Games and Weak-Head Reduction for Classical PCF

Hugo Herbelin *

LITP, University Paris 7, 2 place Jussieu, 75252 Paris Cedex 05, France
INRIA-Rocquencourt, B.P. 105, 78153 Le Chesnay Cedex, France
Hugo.Herbelin@inria.fr

Abstract. We present a game model for classical PCF, a finite version of PCF extended by a `catch/throw` mechanism. This model is build from E-dialogues, a kind of two-players game defined by Lorenzen. In the E-dialogues for classical PCF, the strategies of the first player are isomorphic to the Böhm trees of the language.
We define an interaction in E-dialogues and show that it models the weak-head reduction in classical PCF. The interaction is a variant of Coquand's debate and the weak-head reduction is a variant of the reduction in Krivine's Abstract Machine.
We then extend E-dialogues to a kind of games similar to Hyland-Ong's games. Interaction in these games also models weak-head reduction. In the intuitionistic case (i.e. without the `catch/throw` mechanism), the extended E-dialogues are Hyland-Ong's games where the innocence condition on strategies is now a rule.
Our model for classical PCF is different from Ong's model of Parigot's lambda-mu-calculus. His model works by adding new moves to the intuitionistic case while ours works by relaxing the game rules.

Introduction

We investigate the links between Lorenzen's and Coquand's game-theoretic approach of provability, Hyland-Ong's game-theoretic approach of λ-calculus, and weak-head reduction as implemented by Krivine's Abstract Machine.

To exemplify these links, we choose as framework the finite Böhm trees of a variant of PCF extended by a catch/throw mechanism. This variant of PCF is classical in the sense that its typing system includes implicational classical logic.

We refer to Felscher [8] for the works of Lorenzen and his school. According to Felscher, the goal of Lorenzen was to give a game-based foundation of intuitionistic logic. Several kinds of two-players games parametrized by formulas were defined. Of these games, we note one in particular, called E-dialogues in Felscher [8]. As noticed by Lorenzen & Schwemmer [16], the strategies for the first player in E-dialogues have a structure of proofs in a certain cut-free sequent

* This research was partly supported by ESPRIT Basic Research Action "Types for Proofs and Programs" and by Programme de Recherche Coordonnées "Mécanisation du raisonnement".

calculus. The propositional fragment of this calculus is described in [10]. It is a variant of Gentzen's calculus LJ.

More generally, we call E-dialogue the kind of game which game-theoretically expresses the terms or proofs typed by a system having the subformula property. In particular, Coquand's games [3] (inspired from Gentzen [2]) for infinitary propositional logic are E-dialogues. Similarly, the typing system of Böhm trees for PCF has the subformula property and we can define E-dialogues for PCF.

We focus in section 1 on Böhm trees for classical PCF. Before defining E-dialogues for classical PCF, we define in section 2 a generic notion of two-players games parametrized by types. It's only by restricting the rules of the generic games that we get E-dialogues (see section 3). An isomorphism between Böhm trees for classical PCF and strategies for the first player in E-dialogues can now be stated. However, the rules of the E-dialogues are not the same for both players. So we define in section 4 spread E-dialogues where the players have dual roles. In spread E-dialogues both the strategies of the first player and of the second player are in one-to-one correspondence with Böhm trees.

In section 5, we recover E-dialogues and spread E-dialogues for intuitionistic PCF by another extra restriction, called *last asked first answered* condition. Intuitionistic spread E-dialogues are Hyland-Ong's dialogues where the innocence condition on strategies is now a rule of the game. The observation that the *last asked first answered* condition distinguishes between classical and intuitionistic type systems comes from Lorenz [13].

It is possible to define an interaction between strategies in an E-dialogue (in a way similar to Coquand's interaction [3]), but also in a spread E-dialogue (in this case it is just a "ping-pong"-like process). On the other side, we can evaluate the application of a Böhm tree to another by using a variant of Krivine's Abstract Machine. This is the weak-head reduction. We show that both interactions model the weak-head reduction.

The correspondence between weak-head reduction and interaction in Hyland-Ong's style of games is also proven in Danos *et al* [6]. The framework was the simply-typed pure λ-calculus and the starting point was the analogy between the justification pointers in Hyland-Ong's games and the pointers introduced in Danos & Regnier [7] to implement Krivine's Abstract Machine.

A game-theoretic model of simply-typed pure $\lambda\mu$-calculus (see Parigot [18]) has been given by Ong [17]. It derives from the intuitionistic game by adding new moves. In contrast, our model for classical PCF results from a liberalization of the rules of the intuitionistic game.

Interaction between strategies can be formalized through abstract machines too. The relations between these machines and Krivine's Abstract Machine are shown in [5].

1 Classical Simply-Typed Böhm Trees

We consider a language of Böhm trees for a simply-typed λ-calculus including constants, a case operator and a `catch/throw` mechanism (with static binding).

We adopt for the `catch` and `throw` operators a syntax (and later a behaviour) reminiscent of Parigot's $\lambda\mu$-calculus [18]. The operator $\texttt{catch}_\alpha$ is written $\mu\alpha$ and the operator $\texttt{throw}_\alpha$ is written $[\alpha]$. Classical Böhm trees are defined by the following grammar:

$$\begin{aligned} t &::= \texttt{case}\ x(u, ..., u)\ \texttt{of}\ (c \rightarrow t, ..., c \rightarrow t) \mid [\alpha]c \\ u &::= \lambda x, ..., x.\mu\alpha.t \end{aligned}$$

The letters x and α range over two distinct domains of names and c over the constants in the base types. The x's are called **λ-variables**. They are supposed to be distinct in the expression $\lambda x, ..., x.\mu\alpha.$. The α's are called **μ-variables** (they roughly correspond to entry points of (local) functions). In the `case` construct, the c's are supposed to be distinct and the t's are called **continuations**. We will often use a vector notation as in $\lambda\vec{x}.\mu\alpha.t$ or $\texttt{case}\ x(\vec{u})\ \texttt{of}\ (\vec{c} \rightarrow \vec{t})$.

The objects defined by the entries t and u are respectively called **classical evaluable Böhm trees** and **classical functional Böhm trees**.

Our Böhm trees are simply-typed. Types are built on a family $\mathcal{V}_C$ of base types, each one being inhabited by a finite number of elements. The following grammar defines types

$$A ::= A, ..., A \rightarrow C$$

where the sequence $A, ..., A$ may be empty and where C ranges over $\mathcal{V}_C$.

In a type $A = B_1, ..., B_p \rightarrow C$ the base type C is called **conclusion** of A and each B_i is called **premise** of A.

