[\![C]\!]; \epsilon[\![\Sigma, \Psi]\!].$$

We can handle the second case similarly. Hence we have the theorem in this case.

Case ($\supset I$): The rule in question is:

$$\frac{\Gamma \vdash P : B; \Delta \quad \Pi \vdash A : \mathsf{Prop}}{\Gamma - x^A, \Pi \vdash \lambda x^A.P : A \supset B; \Delta} \ (\supset I)$$

Suppose that $\epsilon \models \Gamma - x^A, \Pi$. Take any $r \in \epsilon[\![\lambda x^A.P]\!]$. There are two possibilities.

- *Case r is a function such that* $\mathrm{dom}(r) = \epsilon[\![A]\!]$ *and* $r(p) \in \epsilon[x = p][\![P]\!]$ *for all* $p \in \epsilon[\![A]\!]$. Let $p \in \epsilon[\![A]\!]$. Then, we have $r(p) \in \epsilon[x = p][\![P]\!]$ and $\epsilon[x = p] \models \Gamma$. Hence, by induction hypothesis, we have

$$r(p) \in \epsilon[x = p][\![B]\!]; \epsilon[x = p][\![\Delta]\!] = \epsilon[\![B]\!]; \epsilon[\![\Delta]\!].$$

  Therefore, we have

$$r \in \epsilon[\![A]\!] \to (\epsilon[\![B]\!]; \epsilon[\![\Delta]\!]) = (\epsilon[\![A]\!] \to \epsilon[\![B]\!]); \epsilon[\![\Delta]\!] = \epsilon[\![A \supset B]\!]; \epsilon[\![\Delta]\!].$$

- *Case:* $r \in \bigcup_{p \in \epsilon[\![A]\!]} \epsilon[x = p][\![P]\!]_{\mathsf{e}}$. In this case, we have $r \in \epsilon[x = p][\![P]\!]_{\mathsf{e}}$ for some $p \in \epsilon[\![A]\!]$. Hence, by induction hypothesis, we have

$$r \in \epsilon[x = p][\![B]\!]; \epsilon[x = p][\![\Delta]\!] = \epsilon[\![B]\!]; \epsilon[\![\Delta]\!].$$

  But, since r is exceptional, we have $r \in |\epsilon[\![\Delta]\!]| \subseteq \epsilon[\![A \supset B]\!]; \epsilon[\![\Delta]\!]$.

  Case ($\supset E$): The rule in question is:

$$\frac{\Gamma \vdash P : A \supset B; \Delta \quad \Pi \vdash Q : A}{\Gamma, \Pi \vdash \mathsf{apply}(P, Q) : B; \Delta} \ (\supset E)$$

Suppose that $\epsilon \models \Gamma, \Pi$ and take any $r \in \epsilon[\![\mathsf{apply}(P, Q)]\!]$. Then we have two cases.

- *Case:* $r = p(q)$ *for some* p, q *such that* $p \in \epsilon[\![P]\!]_{\mathsf{p}}$ *and* $q \in \epsilon[\![Q]\!]$. Then, by induction hypothesis, we have $p \in \epsilon[\![A]\!] \to (\epsilon[\![B]\!]; \epsilon[\![\Delta]\!])$ and $q \in \epsilon[\![A]\!]$, so that

$$r = p(q) \in \epsilon[\![B]\!]; \epsilon[\![\Delta]\!].$$

- *Case:* $r \in \epsilon[\![P]\!]_{\mathsf{e}}$. In this case, we have $r \in |\epsilon[\![\Delta]\!]| \subseteq \epsilon[\![B]\!]; \epsilon[\![\Delta]\!]$.

□

We give a simple example. Let ϵ be the empty environment and suppose that $\vdash A : \mathsf{Prop}$, that is, A is a closed formula. Then, as we saw before, the judgment

$$\vdash ?u^A.\ \lambda x^A.!_\perp u^A(x^A) : \neg A \vee A$$

is derivable in $\mathbf{PA}_{c/t}$. By the definition of our BHK interpretation, it follows that:

$$\epsilon[\![?u.\ \lambda x^A.!_\perp u(x^A)]\!] = \begin{cases} \{(0, \emptyset)\} & \text{if } \epsilon[\![A]\!] = \emptyset, \\ \{(1, p) \mid p \in \epsilon[\![A]\!]\} & \text{if } \epsilon[\![A]\!] \neq \emptyset. \end{cases}$$

So, if the proposition (denoted by) A is false, then the derivation denotes the left injection of the unique proof of (the denotation of) $\neg A$, that is the left injection of the empty function, and if A is true, then the derivation denotes the right injection of the set of all proofs of A.

Let us consider the reduction of $\mathbf{PA}_{c/t}$ derivations. We have the following reduction rules for $\mathbf{PA}_{c/t}$ in addition to those for **HA**. Via the Curry-Howard isomorphism, derivations in $\mathbf{PA}_{c/t}$ can be seen as functional programs and these reduction rules can be thought of as giving operational semantics to these functional programs.

