Game Semantics for the Pure Lazy λ-calculus ⋆

Pietro Di Gianantonio

Dipartimento di Matematica e Informatica, Università di Udine
via delle Scienze 206 I-33100 Udine Italy
e-mail: digianantonio@dimi.uniud.it

Abstract. In this paper we present a fully abstract game model for the pure lazy λ-calculus, i.e. the lazy λ-calculus without constants. In order to obtain this result we introduce a new category of games, the monotonic games, whose main characteristic consists in having an order relation on moves.

1 Introduction

The aim of this paper is to present a fully abstract model, based on game semantics, for the lazy λ-calculus. The λ-calculus we consider is the untyped one, with a lazy, call-by-name, reduction strategy. The model we construct lies in the category of monotonic games introduced in this paper. This new category is derived from the one defined by Abramsky Jagadeesan and Malacaria in [AJM94].

This paper is quite similar to the article [AM95a]. It has the same aims and uses a similar model, but there is also an important difference: the lazy λ-calculus considered in [AM95a] contains a constant C that, in the operational semantics, is able to perform a sequential test for convergence. The introduction of the constant C is essential in order to obtain a full definability result and, as a consequence, the full abstraction of the model. Similarly in [AO93], through syntactic methods, it was obtained a fully abstract model for the lazy λ-calculus extended with the constant C; while the problem of finding a fully abstract model for the pure lazy λ-calculus was left open.

In this paper we show that it is possible to have a fully abstract model for the pure lazy λ-calculus without constants. In order to obtain this result, we need to introduce a new category of games that we call monotonic games and indicate with $\mathcal{G}_{\mathcal{M}}$. The category $\mathcal{G}_{\mathcal{M}}$ differs from the more standard category of AJM-games $\mathcal{G}$ in several aspects. In $\mathcal{G}_{\mathcal{M}}$, moves are questions or answers and are ordered according to a notion of strength. Intuitively, a question a is stronger than a question b if it asks for more information. This means that if question a can receive an answer, then also b can receive an answer, or, from another point of view, a requires more work than b to be fulfilled. Similarly, an answer is stronger than another if it gives more information. Using this notion of

⋆ Research partially supported by Esprit Working Group TYPES and TMR Network LINEAR

strength, we impose some new restrictions on the way that a play can evolve and in the way that a strategy can behave. Intuitively, we ask that a play proceeds with stronger and stronger questions, and that a strategy preserves the strength order relation. By these restrictions, in our game λ-model strategies are forced to behave as interpretations of λ-terms, and hence we have a fully complete and fully abstract model.

2 The calculus

We define here the language λl, together with its operational semantics. Language λl is a lazy λ-calculus, its set of terms constructed from a set of variables $Var(\ni x)$ by the grammar:

$$M ::= x \mid MM \mid \lambda x.M$$

The operational semantics is given by a big-step reduction relation, $M \Downarrow N$, evaluating a term to a weak head normal form. The strategy of evaluation is lazy and call-by-name.

$$\frac{}{\lambda x.M \Downarrow \lambda x.M}$$

$$\frac{M \Downarrow \lambda x.P \qquad P[N/x] \Downarrow Q}{MN \Downarrow Q}$$

The above reduction strategy gives rise to a contextual pre-order ($\sqsubseteq_l$) on closed λ-terms (Λ^0) defined by:

$$M \sqsubseteq_l N \;\Leftrightarrow\; (\forall C[\,] \in \Lambda^0 \,.\, C[M] \Downarrow \Rightarrow C[N] \Downarrow)$$

We indicate with $\approx_l$ the equivalence relation induced by $\sqsubseteq_l$.
The following properties will be used:

- the lazy reduction strategy converges on any term M β-equivalent to a λ-abstraction;
- the relation $\approx_l$ is a $\lambda\beta$-theory.

The above properties follow immediately from the fact that there exist adequate models for the lazy λ-calculus (see [AO93,AM95a,BPR98]).

3 The categories of monotonic games

In this section, we define the two categories of games employed in this article. These two categories are closely related to the categories $\mathcal{G}$ and $K_!(\mathcal{G})$ presented in [AJM94]. They are defined following similar patterns; essentially, they only

differ with respect to the strength order relation on moves. We begin by giving the basic definitions.

As usual, we consider games between two participants: the *Player* and the *Opponent*. A play consists in an alternate sequence of moves, while each move consists in posing a question ($\in M^Q$) or giving an answer ($\in M^A$). Before giving the definition of games, we introduce the notation that will be used in the following.

- We use the metavariables A, B, C to range on games, the metavariables s, t, r, q to range on plays and the metavariables a, b, c to range on moves.
- The empty sequence is denoted by ϵ, concatenation of sequences is denoted by juxtaposition, the prefix relation between sequences is denoted by $\sqsubseteq$.
- Given a sequence s of moves in M and a subset M' of M, $s \restriction_{M'}$ denotes the subsequence of s formed by elements contained in M', and $|s|$ denotes the length of s.
- The function nl (nesting level) from plays to integers is defined as follows:

$$\mathsf{nl}(\epsilon) = 0$$
$$\mathsf{nl}(sa) = |s \restriction_{M^Q}| - |sa \restriction_{M^A}|$$

  In a play questions and answers match like opened and closed parenthesis in an expression, the value $\mathsf{nl}(sa)$ gives the nested level of questions at which the move a lies in the sequence sa. Note that, in the above definition of $\mathsf{nl}(sa)$, the move a is "counted" only if it is an answer; as a consequence, the function nl has the same value on a question and on the corresponding answer.