The typing system has two kinds of sequents. The sequents $(\Gamma \vdash \Delta)$ type evaluable Böhm trees and the sequents $(\Gamma \vdash \Delta; A)$ type functional Böhm trees. The Γ's are sequences of types annotated by λ-variables. The Δ's are sequences of base types annotated by μ-variables. The typing rules are:

$$\frac{t : (\Gamma, A_1^{x_1}, ..., A_n^{x_n} \vdash \Delta, C^\alpha)}{\lambda x_1, ..., x_n.\mu\alpha.t : (\Gamma \vdash \Delta; A_1, ..., A_n \rightarrow C)}\ Abs \qquad \frac{c \text{ is in } C}{[\alpha]c : (\Gamma \vdash \Delta, C^\alpha)}\ Cst$$

$$\frac{u_1 : (\Gamma \vdash \Delta; A_1) \quad \dots \quad u_n : (\Gamma \vdash \Delta; A_n) \qquad t_1 : (\Gamma \vdash \Delta) \quad \dots \quad t_p : (\Gamma \vdash \Delta)}{\texttt{case}\ x(u_1, ..., u_n)\ \texttt{of}\ (c_1 \rightarrow t_1, ..., c_p \rightarrow t_p) : (\Gamma \vdash \Delta)}\ App$$

with $(A_1, ..., A_n \rightarrow C)^x$ in Γ and $C = \{c_1, ..., c_p\}$.

If $u : (\vdash; A)$, we say that u is a **closed Böhm tree of type** A.

Remarks: 1) The extension to infinite set of constants and/or to non well-founded Böhm trees poses no difficulties. The extension to Böhm trees with undefined nodes is also direct.

2) We justify our langage of Böhm trees as follows. Let

$$
\begin{aligned}
t ::= {} & x \mid (t\ t) \mid \lambda x.t \\
\mid {} & c \mid \texttt{case}\ t\ \texttt{of}\ (c \to t, ..., c \to t) \\
\mid {} & \texttt{catch}_\alpha t \mid \texttt{throw}_\alpha t
\end{aligned}
$$

be a grammar for a finite version of PCF with a `catch/throw` mechanism. In fact, we intend $\texttt{catch}_\alpha t$ to represent the construction $\mu\alpha[\alpha]t$ of $\lambda\mu$-calculus and $\texttt{throw}_\alpha t$ the construction $\mu\delta[\alpha]t$ with δ not in t. This is justified by the following derived rule of $\lambda\mu$-calculus (δ and α not in t)

$$E(\mu\alpha[\alpha]E'(\mu\delta[\alpha]t)) \to E(t)$$

where E and E' are applicative contexts and $\mu\delta[\alpha]t$ is in evaluation position in E'.

Consider the theory of λ-calculus with constants and `case` operators

$$
\begin{aligned}
(\lambda x.t\ u) &= t[x:=u] \\
t &= \lambda x.(t\ x) && x \text{ fresh for } t \\
t &= \texttt{case}\ t\ \texttt{of}\ (\vec{c} \to \vec{c}) \\
\texttt{case}\ c_i\ \texttt{of}\ (\vec{c} \to \vec{t}) &= t_i \\
(\texttt{case}\ t\ \texttt{of}\ (\vec{c} \to \vec{u})\ v) &= \texttt{case}\ t\ \texttt{of}\ (\vec{c} \to \overrightarrow{(u\ v)}) \\
\texttt{case}\ (\texttt{case}\ t\ \texttt{of}\ (\vec{c} \to \vec{t}))\ \texttt{of}\ (\vec{c'} \to \vec{t'}) &= \texttt{case}\ t\ \texttt{of}\ (\vec{c} \to \overrightarrow{\texttt{case}\ t\ \texttt{of}\ (\vec{c'} \to \vec{t'})})
\end{aligned}
$$

enriched by equations coming from the theory of $\lambda\mu$-calculus

$$
\begin{aligned}
t &= \texttt{catch}_\alpha t && \alpha \text{ fresh for } t \\
(\texttt{catch}_\alpha t\ u) &= \texttt{catch}_\alpha(t[\texttt{throw}_\alpha v := \texttt{throw}_\alpha(v\ u)]\ u) \\
(\texttt{throw}_\alpha t\ u) &= \texttt{throw}_\alpha t \\
\texttt{case}\ (\texttt{catch}_\alpha t)\ \texttt{of}\ (\vec{c} \to \vec{u}) &= \texttt{catch}_\alpha(\texttt{case}\ t'\ \texttt{of}\ (\vec{c} \to \vec{u})) \\
& \quad \text{where } t' = t[\texttt{throw}_\alpha v := \texttt{throw}_\alpha(\texttt{case}\ v\ \texttt{of}\ (\vec{c} \to \vec{u}))] \\
\texttt{case}\ (\texttt{throw}_\alpha t)\ \texttt{of}\ (\vec{c} \to \vec{u}) &= \texttt{throw}_\alpha t \\
\texttt{throw}_\alpha(\texttt{case}\ t\ \texttt{of}\ (\vec{c} \to \vec{t})) &= \texttt{case}\ t\ \texttt{of}\ (\vec{c} \to \overrightarrow{\texttt{throw}_\alpha t}) \\
\texttt{throw}_\alpha(\texttt{throw}_\beta t) &= \texttt{throw}_\alpha t \\
\texttt{throw}_\alpha(\texttt{catch}_\beta t) &= \texttt{throw}_\alpha(t[\beta := \alpha]) \\
\texttt{catch}_\alpha(\texttt{catch}_\beta t) &= \texttt{catch}_\alpha(t[\beta := \alpha]) \\
\texttt{catch}_\alpha c &= \texttt{catch}_\alpha \texttt{throw}_\alpha c
\end{aligned}
$$

where the various substitutions are defined as in Parigot [18] (replacing $\texttt{throw}_\alpha$ by $[\alpha]$).

Orient the rules from left to right to get a rewriting system. We can show that any typed term reduced to a typed Böhm tree. Up to the 2nd, 3rd and 7th rules, typed Böhm trees are normal. Thus, assuming the confluence of the rewriting system, the typed Böhm trees describe the equivalence classes of typed terms. This justifies the terminology.

3) According to the theory of $\lambda\mu$-calculus in Parigot [18], any term of the form $\mu\alpha[\beta]t$ is equivalent to $\mu\alpha[\alpha]\mu\delta[\beta]t$ with δ fresh for t. This justifies to abbreviate our $\texttt{catch}_\alpha t$ (i.e. $\mu\alpha[\alpha]t$) by $\mu\alpha.t$ and our $\texttt{throw}_\alpha t$ (i.e. $\mu\delta[\alpha]$t with δ not in t) by $[\alpha]t$.

A precise study of the relations between $\lambda\mu$-calculus and λ-calculus extended by `catch` and `throw` can be found in Crolard [4].

