Available online at www.sciencedirect.com

SCIENCE DIRECT®

Theoretical Computer Science 333 (2005) 199–224

Theoretical Computer Science

www.elsevier.com/locate/tcs

Game semantics and linear CPS interpretation☆

J. Laird

Department of Informatics, University of Sussex, Sussex, UK

Abstract

We present a semantic analysis of the "linearly used continuation-passing interpretation" of functional languages, based on game semantics. This consists of a category of games with a coherence condition on moves—yielding a fully complete model of an affine-type theory—and a *syntax-independent* and *full* embedding of a category of Hyland–Ong/Nickau-style "well-bracketed" games into it. We show that this embedding corresponds precisely to linear CPS interpretation in its action on a games model of call-by-value PCF, yielding a proof of *full abstraction* for the associated translation. We discuss extensions of the semantics to deal with recursive types, call-by-name evaluation, non-local jumps, and state.

Keywords: Game semantics; Continuation passing; Linear type theory

1. Introduction

Continuation-passing-style (CPS) interpretation is widely used for implementing and reasoning about typed and untyped functional languages. However, a limitation of the standard CPS interpretation is its failure to capture the constraints on control flow which typically exist in such languages. This is because continuations become first-class objects in the interpretation which may passed as arguments to procedures, although this corresponds to behaviour in the source language only if the latter also treats continuations as first-class objects (using a construct such as **call**/**cc**).

One solution to this problem is based on the observation that continuations in functional languages are always used *linearly* [11,17,8]. Thus, we may obtain a finer-grained analysis

☆ This research was supported by UK EPSRC Grant GR/N 38824.

E-mail address: jiml@sussex.ac.uk (J. Laird).

0304-3975/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2004.10.022

of CPS translation, for example, by using a target language with *linear types*. This paper is a semantic investigation of such a "linear CPS interpretation".

Our interpretation will be based on *game semantics*. Games have been used to give models of both purely functional languages [6,14,24,22,3] and other features, including first-class continuations [16], but also state [2,5] and other side-effects, which are free of "junk". We will prove such a result for a target model for linear CPS, using the fact that games allow intensional phenomena such as control flow and linear use of resources to be represented concretely in terms of behavioural constraints.

1.1. Related work

Connections between CPS and linearity have been made by several researchers, including Filinski [11]. Danvy and Pfenning [10], observed linearity invariants for CPS which have subsequently been formalized by Polakow and Pfenning using *ordered* linear logic [29]. This work is associated with the use of linearity in the analysis of CPS implementations, which could also be studied from a game semantics perspective, but it will not be pursued here.

Explicit linear typings for CPS translations (such as that used here for the call-by-value λ-calculus) were given by Berdine et al. [8]. Their work also examines of the rôle played by linear continuations in a range of other control features (short of first class continuations) such as exceptions, jumps and coroutines. In a companion paper [9], the problem of showing that the first of these translations is complete (in the sense that the target language does not contain any "junk" at linear types) is considered, with a result dependent on a heavy restriction of the types in the target language. Other work along these lines includes a simple and fully complete linear CPS translation of (finitary) PCF presented independently by the author [17]. Hasegawa [12] describes a similar result for call-by-value in a more general setting. We will show how similar results may be obtained by semantic means.

On the applications side, Zdancewic and Myers [32] have describe a linear CPS interpretation of a language with security, using ordered linear continuations to reason about interference between high- and low-security variables. By combining the approach described here with work on the game semantics of references [5], we would seem to have the basis for a semantic approach to analysing linear continuations and security, but much work remains to be done.

A game semantic study of CPS interpretation appeared in the author's Ph.D. Thesis [16]. This contains results analogous to those described here; it was shown that a "direct" games model of PCF with first-class continuations is equivalent to an interpretation via CPS translation.

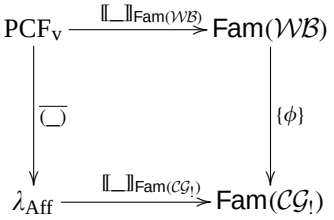
1.2. Overview

The primary aim of this paper is to give a (games) semantics of linear CPS interpretation by describing a category of "coherence games", $\mathcal{CG}$, with which we model a target language for linear CPS translation (λ_{Aff}), a dual affine/intuitionistic λ-calculus) and a functor into $\mathcal{CG}$ from a category of games used to model call-by-value PCF, and showing that this functor corresponds to linear CPS translation.

We obtain coherence games by adding a "coherence relation" to Hyland–Ong-style arenas. We show that this yields sufficient structure to model λ_{Aff} (a symmetric monoidal product with certain exponentials, cartesian products and a co-monad !) and that this model is universal for λ_{Aff} extended with a constant for divergence.

We then define a full and faithful functor ϕ from a category $\mathcal{WB}$ of "well-bracketed" Hyland–Ong/Nickau-style games (essentially those used to model PCF in [14,24]) into the co-Kleisli category $\mathcal{CG}_!$, using the question and answer labelling on each well-bracketed arena to define an associated coherence relation.