Definition 1. *A game A is a tuple $(M_A, \lambda_A, \prec_A, P_A, \approx_A)$ where*

- *M_A is a set of* moves*,*
- *$\lambda_A : M_A \to \{O, P\} \times \{Q, A\}$ is the* labelling *function: it tells us if a move is taken by the Opponent or by the Player, and if it is a Question or an Answer. We can decompose λ_A into $\lambda_A^{OP} : M_A \to \{O, P\}$ and $\lambda_A^{QA} : M_A \to \{Q, A\}$ and put $\lambda_A = \langle \lambda_A^{OP}, \lambda_A^{QA} \rangle$. We denote by $^-$ the function which exchanges Player and Opponent, and Question and Answer i.e. $\overline{O} = P$, $\overline{P} = O$, $\overline{Q} = A$ and $\overline{A} = Q$. We also denote with $\overline{\lambda_A^{OP}}$ the function defined by $\overline{\lambda_A^{OP}}(a) = \overline{\lambda_A^{OP}(a)}$ and with $\overline{\lambda_A}$ the function $\langle \overline{\lambda_A^{OP}}, \lambda_A^{QA} \rangle$.*
- *$\prec_A \subseteq (M_A \times M_A)$ is a strict order relation on the set of moves.*
- *P_A is the set of* plays *of the game A, that is a non-empty and prefix-closed subset of the set $M_A^{\circledast}$, where $M_A^{\circledast}$ is the set of all sequences of moves which satisfy the following conditions:*
  - *$s = as' \Rightarrow \lambda_A(a) = OQ$, a play starts with a question made by the Opponent.*
  - *$s = rabt \Rightarrow \lambda_A^{OP}(a) = \overline{\lambda_A^{OP}(b)}$, Player and Opponent alternate.*
  - *$s = rt \Rightarrow \mathsf{nl}(r) \geq (0)$, it is possible to play an answer only if there exists a pending question.*

- $s = qarbt \vee nl(qa) = nl(qarb) \Rightarrow a \prec_A b$, *a question is weaker than the corresponding answer, if an answer a is followed by a new question b, then b is stronger than a, and this condition recursively apply to nested moves.*

– $\approx_A$ *is an equivalence relation on P_A which satisfies the following properties:*

- $s \approx_A \epsilon \;\Rightarrow\; s = \epsilon$
- $sa \approx_A s'a' \;\Rightarrow\; s \approx_A s'$
- $s \approx_A s' \;\wedge\; sa \in P_A \;\Rightarrow\; \exists a' \,.\, sa \approx_A s'a'$
- $sa \approx_A s'a' \wedge sarb \approx_A s'a'r'b' \;\Rightarrow\; ((a \prec_A b \Rightarrow a' \prec_A b') \wedge (b \prec_A a \Rightarrow b' \prec_A a'))$

Definition 2 (Strategies).

A strategy σ in a game A is a non-empty set of plays of even length such that $\sigma \cup dom(\sigma)$ is prefix-closed, where $dom(\sigma) = \{t \in P_A \mid \exists a \,.\, ta \in \sigma\}$.

A strategy can be seen as a set of rules which tells the Player which move to make after the last move by the Opponent.

In this paper we shall consider strategies that are *deterministic*, *history-free* and *monotone*. A strategy is history-free if it depends only on the *last* move by the Opponent; it is monotone if, in some particular cases, it respects the partial order $\prec$ (see bellow).

Before giving the definition of monotone strategy, we need to introduce two new concepts: the set of derived questions and the set of derived answers. The intuitive idea is the following: if, in a play s, a question a of the Opponent is followed by a question b of the Player, one can consider b an effect of question a, since in order to answer to a, the Player needs to know the answer to b. Moreover, if after receiving an answer c to b the Player asks a second question b', this means that the information given by c was not sufficient and new information is required; that is, also b' can be considered a direct consequence of a. The above argument can be repeated until a receives an answer, in this way defining the set of the derived questions of question a. Formally, the set of the derived questions of a question a in a play s is defined by:

$$\mathsf{drv}(s, a) = \{b \mid b \in M^Q,\ s = s'arbs'',\ \mathsf{nl}(r) = 0,\ \forall r' \subseteq r \,.\, nl(r') \geq 0\}$$

Similarly, one can associate to an answer a the set of answers generated thanks to the information given by a, and define the set of the derived answers of an answer a in play s

$$\mathsf{drv}(s, a) = \{b \mid b \in M^A,\ s = s'arbs'',\ \mathsf{nl}(r) = 0,\ \forall r' \subseteq r \,.\, nl(r) \leq 0\}$$

The function drv can be extended to strategies. Given a strategy σ and an Opponent move a, the set of moves derived from a in strategy σ is defined by:

$$\mathsf{drv}(\sigma, a) = \bigcup_{s \in \sigma} \mathsf{drv}(s, a)$$

Definition 3 (Deterministic, history-free and monotone strategies).
A strategy σ for a game A is deterministic if:

$$sb, sc \in \sigma \;\Rightarrow\; b = c$$

The strategy σ is history-free if:

$$sab, t \in \sigma \;\wedge\; ta \in P_A \;\Rightarrow\; tab \in \sigma$$

The strategy σ is monotone if:

$$s \in \sigma \;\wedge\; sa \in P_A \;\wedge\; a' \prec_A a \wedge \mathsf{drv}(\sigma, a') \neq \emptyset \\ \Rightarrow\; \exists b.\, (sab \in \sigma \;\wedge\; \forall b' \in \mathsf{drv}(\sigma, a')\,.\, b' \prec_A b)$$

In the following we implicitly assume strategies to be deterministic, history free and monotone.

The condition of monotonicity requires that if a strategy σ reacts to a question a with another question b (or to an answer a with an answer b), then σ needs to react to any move stronger than a with a move that is stronger than b (and stronger that any other moves derived from a). The notion of derived moves is essential in order to assure that the composition of two monotonic strategies is a monotonic strategies.

The condition of monotonicity is quite strong. In particular, there are "few" finite monotone strategies: in general, a monotone strategy cannot be approximated by a chain of finite and monotone strategies. This shortage of finite strategies is necessary in order to have a full definability result. In game semantics the interpretation of a solvable λ-term is always an infinite object. In [KNO00,KNO99] the semantic interpretations of λ-terms are characterised as almost everywhere copy-cat strategies. On the game λ-models we are going to construct, the condition of monotonicity essentially forces the behaviour of strategies to be almost everywhere copy-cat strategies.