Numeric Böhm trees are a special case of Böhm trees. In numeric Böhm trees, the λ-variables range over pairs of natural numbers and μ-variables on natural numbers. We will use numeric Böhm trees in section 3.1 to prove the correspondence with strategies. The numbers are imposed by the following annotated typing system:

$$\frac{t:(\Gamma, A_1^{\binom{p}{1}}, ..., A_n^{\binom{p}{n}} \overset{p}{\vdash} \Delta, C^{\alpha})}{\lambda\binom{p}{1}, ..., \binom{p}{n}.\mu\alpha.t:(\Gamma \overset{p}{\vdash} \Delta; A_1, ..., A_n \to C)}\ Abs \qquad \frac{c \text{ is in } C}{[p]c:(\Gamma \overset{p}{\vdash} \Delta, C^p)}\ Cst$$

$$\frac{u_1:(\Gamma \overset{p+1}{\vdash} \Delta; A_1)\ \ \ldots\ \ u_n:(\Gamma \overset{p+1}{\vdash} \Delta; A_n)\qquad t_1:(\Gamma \overset{p+1}{\vdash} \Delta)\ \ \ldots\ \ t_p:(\Gamma \overset{p+1}{\vdash} \Delta)}{\texttt{case}\ \binom{q}{j}(u_1, ..., u_n)\ \texttt{of}\ (c_1 \to t_1, ..., c_p \to t_p):(\Gamma \overset{p}{\vdash} \Delta)}\ App$$

with $(A_1, ..., A_n \to C)^{\binom{q}{j}}$ in Γ and $C = \{c_1, ..., c_p\}$.

1.1 Weak-Head Reduction

We now define a computation on classical Böhm trees: the weak-head computation of a functional Böhm tree applied to arguments which are themselves Böhm trees. For this purpose, we define a variant of Krivine's Abstract Machine. The machine is described in a syntax reminiscent of $\lambda\sigma$-calculi (see Abadi *et al* [1]), in the style of Leroy [12] or Hardin *et al* [9].

$$\begin{aligned} s &::= t[e] \\ e &::= w; ...; w \\ w &::= (\vec{x} \leftarrow \vec{u}; [\alpha]\vec{c} \to \vec{t})[e] \end{aligned}$$

A **state** of the machine (entry s of the grammar) consists of an evaluable Böhm tree in an environment. An **environment** e is a sequence of windows. A **window** w contains bindings of two kinds. First the bindings of variables to arguments. Second the bindings of a μ-variable and of constants to continuations. Both terms and continuations of a window have meaning in an environment local to them.

There are two kinds of rules. The first rule applies when the computation needs to know the value of a variable x_{ij}. If x_{ij} is bound to $u_{ij} = \lambda\vec{y}.\mu\beta.t'$ in e then its arguments $\vec{u}$ are bound to the formal parameters $\vec{y}$ and the current continuation $\vec{t}$ (what to do when t' returns a constant) is bound to the entry point β of u_{ij}.

$$\texttt{case}\ x_{ij}(\vec{u})\ \texttt{of}\ (\vec{c} \to \vec{t})[e] \quad \overset{wh}{\to}_1 \quad t'[(\vec{y} \leftarrow \vec{u}; [\beta]\vec{c} \to \vec{t})[e]; e_i]$$

with $e = w_1; ...; w_r$ and $w_i = (\vec{x_i} \leftarrow \vec{u_i}; [\alpha_i]\vec{c_i} \to \vec{t_i})[e_i]$ and $u_{ij} = \lambda\vec{y}.\mu\beta.t'$.

The second rule applies when a constant is return to some entry point.

$$[\alpha_i]c_j\ [e] \quad \overset{wh}{\rightarrow}_1 \quad t_{ij}[e_i]$$

with $e = w_1; ...; w_r$ and $w_i = (\vec{x_i} \leftarrow \vec{u_i}; [\alpha_i]\vec{c_i} \rightarrow \vec{t_i})[e_i]$.

The weak-head reduction works by repeatedly applying the two rules. Let $u_0 = \lambda\vec{x_0}.\mu\alpha_0.t_0$ be a Böhm tree of type $\vec{A} \rightarrow C$ and $\vec{v_0}$ a family of Böhm trees of respective types $A_1, ..., A_n$. The **weak-head reduction of u_0 applied to $\vec{v_0}$** is the following sequence:

$$r_0 = t_0[(\vec{x_0} \leftarrow \vec{v_0}; [\alpha_0]\epsilon)[\,]\ \overset{wh}{\rightarrow}_1\ r_1\ \overset{wh}{\rightarrow}_1\ ...\ \overset{wh}{\rightarrow}_1\ r_n\ \overset{wh}{\rightarrow}_1\ ...$$

The symbol ϵ denotes the empty sequence of continuations bindings. When u_0 and $\vec{v_0}$ are closed (both for λ-variables and μ-variables), only the constants returned on α_0 are not bound in the environment. This property is preserved from step to step. Thus, if it stops, the sequence stops in a state of the form $[\alpha_0]c_i\ [e]$. Since no continuation is bound to $[\alpha_0]c_i$, no reduction rule applies.

Remark: Our machine arises as a stack-free form of Krivine's Abstract Machine. The typing ensures that the arity of arguments always matches the arity of formal parameters. This is what allows to avoid a stack. The way to handle case operators without stack comes from [5]. More generally, we refer to [5] for a comparison of our machine with Danos-Regnier Pointer Abstract Machine (see [7, 6]) and for an extension to pure λ-calculus. A short proof of correctness of the machine w.r.t. Coquand's debate (as done in section 3.2) also appears in [5].

2 Games

Games interpret types. If A is a type, we write G_A the game which interprets it. It is a game between two players called *Player* and *Opponent*. Moves consist in attacks of subtypes of A or in answers to attacks. Plays are alternating sequences of numbered Player's and Opponent's moves starting from an initial attack of A by Opponent.

Assume A is $B_1, ..., B_n \rightarrow C_0$. The **initial attack** of A is written $[_*$ and numbered 0. This attack means both a question on the conclusion C_0 of A and the assertion of the premises B_i of A. All subsequent **attacks** are written $[_i^p$. The exponent p is called **justification**. It is a reference to a previous attack of the other player, say the attack of $B'_1, ..., B'_n \rightarrow C'$. This attack was asserting B'_1, ...,B'_n and i is the index of one of the B'_1, ..., B'_n. Here again, the attack means both a question on the conclusion of B'_i and the assertion of each premise of B'_i.

An attack is waiting for an answer. An **answer** is written $]_c^p$. The exponent p is also a **justification**. It corresponds to the number of the attack to which it answers. This attack was questioning a base type, say C'' and c is a constant in C''.

Formally, the **game** G_A is a set of legal positions on A. A **legal position** on A is a finite or infinite sequence $d_0d_1d_2...$ of moves. For $n \geq 1$, it may be convenient to write d_n as $m_n^{p_n}$. The number p_n is the justification and m_n is either $[_i$ or $]_c$. To be a legal position, the sequence has to satisfy the following properties:

- Initial attack of Opponent
  We have $d_0 = [_*$.
- Moves are justified by previous moves of the other player
  For all $n > 0$, p_n is less than n and of distinct parity.
- Correctness of attacks
  For $n \geq 1$, if $d_n = [_i^p$, the move d_p is an attack of some type $B_1, ..., B_m \rightarrow C$ and $1 \leq i \leq m$. We say that d_n is an attack of B_i.
- Correctness of answers
  For $n \geq 1$, if $d_n =]_c^p$, the move d_p is an attack of some type $B_1, ..., B_m \rightarrow C$ and c is in C.