1. If $u \notin \mathrm{FV}(P)$, then $?u.\ P \mapsto \mathsf{inl}(P)$.
2. If $u \notin \mathrm{FV}(P)$, then $?u.\ !u(P) \mapsto \mathsf{inr}(P)$.
3. $!v(!u(P)) \mapsto !u(P)$.
4. $\mathsf{abort}(!u(P)) \mapsto !u(P)$.
5. $\mathsf{pair}(!u(P), Q) \mapsto !u(P)$.
6. $\mathsf{pair}(P, !u(Q)) \mapsto !u(Q)$.
7. $\mathsf{pair}(a, !u(P)) \mapsto !u(Q)$.
8. $\mathsf{car}(!u(P)) \mapsto !u(P)$.
9. $\mathsf{cdr}(!u(P)) \mapsto !u(P)$.
10. $\mathsf{inl}(!u(P)) \mapsto !u(P)$.
11. $\mathsf{inr}(!u(P)) \mapsto !u(P)$.
12. $\mathsf{apply}(!u(P), Q) \mapsto !u(P)$.
13. $\mathsf{apply}(!u(P), a) \mapsto !u(P)$.

Then these reduction rules enjoy the following subject reduction property.

Theorem 13. *If P has type A under $\Gamma;\Delta$ and $P \mapsto Q$, then Q has type A under some $\Pi;\Sigma$ smaller than $\Gamma;\Delta$.*

Moreover, the BHK interpretation respects these reduction rules in the following way.

Theorem 14. *If P has type A under $\Gamma;\Delta$, $\epsilon \models \Gamma$ and $P \mapsto Q$, then $\epsilon[\![P]\!] \supseteq \epsilon[\![Q]\!]$.*

5 Concluding remarks

In this paper, we defined propositions and proofs as abstract mathematical entities and showed that formulas and derivations, which are introduced as formal counterparts to propositions and proofs, can be interpreted by the classical version of BHK interpretation. As a concrete example of classical BHK interpretation, we gave an interpreration for classical first order arithmetic **PA**. This interpretation is almost constructive, since the interpretation of a derivation is the same as the interpretation for constructive arithmetic **HA** if the derivation is obtained without using the law of the excluded middle. We showed that the interpretation is sound and the denotation of derivation is preserved under reductions of derivations.

We also gave an interpretation for the system $\mathbf{PA}_{c/t}$ of the catch/throw calculus which is obtained from **HA** by adding the rules for the catch and throw operations. In $\mathbf{PA}_{c/t}$, the law of the excluded middle is derivable by using the cath/throw rules. If we look at such a derivation as a program, then we cannot compute (reduce) it in a constructive way. However, Murthy [4] formulated **PA** (as a programming language), roughly, as **HA** + the control operator $\mathcal{C}$ which serves as a witness of the law of the double negation elimination, and showed that for any Π^0_2 sentence A, if $P : A$, then P can be computed constructively. We take this as a hint that we might be able to extract constructive contents

from classical derivations derived in $\mathbf{PA}_{c/t}$, since the law of the double negation elimination is certainly derivable in $\mathbf{PA}_{c/t}$.

In order to pursue the possibility of extracting constructive contents from the catch/throw calculus, we do a case study using an example given in [4]. We will work in $\mathbf{PA}^{+}_{c/t}$ informally. We consider the sentence

$$A \stackrel{\Delta}{=} \exists n.\ prime(n) \wedge n < 100$$

Let P_2 be a derivation of $prime(2) \wedge 2 < 100$ and P_3 be a derivation of $prime(3) \wedge 3 < 100$, so that we have $\vdash \mathsf{pair}_A(2, P_2) : A$ and $\vdash \mathsf{pair}_A(3, P_3) : A$. We write $\vdash \mathsf{P}_2 : A$ and $\vdash \mathsf{P}_3 : A$ for these derivations. Then, we can derive the following judgment[1], where the first abort is $\mathsf{abort}_{prime(102)}$ and the second abort is $\mathsf{abort}_{102<100}$.

$$\vdash \lambda x^{\neg A}.\ x(\mathsf{pair}_A(102, \mathsf{pair}(\mathsf{abort}(x(\mathsf{P}_2)), \mathsf{abort}(x(\mathsf{P}_3))))) : \neg\neg A$$

We can also derive the following judgment.

$$\vdash \lambda y^{\neg\neg A}.\ ?u^{A}.\ y(\lambda x^{A}.\ !_{\perp} u(x)) : \neg\neg A \supset \perp \vee A$$

We write $\vdash Y : \neg\neg A$ and $\vdash C : \neg\neg A \supset \perp \vee A$ for the above two judgments[2] so that we have $\vdash C(Y) : \perp \vee A$. We can compute $C(Y)$ as follows, where we put $X \stackrel{\Delta}{=} \lambda x^{A}.\ !_{\perp} u(x)$.