The equivalence relation on plays $\approx$ generates a relation $\sqsubseteq_{\approx}$ and a partial equivalence relation $\approx$ on strategies in the following way.

Definition 4 (Order-enrichment). *Given strategies σ and τ we write $\sigma \sqsubseteq_{\approx} \tau$ iff*

$$sab \in \sigma \wedge\; s' \in \tau \wedge\; sa \approx s'a' \Longrightarrow \exists b'.(s'a'b' \in \tau \;\wedge\; sab \approx s'a'b')$$

The relation $\approx$ on strategies is the reflexive closure of the relation $\sqsubseteq_{\approx}$

It is easy to check that $\approx$ is a partial equivalence relation. It is not an equivalence since it might lack reflexivity. If σ is a strategy for a game A such that $\sigma \approx \sigma$, we write $\sigma : A$. It is also immediate that $\sqsubseteq_{\approx}$ defines a partial order on the equivalence classes of strategies.

Definition 5 (Tensor product).
Given games A and B the tensor product *$A \otimes B$ is the game defined as follows:*

- $M_{A\otimes B} = M_A + M_B$;
- $\lambda_{A\otimes B} = [\lambda_A, \lambda_B]$;
- $\prec_{A\otimes B} = \prec_A \cup \prec_B$;
- $P_{A\otimes B} \subseteq M^{\circledast}_{A\otimes B}$ *is the set of plays,* s*, satisfying the projection condition:* $s\restriction_{M_A} \in P_A$ *and* $s\restriction_{M_B} \in P_B$ *(the projections on each component are plays for the games* A *and* B *respectively);*
- $s \approx_{A\otimes B} s'$ *iff* $s\restriction_A \approx_A s'\restriction_A \ \wedge\ s\restriction_B \approx_B s'\restriction_B \ \wedge\ \forall i.(s_i \in M_A \Leftrightarrow s'_i \in M_A)$.

Here $+$ *denotes disjoint union of sets, that is* $A + B = \{in_l(a) \mid a \in A\} \cup \{in_r(b) \mid b \in B\}$*, and* $[-,-]$ *is the usual (unique) decomposition of a function defined on disjoint unions.*

One should notice that, differently from the standard definition of [AJM94], it is not necessarily to impose the Stack discipline, which says that in a play every answer must be in the same component game as the corresponding question. The stack discipline is forced by monotonicity condition on plays, in fact a question a and the corresponding answer b have the same nested level, therefore $a \prec_{A\otimes B} b$, and by the definition of $\prec_{A\otimes B}$, a and b lie in the same component. It is also useful to observe that if $sab \in P_{A\otimes B}$, and a, b are in different components then $\lambda^{QA}(a) = \lambda^{QA}(b)$. As a consequence, in a product game, only the Opponent can switch component, and this can happen only by reacting to a question of the Player with another question, or by giving an answer in the correct component.

Definition 6 (Unit). *The unit element for the tensor product is given by the empty game* $I = (\varnothing, \varnothing, \varnothing, \{\epsilon\}, \{(\epsilon,\epsilon)\})$.

Definition 7 (Linear implication). *Given games* A *and* B *the compound game* $A \multimap B$ *is defined as follows:*

- $M_{A\multimap B} = M_A + M_B$
- $\lambda_{A\multimap B} = [\overline{\lambda_A}, \lambda_B]$
- $\prec_{A\otimes B} = \prec_A \cup \prec_B$
- $P_{A\multimap B} \subseteq M^{\circledast}_{A\multimap B}$ *is the set of plays,* s*, which satisfy the Projection condition:* $s\restriction_{M_A} \in P_A$ *and* $s\restriction_{M_B} \in P_B$
- $s \approx_{A\multimap B} s'$ *iff* $s\restriction_A \approx_A s'\restriction_A \ \wedge\ s\restriction_B \approx_B s'\restriction_B \wedge \forall i\,.\,(s_i \in M_A \Leftrightarrow s'_i \in M_A)$

By repeating the arguments used for the tensor product, it is not difficult to see that in a "linear implication game" only the Player can switch component, and this can happen only by reacting to question of the Opponent with another question, or by giving an answer in the correct component.

Definition 8 (Exponential). *Given a game* A *the game* $!A$ *is defined as follows:*

- $M_{!A} = \mathbb{N} \times M_A = \sum_{i\in\mathbb{N}} M_A$
- $\lambda_{!A}(\langle i, a\rangle) = \lambda_A(a)$
- $(i,a) \prec_{!A} (j,b)$ *iff* $i = j$ *and* $a \prec_A b$

– $P_{!A} \subseteq M_{!A}^{\circledast}$ *is the set of plays,* s*, which satisfy the conditions:*
$\forall i \in \mathbb{N} \,.\, s \restriction_{A_i} \in P_{A_i}$
– $s \approx_{!A} s'$ *iff there exists a permutation of* α *on* $\mathbb{N}$ *such that:*
  • $\pi_1^*(s) = \alpha^*(\pi_1^*(s'))$
  • $\forall i \in \mathbb{N} \,.\, (\pi_2^*(s' \restriction_{\alpha(i)}) \approx \pi_2^*(s \restriction_i))$

  where α^* *denotes the pointwise extension of the function* α *to sequence of naturals.* π_1 *and* π_2 *are the projections of* $\mathbb{N} \times M_A$ *and* $s \restriction_i$ *is an abbreviation of* $s \restriction_{A_i}$*.*

Definition 9 (The category of games $\mathcal{G}_{\mathcal{M}}$).