The moves d_n, with n odd, are called **Player's moves** or **P-moves** and the ones with n even are called **Opponent's moves** or **O-moves**.

2.1 Strategies

Unformally, a strategy for a player is a function mapping legal positions (at which the player is to move) to a move of this player. In a legal position, only the moves of the other player are useful to determine what to move. This leads to the following definition.

Assume A is $B_1, ..., B_{n_0} \rightarrow C$. A **P-strategy** (resp **O-strategy**) ϕ for G_A is a function which maps finite sequences of O-moves (resp P-moves) to P-moves (resp O-moves). The domain $\mathbf{Dom}(\phi)$ of ϕ is structured as a tree. It satisfies the following clauses:

- For a P-strategy: the sequence reduced to the single O-move $[_*$ is in $\mathbf{Dom}(\phi)$.
- For an O-strategy: the one-P-move sequences $]_c^0$ (with c in C^0) and $[_i^0$ (with $1 \leq i \leq n_0$) are in $\mathbf{Dom}(\phi)$.
- If the sequence $d_0 d_1 ... d_n$ is in $\mathbf{Dom}(\phi)$ then,
  1- $d_0 \phi(d_0) d_1 ... d_n \phi(d_0 ... d_n)$ forms a legal position,
  2- $d_0 d_1 ... d_{n+1}$ is in $\mathbf{Dom}(\phi)$ if and only if $d_0 \phi(d_0) d_1 ... d_n \phi(d_0 ... d_n) d_{n+1}$ forms a legal position.

P-strategies are strategies for Player while O-strategies are strategies for Opponent.

Alternatively, a P-strategy can be seen as a (possibly infinite) tree where branches are labelled by O-moves and nodes by P-moves. Moreover, it makes sense to restrict a strategy to what it determines after some point in a play. This leads to the definition of substrategy in tree form beyond a legal position.

A **P-substrategy in tree form beyond** π is inductively defined by:

Let d be a P-move such that πd is legal. If, for any O-move d' such that $\pi dd'$ is legal, $\phi_{d'}$ is a P-substrategy beyond $\pi dd'$, then the tree $(d, (\phi_{d'})_{d'})$ is a P-substrategy beyond π.

The move d labels the root node of the tree and the $\phi_{d'}$ are the branches. When no O-move is allowed after d, the family $(\phi_{d'})_{d'}$ is empty and the P-substrategy is restricted to a leaf. A **P-strategy in tree form** is a P-substrategy in tree form beyond the legal position restricted to the initial move $[_*$.

Proposition 1. *There is a one-to-one correspondence between strategies (as defined by the first definition) and strategies in tree-form. This correspondence preserves the tree structure of the domains of the strategies.*

The notion of P-substrategies in tree form beyond a legal position is a technical notion used to prove the correspondence with Böhm trees.

3 The Model of E-Dialogues

Coquand [3] interprets proofs of the "Calculus of Novikoff" as strategies in a two-players game. The calculus is a (cut-free) sequent calculus (as LJ and LK of Gentzen) for infinitary logic. The game is as in section 2 except for Opponent: the O-moves must be justified by the preceding P-move. Such a game is similar to Lorenzen's classical E-dialogues in Felscher [8].

Here, we define E-dialogues for classical Böhm trees.

3.1 E-Dialogues

The E-dialogue G^E_A interpreting the type A is defined by its set of legal E-positions. A **legal E-position** on A is a legal position in G_A which satisfies the following extra condition:

- O-moves are justified by the preceding P-move:
  For all even $n \neq 0$, we have $p_n = n - 1$.

The justifications of O-moves are trivial in E-dialogues. We do not write them in the sequel.

An **E-P-strategy for** A is a P-strategy for the E-dialogue G^E_A.

Proposition 2. *E-P-strategies for A and closed functional Böhm trees of type A are isomorphic.*

Proof. We show rather the correspondence between E-P-strategies in tree form and numeric Böhm trees. Let π be a legal E-position of odd length. Let Γ be the set of types asserted by Opponent (and thus attackable by Player) in π. Let Δ be the types of the questions asked by Opponent. We tag each type in Γ by a pair consisting of the move number when the type was asserted and of the premise index (as in the definition of numeric Böhm trees). Similarly, we tag each base type in Δ by the move number when the question was asked. We show that E-P-substrategies in tree form beyond π are isomorphic to evaluable Böhm trees typed by the sequent $(\Gamma \vdash \Delta)$.

For well-founded strategies and Böhm trees, the definition of the correspondence is by recursion on the tree structure:

- A P-attack $[_i^p$ corresponds to an occurrence of the *App* rule with head-variable $\binom{p}{i}$. Branches indexed by an O-move correspond to subderivations of the *App* rule.
  - An O-attack $[_i$ corresponds to the i^{th} left premise of the rule *App* (together with the subsequent *Abs* rule).
  - An O-answer $]_{c_i}$ corresponds to the i^{th} right premise of the rule *App*.
- A P-answer $]_c^p$ corresponds to an occurrence of the *Cst* rule with constant c and type C^p.

Finally, the initial move $[_*$ is in correspondence with the top *Abs* rule of functional Böhm trees. Then, it is direct to show that the correspondence is an isomorphism respecting the tree structure.

Hereafter, we write ϕ_u for the E-P-strategy associated to the Böhm tree u.

3.2 Coquand's Debate

Let ϕ be an E-P-strategy for $A_1, ..., A_n \rightarrow C$ and $\vec{\psi}$ be a family of E-P-strategies for $A_1, ..., A_n$ respectively. Since neither ϕ and $\vec{\psi}$ are forced to play moves justified by the preceding opponent's move, it is not possible to directly let them interact.

Coquand [3] proposed a way to let ϕ and $\vec{\psi}$ interact in such a way that the interaction computes a result for C. The idea of Coquand is as follows: at each step of the debate (which is a sequence of moves), there is a canonical way to extract a subsequence which is a legal E-position. From this legal E-position, the E-strategies can be applied. Following Hyland-Ong's terminology [11], we call *view* the extracted E-position. However, in contrast with [11], when Player (resp Opponent) is to move, we keep in the view only the moves of Opponent (resp Player). This is sufficient to apply the E-strategies.

Roughly, the view (typically for P) of a legal position is obtained by forgetting the moves which occur between an O-move and the P-move which justified it.