$$\begin{aligned}
C(Y) &\to ?u^{A}.\ Y(\lambda x^{A}.\ !_{\perp} u(x)) \\
&\to ?u^{A}.\ X(\mathsf{pair}_A(102, \mathsf{pair}(\mathsf{abort}(X(\mathsf{P}_2)), \mathsf{abort}(X(\mathsf{P}_3))))) \\
&\to ?u^{A}.\ X(\mathsf{pair}_A(102, \mathsf{pair}(\mathsf{abort}(!u(\mathsf{P}_2)), \mathsf{abort}(!u(\mathsf{P}_3))))) \\
&\to ?u^{A}.\ X(\mathsf{pair}_A(102, \mathsf{pair}(!u(\mathsf{P}_2), \mathsf{abort}(!u(\mathsf{P}_3))))) \\
&\to ?u^{A}.\ X(\mathsf{pair}_A(102, !u(\mathsf{P}_2))) \\
&\to ?u^{A}.\ X(!u(\mathsf{P}_2)) \\
&\to ?u^{A}.\ !u(\mathsf{P}_2) \\
&\to \mathsf{inr}(\mathsf{P}_2)
\end{aligned}$$

We could thus read off the constructive content of the classical derivation $C(Y)$ of $\perp \vee A$. We can also get $C(Y) \to \mathsf{inr}(\mathsf{P}_3)$ similarly.

By the above observation, we saw that it is possible to extract constructive content from some classical derivations in $\mathbf{PA}^{+}_{c/t}$. However, at present, we do not know whether we can extract constructive content from any derivation of Π^0_2 sentence in $\mathbf{PA}^{+}_{c/t}$. Also, we do not know if we can give a sound BHK interpretation for $\mathbf{PA}^{+}_{c/t}$. Note that in $\mathbf{PA}_{c/t}$, the argument of any functional application

[1] We write $a(b)$ for $\mathsf{apply}(a, b)$ for simplicity.

[2] We note that C is not a valid derivation in $\mathbf{PA}_{c/t}$, since in $\mathbf{PA}_{c/t}$ we cannot apply a function to an argument having a free tag variable.

must be tag variable free, and the soundness of the BHK interpretation for $\mathbf{PA}_{c/t}$ comes from this restriction.

A number of classical logical systems have been introduced recently and most of them are explicitly designed to extract constructive contents of classical derivations ([1, 2, 4, 7, 8, 9, 10]). We hope that the classical BHK interpretation is useful for the analysis of these systems.

Acknowledgments

We thank Yukiyoshi Kameyama and Izumi Takeuti for discussions on BHK interpretaion and Susumu Hayashi for drawing our attention to Troelstra and van Dalen [12]. This work was supported in part by the Grant-in-Aid for the Science Research of Ministry of Education, Science and Culture (No. 08558023).

References

1. Barbanera, F. and Berardi, S., A symmetric lambda calculus for "classical" program extraction, pp. 495-515, in *Theoretical Aspects of Computer Software*, Lecture Note in Computer Science **789**, Hagiya, M. and Mitchell, J.C. (eds.), Springer-Verlag, 1994.
2. Griffin, T., A formulae-as-types notion of control, pp. 47-58, in *Proc. 17th ACM Symp. on Principle of Programming Languages*, ACM Press, 1990.
3. Kameyama, Y., A New Formulation of the Catch/Throw Mechanism, pp. 106-122, in *Second Fuji International Workshop on Functional and Logic Programming*, Ida, T., Ohori, A. and Takeichi, M. (eds.), World Scientific, 1997.
4. Murthy, C., An evaluation semantics for classical proofs, pp. 96-107, *Proc. 5th IEEE Symp. Logic in Computer Science*, IEEE Computer Society Press, 1991.
5. Nakano, H., A constructive formalization of the catch and throw mechanism, pp. 82-89, in *Proc. 7th Ann. IEEE Symp. on Logic in Computer Science*, 1992.
6. Nakano, H., The non-deterministic catch and throw mechanism and its subject reduction property, pp. 61-72, in *Logic, Language and Computation*, Lecture Notes in Computer Science **792**, Jones, N. D., Hagiya, M. and Sato, M. (eds.), Springer-Verlag, 1994.
7. Ong, C.-H.L., A semantic view of classical proofs, pp. 230-241, in *Proc. 11th IEEE Ann. Symp. of Logic in Computer Science*, Computer Science Press, 1996.
8. Ong, C.-H.L. and Stewart, C.A., A Curry-Howard foundation for functional computation with control, pp. 215-227, in *Proc. 24th Ann. ACM Symp. on Principles of Programming Languages*, ACM Press, 1997.
9. Parigot, M., An algorithmic interpretation of classical natural deduction, pp. 190-210, in *Proc. International Conf. on Logic Programming and Automated Reasoning*, Lecture Notes in Artificial Intelligence **624**, Springer-Verlag, 1992.
10. Rehof, N.J. and Sorensen, pp. 516-542, in *Theoretical Aspects of Computer Software*, Lecture Note in Computer Science **789**, Hagiya, M. and Mitchell, J.C. (eds.), Springer-Verlag, 1994.
11. Sato, M., Intuitionistic and classical natural deduction systems with the catch and the throw rules, pp. 75-92, in *Theoretical Computer Science* **175**, 1997.
12. Troelstra, A.S. and van Dalen, D., *Constructivism in Mathematics, An Introduction*, Studies in Logic and the Foundation of Mathematics **121**, North-Holland, 1988.