The category $\mathcal{G}_{\mathcal{M}}$ *has as objects games and as morphisms, between games* A *and* B*, the equivalence classes, w.r.t. the relation* $\approx_{A \multimap B}$*, of deterministic, history-free and monotone strategies* $\sigma : A \multimap B$*. We denote the equivalence class of* σ *by* $[\sigma]$*.*

The identity for each game A *is given by the (equivalence class) of the* copy-cat *strategy, recursively defined as follows,*

$$id_A = \{sa'a'' \in P_{A \multimap A} \mid s \in id_A, \{a', a''\} = \{in_l(a), in_r(a)\}\} \cup \{\epsilon\}$$

Composition is given by the extension to equivalence classes of the following composition of strategies. Given strategies $\sigma : A \multimap B$ *and* $\tau : B \multimap C$*,* $\tau \circ \sigma : A \multimap C$ *is defined by*

$$\tau \circ \sigma = \{s \restriction_{(A,C)} \mid s \in (M_A + M_B + M_C)^* \;\&\; s \restriction_{(A,B)} \in \overline{\sigma}, s \restriction_{(B,C)} \in \overline{\tau}\}^{even}$$

where with S^{even} *denote the set of plays in* S *having even length.*

The correctness of the above definition follows, in part, from the correctness of the definition of AJM-games. In addition we need to prove that:

– id_A is a monotone strategy,
– the composition of two monotone strategies is again a monotone strategy.

The monotonicity of id_A follows immediately from the fact that for every pair of moves $a \in M_A^P$, $b \in M_A^O$, $\mathsf{drv}(id_A, \mathrm{in}_l(a)) = \{\mathrm{in}_r(a)\}$ and $\mathsf{drv}(id_A, \mathrm{in}_r(b)) = \{\mathrm{in}_l(b)\}$. The preservation of monotonicity by strategy composition follows easily from the fact that for every pair of strategies $\sigma : A \multimap B$ and $\tau : B \multimap C$ and for every pair of moves $a \in M_A^P$ and $c \in M_C^O$, if $a \in \mathsf{drv}(\tau \circ \sigma, c)$ then there exists a chain $b_1, \ldots b_{2n+1}$ such that $b_1 \in \mathsf{drv}(\tau, c)$, $a \in \mathsf{drv}(\sigma, b_{2n+1})$, $\forall i] in \{0 \ldots n\} \,.\, b_{2i} \in \mathsf{drv}(\sigma, b_{2i-1})$, $b_{2i+1} \in \mathsf{drv}(\tau, b_{2i})$.

The constructions introduced in Definitions 5, 7 and 8 can be made to be functorial.

Definition 10. *Given two strategies* $\sigma : A \multimap B$ *and* $\sigma' : A' \multimap B'$ *the strategies* $\sigma \otimes \sigma' : (A \otimes A') \multimap (B \otimes B')$*,* $\sigma \multimap \sigma' : (A \multimap A') \multimap (B \multimap B')$*,* $!\sigma : !A \multimap !B$ *are recursively defined as follows:*

$$\sigma \otimes \sigma' = \{sab \in P_{(A \otimes A') \multimap (B \otimes B')} \mid s \in \sigma \otimes \sigma',\ sab \restriction_{M_A \cup M_B} \in \sigma,\ sab \restriction_{M_{A'} \cup M_{B'}} \in \sigma'\} \cup \{\epsilon\}$$

$$\sigma \multimap \sigma' = \{sab \in P_{(A \multimap A') \multimap (B \multimap B')} \mid s \in \sigma \multimap \sigma',\ sab\restriction_{M_A \cup M_B} \in \sigma,\ sab\restriction_{M_{A'} \cup M_{B'}} \in \sigma'\} \cup \{\epsilon\}$$

$$!\sigma = \{s \in P_{!A \multimap !B} \mid \forall i \,.\, s\restriction_{M_{A_i} \cup M_{B_i}} \in \sigma\}$$

It is not difficult to check that the above definitions are correct and that $\otimes$ and I indeed provide a categorical tensor product and its unit.

The category $\mathcal{G}_{\mathcal{M}}$ is monoidal closed, but not Cartesian closed. Analogously to what happens in AJM-games, a Cartesian closed category of games can be obtained by taking the co-Kleisli category $K_!(\mathcal{G}_{\mathcal{M}})$ over the co-monad $(!, \mathtt{der}, \delta)$, where for each game A the strategies $\mathtt{der}_A : !A \multimap A$ and $\delta_A : !A \multimap !!A$ are defined as follows:

- $\mathtt{der}_A = [\{s \in P_{!A \multimap A} \mid s\restriction_{(!A)_0} = s\restriction_A\}]$
- $\delta_A = [\{s \in P_{!A \multimap !!A} \mid s\restriction_{(!A)_{p(i,j)}} = s\restriction_{(!(!A)_i)_j}\}]$ where $p : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is a pairing function

Hence one can easily see that the following definitions are well posed.

Definition 11 (A Cartesian closed category of games).
The category $K_!(\mathcal{G}_{\mathcal{M}})$ has as objects games and as morphisms between games A and B the equivalence classes of history-free strategies in the game $!A \multimap B$.

In order to give semantics to the lazy λ-calculus, it is necessary to define the lifting constructor.

Definition 12 (Lifting).
Given a game A, the lifted game $A_\perp$ is defined as follows:

- $M_{A_\perp} = M_A + \{\circ, \bullet\}$
- $\lambda_{A_\perp} = [\lambda_A, \{\circ \to OQ, \bullet \to PA\}]$
- $\prec_{A_\perp} = \prec_A \cup \{\langle b, a\rangle \mid a \in M,\ b \in \{\circ, \bullet\}\} \cup \{\langle \circ, \bullet\rangle\}$
- $P_{A_\perp} = \{\epsilon, \circ\} \cup \{\circ\bullet s \mid s \in P_A\}$
- $s \approx_{A_\perp} s'$ *iff* $s = s'$ *or*
  $s = \circ\bullet t$ *and* $s' = \circ\bullet t'$ *and* $t \approx_A t'$

Note that the above definition cannot be made functorial, at least not in a standard way. Given a strategy $\sigma : A \to B$, strategy $\sigma_\perp : A_\perp \to B_\perp$ is usually defined ([AM95a]) by:

$$\sigma_\perp = \{\circ_A\circ_B\bullet_B\bullet_A s \mid\ s \in \sigma\} \cup \{\epsilon, \circ_A\circ_B\}$$

In the category $\mathcal{G}_{\mathcal{M}}$, with the above definition, $\sigma_\perp$ is not necessarily a monotone strategy. In fact, the initial behaviour of $\sigma_\perp$ ($\circ_A\circ_B\bullet_B\bullet_A \in \sigma_\perp$) imposes conditions on the future behaviour of the strategy that are not necessarily satisfied.