Formally, the **view** $\mathcal{V}(\pi)$ of a legal sequence $\pi = d_0 d_1 ... d_n$ is recursively defined as follows:

- If $d_n = m^p$ with $p \neq 0$ then $\mathcal{V}(d_0...d_n) = \mathcal{V}(d_0...d_{p-1})m$
- If $d_n = m^0$ then $\mathcal{V}(d_0...d_n) = m$
- $\mathcal{V}(d_0) = d_0$

If the last move of π is a P-move, the view is an **O-view**. Otherwise, it is a **P-view**.

Each move in the view of π is a move in π with the justification dropped. The **view renumbering sequence** v_π, defined as follows, tells the number that the moves of the view have in π.

- If $d_n = m^p$ with $p \neq 0$ then $v_{d_0 \dots d_n} = v_{d_0 \dots d_{p-1}} n$
- If $d_n = m^0$ then $v_{d_0 \dots d_n} = n$
- $v_{d_0} = 0$

If $v_\pi = v_0 v_1 \dots v_{n'}$ is the view renumbering sequence of π. If $d = m^p$ is a move with $p \leq n'$, we note $v_\pi(d)$ for the move m^{v_p}.

The **debate** $d_E(\phi, \vec{\psi})$ between ϕ and $\vec{\psi}$ forms a legal position. Both ϕ and the ψ_i in $\vec{\psi}$ are E-P-strategies but in the resulting play, who plays according to ϕ is Player and who plays according to the ψ_i is Opponent. The first move is an initial attack of Opponent questioning C. Then, both players play in turn:

$$\begin{aligned}
&d_0 = [_* \\
&d_{2n+1} = v_{d_0 \dots d_{2n}}(\phi(\mathcal{V}(d_0 \dots d_{2n}))) \\
&d_{2n+2} = v_{d_0 \dots d_{2n+1}}(\vec{\psi}(\mathcal{V}(d_0 \dots d_{2n+1}))) \\
&d_E(\phi, \psi) = d_0 d_1 \dots d_n \dots
\end{aligned}$$

where $\vec{\psi}(m_0 m_1 \dots m_q)$ is $\psi_i([_* m_1 \dots m_q)$ when m_0 is $[_i$.

Thus, the view transposes the current state of the debate into a subsequence of O-moves in an E-dialogue. From this, the player who is to move can apply its strategy. The view renumbering sequences serve to transpose back the justification in the whole position.

3.3 Debate Models Weak-Head Reduction

Let $u_0 = \lambda \vec{x_0}.\mu\alpha_0.t_0$ be a Böhm tree of type $A_1, \dots, A_n \to C$ and $\vec{v_0}$ a family of Böhm trees of respective types $A_1, \dots, A_n$. The debate $d_E(\phi_{u_0}, \vec{\phi_{v_0}})$ follows step by step the weak-head reduction of u_0 applied to $\vec{v_0}$.

To express this correspondence, we consider the transposition $d_{\mathrm{wh}}(u_0, \vec{v_0})$ of the weak-head reduction into a legal position. We annotate instances of the first reduction rule by attacks and instances of the second by answers. A superscript on windows is necessary too. At the end, we add a dummy window $()[\,]$ in the second reduction rule (this simplifies the numbering of windows in the next proof).

$$\texttt{case}\ x_{ij}(\vec{u})\ \texttt{of}\ (\vec{c} \to \vec{t})[e] \quad \stackrel{wh(n)=[_j^p}{\longrightarrow}_1 \quad t'[(\vec{y} \leftarrow \vec{u}; [\beta]\vec{c} \to \vec{t})^n[e]; e_i]$$

with $e = w_1; \dots; w_r$ and $w_i = (\vec{x_i} \leftarrow \vec{u_i}; [\alpha_i]\vec{c_i} \to \vec{t_i})^p[e_i]$ and $u_{ij} = \lambda\vec{y}.\mu\beta.t'$.

$$[\alpha_i]c_j[e] \quad \stackrel{wh(n)=]_{c_i}^p}{\longrightarrow}_1 \quad t_{ij}[()[\,]; e_i]$$

with $e = w_1; \dots; w_r$ and $w_i = (\vec{x_i} \leftarrow \vec{u_i}; [\alpha_i]\vec{c_i} \to \vec{t_i})^p[e_i]$.

Starting from $r_0 = t_0[(\vec{x_0} \leftarrow \vec{v_0}; [\alpha_0]\epsilon)^0[\,]]$, we get the sequence

$$r_0 \overset{wh(1)=d_1}{\longrightarrow}_1 r_1 \overset{wh(2)=d_2}{\longrightarrow}_1 \dots \overset{wh(n)=d_n}{\longrightarrow}_1 r_n \overset{wh(n+1)=d_{n+1}}{\longrightarrow}_1 \dots$$

We write $d_{\mathrm{wh}}(u_0, v_0)$ for the sequence $[_*, d_1, ..., d_n, ...$

We now state the correspondence.

Theorem 3. *If u_0 is a closed Böhm tree of type $A_1, ..., A_n \to C$ and $\vec{v_0}$ a family of closed Böhm trees of respective types $A_1, ..., A_n$ then we have*

$$d_{\mathrm{wh}}(u_0, \vec{v_0}) = d_E(\phi_{u_0}, \vec{\phi_{v_0}})$$

We need first some definitions.

We define **occurrences** of evaluable Böhm trees and **hat occurrences** of functional Böhm trees in a functional Böhm tree u. Böhm subtrees and P-views are in correspondence. To enforce this link, occurrences are taken to be P-views, i.e. sequences of O-moves with the justification dropped.

- $\lambda\vec{x}.\mu\alpha.t$ has hat occurrence $\widehat{[_*}$ in itself
- t has occurrence $[_*$ in $\lambda\vec{x}.\mu\alpha.t$
- If $\texttt{case}\ x_p(\lambda\vec{y_1}.\mu\alpha_1.t_1, ..., \lambda\vec{y_n}.\mu\alpha_n.t_n)\ \texttt{of}\ (c_1 \to t'_1, ..., c_q \to t'_q)$ has occurrence π in u then $\lambda\vec{y_i}.\mu\alpha_i.t_i$ has hat occurrence $\widehat{\pi[_i}$ in u, t_i has occurrence $\pi[_i$ in u and t'_i has occurrence $\pi]_{c_i}$ in u.

If u is a Böhm tree and π an occurrence of t in u, we define $u_{|\pi}$ as t. If $\vec{v}$ is a family of Böhm trees and $\pi = [_* m_1 ... m_n$ is an occurrence of t in v_i, we let $\vec{v}_{|[_i m_1 ... m_n} = (v_i)_{|\pi}$. Similarly for hat occurrences.

We can now give the proof.