However, given any game A having a single initial move a, it is possible to define two strategies: $\mathsf{up}_A : A \multimap A_\perp$ and $\mathsf{dn}_A : A_\perp \multimap A$ as follows:

$$\mathsf{up}_A = \{\circ_{A_\perp}\bullet_{A_\perp} s \mid\ s \in id_A\} \cup \{\epsilon\}$$

$$\mathsf{dn}_A = \{a \circ_{A_\perp} \bullet_{A_\perp} s \in P_{!(A_\perp \multimap A} \mid as \in id_A\} \cup \{\epsilon, a \circ_{A_\perp}\}$$

It is not difficult to prove that the above strategies are well defined and that $\mathsf{dn}_A \circ \mathsf{up}_A \approx id_A$.

In order to define a model for the lazy λ calculus, the functoriality of the lifting constructor is not necessary, the existence of the strategies, dn_A and up_A suffices.

4 Solution of recursive game equations

The categories of games $\mathcal{G}_\mathcal{M}$ and $K_!(\mathcal{G}_\mathcal{M})$ allow for the existence of *recursive* objects, *i.e.* objects that are fixed points of game constructors. We present the method proposed by Abramsky and McCusker ([AM95b]) for defining recursive games. This method allows to define *initial* fixed points for a large set of functors and it follows the pattern used for building initial fixed points in the context of information systems. First a complete partial order $\trianglelefteq$ on games is introduced.

Definition 13. *Let A, B be games, A is a* sub-game *of B ($A \trianglelefteq B$) if*

- $M_A \subseteq M_B$;
- $\lambda_A = \lambda_B \restriction_{M_A}$;
- $\prec_A = \prec_B \restriction_{M_A}$;
- $P_A = P_B \cap M_A^{\circledast}$;
- $s \approx_A s'$ *iff* $s \approx_B s'$ *and* $s \in P_A$.

One can easily see that the sub-game relation defines a complete partial order on games. Hence a game constructor F which is continuous with respect to $\trianglelefteq$ has a (minimal) fixed point $D = F(D)$ given by $\bigsqcup_{n\in\mathbb{N}} F^n(I)$. Notice that we have indeed an identity between D and $F(D)$ and that we do not need the game constructor F to be a functor; as a result we can also apply this method to the lifting game constructor.

One can easily see that the game constructors $\otimes$, $\multimap$, $!$, $(\ )_\perp$ and their compositions are continuous with respect to $\trianglelefteq$; therefore, the method applies to them.

5 Lazy λ-models in $K_!(\mathcal{G}_\mathcal{M})$

A standard way to construct a model for the lazy λ-calculus consists in taking the initial fixed-point of the functor $F(D) = (D \to D)_\perp$ [AO93], [AM95a], [BPR98], [EHR92]. Here we use the same technique. We denote with D the least fixed-point of the game constructor: $F(A) = (A \to A)_\perp = (!A \multimap A)_\perp$ in the category of monotonic games ($D = \bigcup F^n(I)$). We denote with $\varphi : !D \multimap (!D \multimap D)$ the morphism $\mathsf{dn}_{!D\multimap D} \circ \mathsf{der}_D$ and with $\psi : !(!D \multimap D) \multimap D$ the morphism $\mathsf{up}_{!D\multimap D} \circ \mathsf{der}_{!D\multimap D}$.

The morphisms φ and ψ define a retraction between D and $!D \multimap D$ such that $\psi \circ \perp \not\approx_D \perp$, where with $\perp$ we indicate the smallest strategy $\{\epsilon\}$. It follows that the tuple $\mathtt{D} = \langle D, \varphi, \psi\rangle$ defines a categorical model of the lazy λ-calculus.

Definition 14. *The interpretation of a λ-term M (whose free variables are among the list $\Gamma = \{x_1, \ldots, x_n\}$) in the model $\mathtt{D} = \langle D, \varphi, \psi \rangle$ is strategy $[\![M]\!]_\Gamma : \overbrace{(!D \otimes \cdots \otimes !D)}^{|\Gamma|} \to D$ defined inductively as follows:*

$[\![x_i]\!]_\Gamma = \pi_i^\Gamma$;
$[\![MN]\!]_\Gamma = ev \circ \langle (\varphi \circ [\![M]\!]_\Gamma) \ , \ [\![N]\!]_\Gamma \rangle$;
$[\![\lambda x.M]\!]_\Gamma = \psi \circ \Lambda(\ [\![M]\!]_{\Gamma,x})$;

where π_i^Γ are the canonical projection morphisms, ev and Λ denote "evaluation" and "abstraction" in the Cartesian closed category $K_!(\mathcal{G}_\mathcal{M})$.

It is useful to give some intuitive explanations concerning the plays in the game λ-model D. In game D the Opponent can be identified with the environment, while the Player can be identified with a program (λ-term) interrogated by the Opponent. A possible play s in game D proceeds as follows: the initial move of s is a request of the Opponent to know if the Player is a λ-abstraction; the Player fails to reply if it is a strongly unsolvable term, otherwise it answers (positively) to the question. After that, the play proceeds by a consecutive question of the Opponent asking if there is another λ-abstraction inside the first λ-abstraction. Again the Player may fail to answer, if it is an unsolvable term of order 1, or it may answer, if it contains two λ-abstractions. This time however, the Player can also pose a question to the Opponent; this happens if the Player is in the form $\lambda x \, . \, xM_1 \ldots M_m$. In this case, the Player contains a second λ-abstraction depending on value (behaviour) of x (the first argument passed by the Opponent). In particular, in the above case, the Player needs to check whether x contains $m + 1$ λ-abstractions. In reaction to the questions of the Player, on the argument x, the Opponent can reply by posing questions on the arguments passed to x; in this case, the Player will answer according to the terms M_i. The plays can proceed with questions and answer in an arbitrary nested level, with the Opponent asking information on the deeper structure of the Player.

The order relation $\prec_D$ models the fact that consecutive questions, at the same nested level, ask for more and more λ-abstractions. The condition of monotonicity on plays models the fact that the questions posed by of a λ-term (Player) at nested level 1 always concern the argument appearing as head variable.

More formally, we present a set of results describing the theory induced by the game λ-model D. Since $\mathtt{D}$ is a λ model and since the interpretation of a λ-abstraction is never equivalent to strategy $\bot$, one immediately has:

Proposition 1. *For any pair of closed λ-terms M, N, if $M \Downarrow N$ then $[\![M]\!] \approx_D [\![N]\!] \neq \bot$.*

The proof of adequacy need to be more complex. In general sophisticated proof techniques, such as the computability method, the invariants relations or the approximation theorem, are needed to prove the adequacy of a model. In

this case we can use a previous result concerning the games semantics of the untyped λ-calculus. In this way we are also able to characterize precisely the theory induced by D.

In [DGF00], a complete characterisation of the theories induced by game models in the category $\mathcal{G}$ of games and history free strategies has been carried out. In particular it has been shown that every categorical game model $\langle A, \varphi_A, \psi_A \rangle$, such that $\psi_A \circ \bot \neq \bot$, induces the theory $\mathcal{LT}$. In theory $\mathcal{LT}$ two terms are identified if and only if they have the same Lévy-Longo tree [L75,Lon83]. We briefly recall the definitions.

Definition 15. *Let* $\Sigma = \{\lambda x_1 \dots x_n.\bot \mid n \in \mathbb{N}\} \bigcup \{T\} \bigcup \{\lambda x_1 \dots x_n.y \mid n \in \mathbb{N}\}$, *with* $x_1, \dots, x_n, y \in Var$

Lévy-Longo tree associated to λ*-term* M, $LLT(M)$, *is a* Σ*-labelled infinitary tree defined informally as follows:*

– $LLT(M) = T$ *if* M *is unsolvable of order* ∞, *that is for each natural numbers* n *there exists a* λ*-term* $\lambda x_1 \dots x_n.M'$ β*-equivalent to* M.

– $LLT(M) = \lambda x_1 \dots x_n.\bot$ *if* M *is unsolvable of order* n

– $LLT(M) = \lambda x_1 \dots x_n.y$

$$\begin{array}{c} \\ / \quad \backslash \\ LLT(M_1) \dots LLT(M_m) \end{array}$$

if M *is solvable and has principal head normal form* $\lambda x_1 \dots x_n.yM_1 \dots M_m$.

The arguments used in [DGF00] can be straightforwardly applied also to category $\mathcal{G}_{\mathcal{M}}$. In particular, through an application of the Approximation Theorem it is possible to derive that:

Proposition 2. *For any pair of closed* λ*-terms* M, N, *if* $LLT(M) \Downarrow LLT(N)$ *then* $[\![M]\!] \approx_D [\![N]\!]$.

Proposition 3 (Adequacy and Soundness). *For any pair of closed* λ*-terms* M, N*:*

– *if* $[\![M]\!] \not\approx_D \bot$ *then* $M \Downarrow$;
– *if* $[\![M]\!] \sqsubseteq\!\!\!\approx_D [\![N]\!]$ *then* $M \sqsubseteq_l N$.

Proof. By Proposition 2, if a λ-term M is such that $[\![M]\!] \not\approx_D \bot$, then M is not strongly unsolvable, and the lazy reduction strategy converges on M (see Section 2). The second point is readily proved observing that denotational semantics is compositional and monotonic, therefore for every closed context $C[\,]$, if $[\![M]\!] \sqsubseteq\!\!\!\approx_D [\![N]\!]$, then $[\![C[M]]\!] \sqsubseteq\!\!\!\approx_D [\![[N]]\!]$ and therefore $C[M] \Downarrow \Rightarrow C[N] \Downarrow$. □

5.1 Extensional collapse

Theory $\mathcal{LT}$ is strictly weaker than theory λ_l; for example, the terms $\lambda x.xx$ and $\lambda x.x(\lambda y.xy)$ have different Lévy-Longo trees but they are equated in λ_l. In order to obtain a fully abstract model, we need to interpret λ-terms in the category $\mathcal{E}_{\mathcal{M}}$, defined as an extensional collapse of category $\mathcal{G}_{\mathcal{M}}$. We need to use the Sierpinski game, that is the game $I_\bot$.

Definition 16 (Intrinsic pre-order). *Give a game A, the intrinsic pre-order $\preceq_A$ on the strategies for A is defined by:*

$$\sigma_1 \lesssim_A \sigma_2 \quad \textit{iff} \quad \forall \tau : A \multimap I_\perp \;.\; \tau \circ \sigma_1 \sqsubseteq_{I_\perp} \tau \circ \sigma_2$$

In the above expression we implicitly coerce the strategies in A into $I \multimap A$.

We indicate with $\simeq_A$ the partial equivalence relation induced by $\lesssim_A$.

The category $\mathcal{E}_\mathcal{M}$ has as objects games and as morphism equivalence classes w.r.t. $\simeq$ of strategies.

It is not difficult to verify that intrinsic pre-order is preserved by all the categorical constructions presented above. Therefore, the category $\mathcal{E}_\mathcal{M}$ can be used in modeling the λ-calculus. In particular, the game λ-model $\mathtt{D}$ with the interpretation of Definition 14 gives rise to a λ-model also inside the category $\mathcal{E}_\mathcal{M}$.

6 Full-abstraction

As usual, the proof of full-abstraction splits in two proofs.