Proof. Let $u_0 = \lambda\vec{x_0}.\mu\alpha_0.t_0$. Let

$$r_0 \overset{wh(1)=d_1}{\longrightarrow}_1 r_1 \overset{wh(2)=d_2}{\longrightarrow}_1 \dots \overset{wh(n)=d_n}{\longrightarrow}_1 r_n \overset{wh(n+1)=d_{n+1}}{\longrightarrow}_1 \dots$$

be the weak-head reduction originating from $r_0 = t_0[(\vec{x_0} \leftarrow \vec{v_0}; [\alpha_0]\epsilon)^0[\,]]$. Let π be the debate $d_E(\phi_{u_0}, \vec{\phi_{v_0}})$. We note $\pi_{|n}$ for the restriction of the debate to the moves 0 to n.

We show by induction on n that $r_n = t_n[e_n]$ where

- if n is even, $t_n = (u_0)_{|\mathcal{V}(\pi_{|n})}$
  if n is odd, $t_n = (v_0)_{|\mathcal{V}(\pi_{|n})}$
- e_n is $w_{q_r}; ...; w_{q_1}$ where $r = |\mathcal{V}(\pi_{|n})|$ and $q_{i+1} = v_{\pi_{|n}}(i)$
- w_0 is $(\vec{x_0} \leftarrow \vec{v_0}; [\alpha_0]\epsilon)^0[\,]$
  if q is odd (resp q is even) and $(\vec{v_0})_{|\mathcal{V}(\pi_{|q})}$ (resp $(u_0)_{|\mathcal{V}(\pi_{|q})}$) is a constant c then $w_{q+1} = ()[\,]$ otherwise $w_{q+1} = (\vec{x_q} \leftarrow \vec{u_q}; [\alpha_q]\vec{c_q} \to \vec{t'_q})^q[e_q]$ where

- if q is even, $u_{qj} = (u_0)_{|\widehat{\mathcal{V}(\pi_{|q})[_j}}$ and $t'_{qj} = (u_0)_{|\mathcal{V}(\pi_{|q})]_{c_j}}$ $(q \neq 0)$
- if q is odd, $u_{qj} = (\vec{v_0})_{|\widehat{\mathcal{V}(\pi_{|q})[_j}}$ and $t'_{qj} = (\vec{v_0})_{|\mathcal{V}(\pi_{|q})]_{c_j}}$

Clearly, r_0 satisfies this property.

Now if $r_n = t_n[e_n]$, what happens for r_{n+1}? We suppose n odd (the case $n \neq 0$ even is similar). We have to consider the two possible reduction steps:

- $\left|\begin{array}{l} t_n = \texttt{case}\ x(\vec{u'})\ \texttt{of}\ (\vec{c''} \to \vec{t''}) \text{ with } x = x_{q_ij} \text{ in } e_n \text{ and } u_{q_ij} = \lambda\vec{x'}.\mu\alpha'.t' \\ r_{n+1} = t'[(\vec{x'} \leftarrow \vec{u'}; [\alpha']\vec{c''} \to \vec{t''})^{n+1}[e_n]; e_{q_i}] \end{array}\right.$
  In this case, we have $wh(n+1) = [^{q_i}_j$ and $\mathcal{V}(\pi_{|n+1}) = \mathcal{V}(\pi_{|q_i})\ [_j$.
  We show first that $t' = t_{n+1}$. We have $u_{q_ij} = (\vec{v_0})_{|\widehat{\mathcal{V}(\pi_{|q})[_j}}$. By definition of the view, we get $u_{q_ij} = (\vec{v_0})_{|\widehat{\mathcal{V}(\pi_{|n+1})}}$ and $t' = (\vec{v_0})_{|\mathcal{V}(\pi_{|n+1})}$.
  Then, we show $[(\vec{x'} \leftarrow \vec{u'}; [\alpha']\vec{c''} \to \vec{t''})^{n+1}[e_n]; e_{q_i}] = [e_{n+1}] = [w_{q_r}; ...; w_{q_1}]$ with $r = |\mathcal{V}(\pi_{|n+1})|$ and $q_{i+1} = v_{\pi_{|n+1}}(i)$. By definition of the view, we actually have $r = |\mathcal{V}(\pi_{|q_i})| + 1 = |\mathcal{V}(\pi_{|n+1})|$. By definition of the view and of e_{q_i}, the r first windows actually are the windows w_{q_1} to $w_{q_{r-1}}$ with $q_{k+1} = v_{\pi_{|n+1}}(k)$. But also $v_{\pi_{|n+1}}(r) = n+1$. Then, u'_j $(= (u_0)_{|\widehat{\mathcal{V}(\pi_{|q})[_j}}$ by definition of hat occurrences) is actually $u_{(n+1)j}$ and t''_j $(= (u_0)_{|\mathcal{V}(\pi_{|q})]_{c_j}}$ by definition of occurrences) is actually $t'_{(n+1)j}$.
- $\left|\begin{array}{l} t_n = [\alpha]c_j \text{ with } \alpha = \alpha_{q_i} \text{ in } e_n \\ r_{n+1} = t'_{q_ij}[()[\,]; e_{q_i}] \end{array}\right.$
  Then we have $wh(n+1) =]^{q_i}_{c_j}$ and $\mathcal{V}(\pi_{|n+1}) = \mathcal{V}(\pi_{|q_i})\ ]_{c_j}$.
  We show first that $t'_{q_ij} = t_{n+1}$. We have $t'_{q_ij} = (\vec{v_0})_{|\mathcal{V}(\pi_{|q_i})]_{c_j}}$. By definition of the view, this means $t'_{q_ij} = (\vec{v_0})_{|\mathcal{V}(\pi_{|n+1})}$ as wanted.
  We show then that $[()[\,]; e_{q_i}] = [e_{n+1}] = [w_{q_r}; ...; w_{q_1}]$ with $r = |\mathcal{V}(\pi_{|n+1})|$ and $q_{i+1} = v_{\pi_{|n+1}}(i)$. As above, this comes by induction hypothesis for windows in e_{q_i}. Moreover, $t_n = (u_0)_{|\mathcal{V}(\pi_{|n})}$ is a constant and w_{n+1} is actually dummy.

This ends the proof

4 The Model of Spread E-Dialogues

The E-dialogues are highly asymmetrical between Player and Opponent. A liberalization of the rules leads to the spread E-dialogues. These dialogues can be seen as a variant for classical PCF of Hyland-Ong's dialogues (see section 5).

4.1 Spread E-Dialogues

We now allow Opponent to play moves justified by a P-move which is not the last move of Player. However, to keep a kind of dialogue which constitutes a model, we internalize the determinism w.r.t. the view in the rules of the game. This is the *same view same move* condition.

The **spread E-dialogue** G_A^S associated to A is defined by its set of legal spread E-positions. A **legal spread E-position** on A is a legal position $\pi = d_0 d_1 ... d_q$ in G_A which satisfies the following extra condition:

- *same view same move* condition

  For all $n, n' \leq q$, if $\mathcal{V}(\pi_{|n}) = \mathcal{V}(\pi_{|n'})$ then there is a move m^p such that both $d_{n+1} = v_{\pi_{|n}}(m^p)$ and $d_{n'+1} = v_{\pi_{|n'}}(m^p)$.