Theorem 1 (Soundness). *For any pair of closed λ-terms M, N, we have:*

$$[\![M]\!] \lesssim_D [\![N]\!] \;\Rightarrow\; M \sqsubseteq_l N$$

Proof. It is immediate to check that the only strategy $\simeq_D$-equivalent to $\perp$ is strategy $\perp$ itself. It follows that model $\mathtt{D}$ is adequate also in category $\mathcal{E}_\mathcal{M}$. By the compositionality of the interpretation, soundness follows immediately. □

In order to prove completeness, some preliminary results need to be presented.

Proposition 4. *The following properties hold in the game λ-model D:*

(i) *Every question has one only possible answer and every answer has one only possible consecutive question. Formally, for every pair of plays $sab, tab' \in P_D$ if $\lambda_D^{QA}(b) = \lambda_D^{QA}(b') = \overline{\lambda_D^{QA}(a)}$ then $b = b'$.*

(ii) *For every move $a \in M_D$, the set of predecessors of a, w.r.t. the order $\prec_D$, is a finite and linearly ordered set.*

(iii) *With respect to the order $\prec_D$, every question move has one successor, the corresponding answer; while every answer move a has infinitely many immediate successors which are the consecutive question at the same nested level, and an infinite number of questions at the next nested level (these questions are the initial moves of a subcomponent $!D$ of game D).*

(iv) *For every strategy $\sigma : D$ and for every move $a \in M_D$ the set $\mathsf{drv}(\sigma, a)$ is linearly order w.r.t. $\prec_D$*

Proof. The first three points can be proved by induction on the chain of games $F^n(I)$. Point (iv) follows from point (i). □

Lemma 1. *Any strategy $\alpha : D \to I_\perp$ can be extended to a strategy $\alpha_D : D \to D$ such that for any strategy $\sigma : D$, $\circ : \bullet \ \in \alpha \circ \sigma$ if and only if $\circ : \bullet \ \in \alpha_D \circ \sigma$*

Proof. Since game $D \to I_\perp$ is a sub-game of game $D \to D$, it is sufficient to extend α to a monotone strategy on game $D \to D$, which can be done incrementally. Let $\alpha_0, \ldots, \alpha_n, \ldots$ be an infinite chain of strategies constructed in the following way:

$\alpha_0 = \alpha$

$\alpha_{i+1} = \alpha_i \cup \{sab \in P_{D\to D} \mid s \in \alpha_i,\ sa \in P_{D\to D},$
$\bigcup_{a' \in M^O, a' \prec a} \mathsf{drv}(\alpha_i, a') \neq \emptyset,$
b minimal upper bound in M^P of $\bigcup_{a' \in M^0, a' \prec a} \mathsf{drv}(\alpha_i, a')\}$.

In the above definition the choice of the element b is not necessarily unique. In some cases the minimal upper bounds of $\bigcup_{a' \in M^0, a' \prec a} \mathsf{drv}(\alpha_i, a')$ can form a countable set: the initial questions or answers in the some subcomponent $!D$ of game D. In these cases, almost any possible choice gives rise to an equivalent (w.r.t. $\approx_D$) strategy; some care have to be taken when the move a is itself an initial question or answer in some other subcomponent $!D$ of game D, in which case it is sufficient to choose for b the same index as for the move a.

Strategy α_D is finally defined as $\alpha_D = \bigcup_{n \in N} \alpha_n$. □

It is interesting to observe that if one performs the above construction starting from strategy $\alpha = \{\epsilon,\ \circ\circ, \circ\circ\bullet\bullet\}$, one obtains a strategy $\alpha_D \approx_D id_D$.

Proposition 5 (Definability). *For any play s in game D such that $\mathsf{nl}(s) = 0$, there exists a closed λ-term M such that $s \in [\![M]\!]$ and $[\![M]\!] \sqsubseteq\!\!\!\approx \sigma$ for any strategy σ with $s \in \sigma$.*

To the above proposition, we just give an informal and intuitive proof. A formal proof will require the introduction of several new concepts and will be more difficult to grasp.

We will associate to play s a Lévy-Longo tree or equivalently a λ-term that represents a Lévy-Longo tree. In order to do that, we decompose play s in several levels, each level determining a node of the Lévy-Longo tree.

We need to introduce some notation. Given a play t and an interval I of natural numbers, we denote with $t\restriction_I$ the subsequence of t formed by the moves whose nested level is a value in the interval I.

Sequence $s\restriction_{[0,1]}$ is a play contained in strategy σ. In fact, sequence $s\restriction_{[0,1]}$ describes the behaviour of strategy σ on the hypothesis that the Opponent answers immediately to questions posed by the Player (without posing nested questions). Since, in game D, the Opponent is always allowed to answer immediately to the questions of the Player, the sequence $s\restriction_{[0,1]}$ is a play. Since in $s\restriction_{[0,1]}$ the behaviour of the Player, in reaction to the last move of the Opponent, is the same that in s, and since σ is a history free strategy, it follows that $s\restriction_{[0,1]} \in \sigma$.

Play $s\restriction_{[0,1]}$ can be in one of the following forms:

– the Player always answers to the questions of the Opponent

– the Player answers for n times to the questions of the Opponent, then at the $n+1$ question q of the Opponent, the Player replays posing a question. In this case, the Player is in the position to make the second move in a game having form: $\underbrace{!D \multimap (\ldots (!D}_{n} \multimap D) \ldots)$ and it can choose to pose a question in one of the n instances of D staying on the left of an arrow. After that, the monotonicity condition on plays forces the Player to react to the answer of the Opponent by either posing a consecutive question in the same component either by giving an answer to question q. In all cases, after having posed m consecutive questions in the same component, the Player will answer to question q. The condition of monotonicity on strategies now forces play $s \restriction_{[0,1]}$ to proceed in only one possible way. At the consecutive question of the Opponent the Player needs to reply with a move that is stronger (w.r.t the $\prec_D$ order) to the last question posed by the Player (q_p). This implies that the Player needs to pose the question consecutive to question q_p. At the answer of the Opponent the Player, by the monotonicity condition on strategies, needs to reply with an answer (only one answer available). And the previous argument applies to all consecutive moves.

In the first case above, play s has all moves at the nested level 0, and it is possible to check that: $s \in [\![\lambda x_1 \ldots x_n.\Omega]\!]$ and that $[\![\lambda x_1 \ldots x_n.\Omega]\!] \sqsubseteq_{\approx} \sigma$. On the second case, it is possible to check that: $s \restriction_{[0,1]} \in [\![\lambda x_1 \ldots x_n.x_i \underbrace{\Omega \ldots \Omega}_{m}]\!]$. Moreover, it is possible to prove that $[\![\lambda x_1 \ldots x_n.x_i \Omega \ldots \Omega]\!] \sqsubseteq_{\approx} \sigma$; this can be done proving, by induction on the length of plays, that the monotonicity condition forces strategy σ to behave in a copy-cat way.

Sequence $s \restriction_{[2,\infty]}$ is a play in a game in the form $!D \multimap !D \ldots \multimap (!D \otimes \ldots \otimes !D)$, where the instances of D, on the left of the $\multimap$ arrow, denote variables that can be interrogated by the Player, and the instances of D on the right of the $\multimap$ arrow denote the arguments of the head-variable. The first move in $s \restriction_{[2,\infty]}$ is a question of the Opponent asking if one of the arguments of the head variable is a λ-abstraction and $s \restriction_{[2,3]}$ is a play in game $!D \multimap !D \ldots \multimap (!D \otimes \ldots \otimes !D)$ defining the external structure of one argument of the head-variable. It follows that there exists a λ-term $P_1 = \lambda x_1 \ldots x_n.x_i \Omega \ldots (\lambda x_{j1} \ldots x_{jn_j}.x_k \Omega \ldots \Omega) \ldots \Omega$ such that $s \restriction_{[0,3]} \in [\![P]\!]$ and $[\![P]\!] \sqsubseteq_{\approx} \sigma$.

The above analysis can be repeated for the consecutive levels of nested moves, each slice $s \restriction_{[2i,2i+1]}$ representing a play where the Opponent interrogates the Player in order to know the structure of some subterms of the Player. In this way play s determines a Lévy-Longo tree approximation of strategy σ at which s belongs.

From the above propositions one can finally conclude:

Theorem 2 (Completeness). *For any pair of closed λ-terms M, N, we have:*

$$M \sqsubseteq_l N \;\Rightarrow\; [\![M]\!] \lesssim_D [\![N]\!]$$

Proof. Suppose there exists a strategy α such that $\alpha \circ [\![M]\!] \neq \bot$, by Lemma 1, there exists a minimal strategy α_D such that $\alpha_D \circ [\![M]\!] \neq \bot$. Let $\circ s \bullet$ be the (initial) sequence of moves generated by in the interaction between the strategies α_D and $[\![M]\!]$. Play $\circ s \bullet$ is contained in strategy $\alpha_D : D \to D$, while play $t = \circ \bullet \circ s \bullet$ is contained in strategy $\mathsf{up} \circ \alpha_D : D$. Let P be the term defining the minimal strategy containing the play t. By a simple calculation it follows that $[\![PM]\!] \neq \bot$ and the following chain of implications is immediate: $[\![PM]\!] \neq \bot \;\Rightarrow\; PM \Downarrow \Rightarrow PN \Downarrow \Rightarrow\; [\![PN]\!] \neq \bot \;\Rightarrow\; \alpha_D \circ [\![N]\!] \neq \bot \;\Rightarrow\; \alpha \circ [\![N]\!] \neq \bot$. □

References

[AJM94] S. Abramsky, R. Jagadeesan, and P. Malacaria. Full abstraction for pcf (extended abstract). In *Theoretical Aspects of Computer software. International Symposium TACS '94*, volume 789 of *LNCS*. Springer, 1994.

[AM95a] S. Abramsky and G. McCusker. Games and full abstraction for the lazy λ-calculus. In *Proceedings LICS '95*, 1995.

[AM95b] S. Abramsky and G. McCusker. Games for recursive types. In I. C. Mackie C. L. Hankin and R. Nagarajan, editors, *Theory and Formal Methods of Computing 1994: Proceedings of the Second Imperial College Department of Computing Workshop on Theory and Formal Methods*. Imperial College Press, October 1995.

[AO93] S. Abramsky and C.H.L. Ong. Full abstraction in the lazy λ-calculus. *Information and Computation*, 105:159–267, 1993.

[BPR98] O. Bastonero, A. Pravato, and S Ronchi. Structures for lazy semantics. In *Programming Concepts and Methods, PROCOMET'98*. Chapman & Hall, 1998.

[DGF00] P. Di Gianantonio and G. Franco. The fine structure of game lambda-models. In *Conference on the Foundation of Software Technology and Theoretical Computer Science (FSTTS '00)*, volume 1974 of *LNCS*, pages 429–441. Springer, 2000.

[EHR92] L. Egidi, F. Honsell, and S Ronchi. Operational, denotational and logical description: A case study. *Fundamenta Informaticae*, 16(2):149–170, 1992.

[KNO99] A. D. Ker, H. Nickau, and C. H. L. Ong. A universal innocent game model for the Böhm tree lambda theory. In *Computer Science Logic: Proceedings of the 8th Annual Conference of the EACSL Madrid, Spain*, volume 1683 of *Lecture Notes in Computer Science*, pages 405–419. Springer, September 1999.

[KNO00] A. D. Ker, H. Nickau, and C. H. L. Ong. Innocent game models of untyped lambda calculus. To appear in *Theoretical Computer Science*, 2000.

[L75] J.J. Lévy. An algebraic interpretation of λ-calculus and a labelled λ-calculus. In *Lambda Calculus and Computer Science*, volume 37 of *LNCS*, pages 147–165. Springer-Verlag, 1975.

[Lon83] G. Longo. Set-theoretical models of λ-calculus: theories, expansions and isomorphisms. *Annals of Pure and Applied Logic*, 24:153–188, 1983.