A **spread E-P-strategy for** A is a P-strategy for the spread E-dialogue G_A^S. Similarly for an E-O-strategy. A **spread E-strategy** is either a spread E-P-strategy or a spread E-O-strategy.

4.2 E-P-Strategies and Spread E-Strategies

Since the moves in spread E-strategies are determined by the views (which are sequences of O-moves in E-dialogues), we can expect a bijection between E-P-strategies and spread E-strategies (whatever the strategy is for Player or for Opponent).

In the rest of the section, we consider P-views of sequences of O-moves and O-views of sequences of P-moves. This makes sense since only the knowledge of the moves of other player are relevant in the definition of the view.

Let ϕ be an E-P-strategy for A. Let $\mathbf{Dom}(\phi^*)$ be the set of sequences π of O-moves such that $\mathcal{V}(\pi)$ is in $\mathbf{Dom}(\phi)$. Let ϕ^* be the extension of ϕ on $\mathbf{Dom}(\phi^*)$ defined by $\phi^*(\pi) = v_\pi(\phi(\mathcal{V}(\pi)))$.

Proposition 4. *If ϕ is an E-P-strategy for A then ϕ^* is a spread E-P-strategy for A.*

Similarly, let $\vec{\psi}$ be a family of E-P-strategies for $A_1, ..., A_n$ respectively. Let $\mathbf{Dom}(\vec{\phi}^+)$ be the set of sequences π of P-moves (replacing the first move $[_i^0$ by $[_*$) such that $\mathcal{V}(\pi)$ is in $\mathbf{Dom}(\phi_i)$. Let $\vec{\phi}^+$ be defined on $\mathbf{Dom}(\vec{\phi}^+)$ by $\phi^+(\pi) = v_\pi(\phi_i(\mathcal{V}(\pi')))$ when π begins with $[_i^0$ and π' is obtained from π by replacing $[_i^0$ by $[_*$.

Proposition 5. *If $\vec{\phi}$ is a family of E-P-strategies for $A_1, ..., A_n$ then $\vec{\phi}^+$ is a spread E-O-strategy for $A_1, ..., A_n \to C$.*

Conversely, a spread E-strategy can be restricted into a E-P-strategy by keeping in the domain only the sequences which come from a legal E-position. Let ϕ be a spread E-P-strategy for A. We define $\mathbf{Dom}(\phi)^-$ as the set of sequences $[_* m_1 ... m_q$ of O-moves such that $[_* m_1^1 ... m_q^{2q-1}$ is in $\mathbf{Dom}(\phi)$. Similarly, let ψ be a spread E-O-strategy for $A_1, ..., A_n \to C$. For each i, we define $\mathbf{Dom}(\phi)_i^-$ as the set of sequences $[_* m_1 ... m_q$ of O-moves such that $[_i^0 m_1^2 ... m_q^{2q}$ is a sequence of P-moves in $\mathbf{Dom}(\phi)$.

Proposition 6. *1. If ϕ is a spread E-P-strategy for A then the restriction of ϕ on $\mathbf{Dom}(\phi)^-$ is an E-P-strategy for A.*

2. *If ψ is a spread E-O-strategy for $A_1, ..., A_n \to C$ then the restriction of ψ on* $\mathbf{Dom}(\phi)_i^-$ *is an E-P-strategy for A_i*
3. *If ϕ is an E-P-strategy for A then the restriction of ϕ^* on* $\mathbf{Dom}(\phi^*)^-$ *is ϕ.*
4. *If $\vec{\psi}$ is a family of E-P-strategies for $A_1, ..., A_n$ respectively then the restriction of $\vec{\psi}^+$ on* $\mathbf{Dom}(\phi^*)_i^-$ *is ψ_i.*

Corollary 7. *The followings are in bijection:*

- *closed Böhm trees of type A*
- *E-P-strategies for A*
- *spread E-P-strategies for A.*

Also, the followings are in bijection:

- *families of closed Böhm trees of respective types $A_1, ..., A_n$*
- *families of E-P-strategies for $A_1, ..., A_n$*
- *families of spread E-P-strategies for $A_1, ..., A_n$*
- *spread E-O-strategies for $A_1, ..., A_n \to C$*

4.3 Interaction Between Spread E-Strategies

Let ϕ be a spread E-P-strategy for $A_1, ..., A_n \to C$ and $\vec{\psi}$ a family of spread E-P-strategies for $A_1, ..., A_n$. By corollary 7, $\vec{\psi}$ can be seen as a spread E-O-strategy Ψ for $A_1, ..., A_n \to C$. Therefore, it is direct to define an interaction between ϕ and $\vec{\psi}$ by letting ϕ and Ψ play the one against the other.

We define the debate $d_S(\phi, \vec{\psi})$ as follows:

$$\begin{aligned} d_0 &= [_* \\ d_{2n+1} &= \phi(d_0 d_2 ... d_{2n}) \\ d_{2n+2} &= \Psi(d_1 d_3 ... d_{2n+1}) \end{aligned}$$

The interaction in a spread E-dialogue follows step by step the debate in an E-dialogue.

Proposition 8. *If ϕ is an E-P-strategy for $A_1, ..., A_n \to C$ and $\vec{\psi}$ a family of E-P-strategies for $A_1, ..., A_n$ respectively, then we have*

$$d_E(\phi, \vec{\psi}) = d_S(\phi^*, \vec{\psi}^+)$$

Proof. Directly since $\phi^*(\pi) = v_\pi(\phi(\mathcal{V}(\pi)))$ and similarly for $\vec{\psi}$.

Corollary 9. *If u_0 is a Böhm tree of type $A_1, ..., A_n \to C$ and $\vec{v_0}$ is a family of Böhm trees of types $A_1, ..., A_n$ respectively then we have*

$$d_{\mathrm{wh}}(u_0, \vec{v_0}) = d_E(\phi_{u_0}, \vec{\phi_{v_0}}) = d_S(\phi^*_{u_0}, \vec{\phi_{v_0}}^+)$$

5 Intuitionistic PCF

5.1 Syntax and Typing of Intuitionistic Böhm Trees

Intuitionistic Böhm trees for PCF are defined by the following restricted syntax:

$$\begin{aligned} t &::= \texttt{case}\ x(u, ..., u)\ \texttt{of}\ (c \to t, ..., c \to t) \mid c \\ u &::= \lambda x, ..., x.t \end{aligned}$$

These Böhm trees correspond, up to the undefined Ω, to Hyland-Ong's Finite Canonical Forms of PCF [11].

The typing rules for intuitionistic Böhm trees are:

$$\frac{t : (\Gamma, A_1^{x_1}, ..., A_n^{x_n} \vdash C)}{\lambda x_1, ..., x_n.t : (\Gamma \vdash A_1, ..., A_n \to C)}\ Abs \qquad \frac{c \text{ is in } C}{\vdash c : (\Gamma \vdash C)}\ Cst$$

$$\frac{u_1 : (\Gamma \vdash A_1) \quad \dots \quad u_n : (\Gamma \vdash A_n) \qquad t_1 : (\Gamma \vdash C) \quad \dots \quad t_p : (\Gamma \vdash C)}{\texttt{case}\ x(u_1, ..., u_n)\ \texttt{of}\ (c_1 \to t_1, ..., c_p \to t_p) : (\Gamma \vdash C)}\ App$$

with $(A_1, ..., A_n \to C')^x$ in Γ and $C' = \{c_1, ..., c_p\}$

5.2 Weak-Head Reduction

Environments in the intuitionistic case do not name the continuations:

$$\begin{aligned} s &::= t[e] \\ e &::= w; ...; w \\ w &::= (\vec{x} \leftarrow \vec{u}; \vec{c} \to \vec{t})[e] \end{aligned}$$

The (annotated) rules of reduction differ slightly from the ones of classical case. The first rules becomes

$$\texttt{case}\ x_{ij}(\vec{u})\ \texttt{of}\ (\vec{c} \to \vec{t})[e] \quad \overset{wh(n)=[^p_j}{\longrightarrow}_1 \quad t'[(\vec{y} \leftarrow \vec{u}; \vec{c} \to \vec{t})^n[e]; e_i]$$

with $e = w_1; ...; w_r$ and $w_i = (\vec{x_i} \leftarrow \vec{u_i}; \vec{c_i} \to \vec{t_i})^p[e_i]$ and $u_{ij} = \lambda\vec{y}.t'$. The second becomes

$$c_j[e] \quad \overset{wh(n)=]^p_{c_i}}{\longrightarrow}_1 \quad t_{ij}[()^n[\,]; e_i]$$

with $e = w_1; ...; w_r$ and $w_i = (\vec{x_i} \leftarrow \vec{u_i}; \vec{c_i} \to \vec{t_i})^p[e_i]$ is the first non dummy window (starting from w_1). Thus, intuitionistic PCF returns constants to the last pushed continuation while classical PCF allows to bypass an arbitrary number of continuations.

5.3 Intuitionistic Games

Intuitionistic E-dialogues are E-dialogues where legal positions should satisfy another extra rule.

- *Last asked first answered* condition.

  Let $d_n =]_c^{p_n}$ an answer, and n' such that $p_n < n' < n$. If $d_{n'}$ is an attack then there is n'' such that $n' < n'' < n$ and $d_{n''} =]_{c''}^{n'}$ for some c''. If $d_{n'} =]_{c'}^{p_{n'}}$ is an answer then $p_{n'} \neq p_n$.

Similarly, we get intuitionistic spread E-dialogues by adding the above extra condition to the rules of (classical) spread E-dialogues.

5.4 Debate

The definitions of the debates d_E and d_S are the same for intuitionistic and classical games. Moreover, the propositions 2 and 3 and the corollaries 7 and 9 still hold in the intuitionistic case.

Intuitionistic spread E-dialogues can be understood as Hyland-Ong's games where the innocence condition on strategies is now a rule of the game. As a consequence, all spread E-O-strategies are innocent in our games and therefore in one-to-one correspondence with families of intuitionistic Böhm trees. On the other side, definability for Hyland-Ong's games states that even against a non-innocent opponent (for instance a non-deterministic player), we still have the bijection between P-strategies and Böhm trees.

Acknowledgement

This work has benefited from the discussions with P.-L. Curien, V. Danos and L. Regnier. I specially thank P.-L. Curien for his fruitful feedback on the paper.

References

1. M. Abadi, L. Cardelli, P.-L. Curien and J.-J. Lévy, *Explicit Substitutions*, Journal of Functional Programming 1, pp 375-416, 1991.
2. P. Bernays, *On the Original Gentzen Consistency Proof for Number Theory*, in Intuitionism and Proof Theory, Kino, Myhill & Vesley eds, pp 409-417. Original proof printed in the *The Collected Papers of Gerhard Gentzen* by M. E. Szabo, North Holland, 1969.
3. T. Coquand, *A Semantics of Evidence for Classical Arithmetic, revised version*, Journal of Symbolic Logic, Vol 60, 1995, pp 325-337. First version in: Proceedings of the CLICS workshop, Århus, 1992.
4. T. Crolard, *Extension de l'isomorphisme de Curry-Howard au traitement des exceptions (application d'une étude de la dualité en logique intuitionniste)*, thèse de doctorat, Université Paris 7, 1996.

5. P.-L. Curien and H. Herbelin, *Computing with Abstract Böhm Trees*, submitted, 1996.
6. V. Danos, H. Herbelin, L. Regnier, *Game Semantics and Abstract Machines*, Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science (LICS'96), pp 394-405, 1996.
7. V. Danos, L. Regnier, *Deus Ex Machina*, unpublished paper, 1990.
8. W. Felscher, *Dialogues as a Foundation of Intuitionistic Logic*, Handbook of Philosophical Logic, Vol 3, pp 341-372, 1986.
9. T. Hardin, L. Maranget and B. Pagano, *Functional Back-Ends within the Lambda-Sigma Calculus*, Proceedings, International Conference on Functional Programming (ICFP'96), ACM Press, pp 25-33, 1996.
10. H. Herbelin, *Séquents qu'on calcule: de l'interprétation du calcul des séquents comme calcul de λ-termes et comme calcul de stratégies gagnantes*, thèse de doctorat, Université Paris 7, 1995.
11. M. Hyland, C.-H. L. Ong, *On Full Abstraction for PCF*, submitted, currently available at ftp://ftp.comlab.ox.ac.uk/pub/Documents/techpapers/Luke.Ong/.
12. X. Leroy, *The ZINC Experiment*, Technical Report, number 117, INRIA, 1990.
13. K. Lorenz, *Arithmetik und Logik als Spiele*, Dissertation, Universität Kiel, 1961. Partially reprinted in [15].
14. P. Lorenzen, *Logik und Agon*, in Atti Congr. Internat. di Filosofia, Vol 4, Sansoni, Firenze, pp 187-194, 1960. Reprinted in [15].
15. P. Lorenzen, K. Lorenz, *Dialogische Logik*, Wissentschaftliche Buchgesellschaft, Darmstadt, 1978.
16. P. Lorenzen, K. Schwemmer, *Konstriktive Logik, Ethik und Wissenschafttheorie*, Bibliograph. Institut Mannheim, 1973.
17. C.-H. L. Ong, *A Semantic View of Classical Proofs*, Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science (LICS'96), pp 230-241, 1996.
18. M. Parigot, *$\lambda\mu$-Calculus: an Algorithmic Interpretation of Classical Natural Deduction*, Springer Lecture Notes in Computer Sciences 624, pp 190-201.

This article was processed using the LaTeX macro package with LLNCS style