We show that the extension of λ_{Aff} with primitive types such as `nat` can be interpreted in a category of indexed families of coherence games $\mathsf{Fam}(\mathcal{CG})$. The functor ϕ extends naturally to a *monad morphism* $\{\phi\}$ which sends the lifting monad on $\mathsf{Fam}(\mathcal{WB})$, used by Abramsky and McCusker [3] to model call-by-value PCF, to a monad of linear continuations on $\mathsf{Fam}(\mathcal{CG}_!)$. We use this result to establish that linear CPS translation from call-by-value PCF to λ_{Aff} corresponds to the action of ϕ by showing that for all PCF terms M, $[\![M]\!]_{\mathsf{Fam}(\mathcal{WB})} = [\![\overline{M}]\!]_{\mathsf{Fam}(\mathcal{CG}_!)}$—i.e. the following diagram commutes:



The key property distinguishing linear CPS translation from its non-linear counterpart is that it does not introduce "junk" into the interpretation of functional languages such as PCF. However, formalizing and proving this satisfactorily for source languages with recursion is not entirely straightforward [9]. In the semantic world, we are able to demonstrate completeness simply by showing that the functor ϕ is *full*. Moreover, we are able to use these properties to prove syntactic completeness results such as *full abstraction* for linear CPS translation. In particular, full abstraction for the well-bracketed model [3] of call-by-value PCF implies full abstraction for the CPS translation.

In terms of game semantics, our interpretation can be seen as an analysis of the *bracketing condition* [6,14]. This is a constraint on games and strategies; intuitively it corresponds to the "stackability" of continuation parameters in linear CPS [10]. The direct games models of PCF with first-class continuations in the author's Ph.D. Thesis [16] are based simply on abandoning this condition. A connection between bracketing and linearity is evident in one direction; in a well-bracketed sequence, every question has at most one answer. Perhaps more surprising is a result proved here—that under appropriate conditions, an innocent strategy which answers each question at most once must be well-bracketed.

Both game semantics and CPS interpretation have been proposed as general settings in which a variety of imperative and control effects can be studied. For example, in [8], it is shown that several other control features, including exception-handling, `GOTO`-style jumps and coroutines can be given linear CPS translations into λ_{Aff}, and hence modelled in our category of games. These examples are mostly confined to languages without higher-order procedures, and so rather than being a complete account are suggestive of a role for linearity

in understanding these hybrid features, which combine state-like aspects with control in a complex way. Game semantics has been proposed as the basis for a semantics of such hybrid effects [18] since it allows state and control to be combined seamlessly. Other phenomena, such as call-by-name versus call-by-value have also been studied both via games [13,20] and continuations [31,30]. Thus, we conclude this paper by examining some of the issues arising when extending or adapting the game semantics of linear continuation passing with other features of programming languages, such as recursive types, call-by-name evaluation, backwards jumps and state.

1.3. Linear CPS translation

Consider Plotkin's call-by-value CPS translation [27], which may be given, naively, as follows:

- $\overline{x} = \lambda k.(k\ x)$,
- $\overline{\lambda x.M} = \lambda k.(k\ (\lambda x.\overline{M}))$,
- $\overline{M\ N} = \lambda k.(\overline{M}\ (\lambda m.\overline{N}\ (\lambda n.(m\ n)\ k)))$.

The linear typing of the interpretation is based on the observation that in each case the current continuation (k) is used precisely once. By contrast, the associated translation of **call**/**cc**:

$$\overline{\mathbf{call/cc}\ M} = \lambda k.(\overline{M}\ (\lambda m.(m\ \lambda ab.(k\ a))\ k))$$

contains multiple occurrences of the current continuation (and passes it as an argument to a non-linear term).

In order to express these properties as linear typings, we adopt as a target language for linear CPS translation, a λ-calculus over dual linear and non-linear contexts in the style of Barber's DILL [7] and essentially the same as the CPS calculi in [29,17,8]. We shall study an *affine* version as this is simpler to model and retains the key property that the affine CPS translation of call-by-value is fully abstract.

Rather than using all of the connectives of intuitionistic linear logic, we restrict ourselves to the connectives $\multimap, \Rightarrow, \&$. This results in a simple calculus with a sufficiently fine-grained-type system to capture the properties of linear CPS. It may be viewed as the negative fragment of a version of polarized linear logic [19]. [1]

The types of λ_{Aff} are generated from a basis B of primitive datatypes such as `nat`, together with a distinguished atomic type R—the "answer type" of the CPS translation—by the connectives $\Rightarrow, \multimap, \&$ (intuitionistic implication, linear implication, and linear additive product):

$$S, T ::= B \mid \mathrm{R} \mid S \multimap T \mid S \& T \mid S \Rightarrow T$$

[1] To capture the linear/non-linear aspect of λ_{Aff}, we require both exponentials and liftings (both swap polarities, but the former also allow contraction, whilst the latter do not). The types of λ_{Aff} correspond to negative types under the following representations:
$\mathrm{R} = \bot$,
$S \multimap T = \uparrow S^{\bot} \parr T$,
$S \Rightarrow T = ?S^{\bot} \parr T$,
Sequents $A_1, \ldots, A_m; B_1, \ldots B_n \vdash T$ in λ_{Aff} correspond to polarized linear logic sequents of the form $?A_1^{\bot}, \ldots, ?A_m^{\bot}, \uparrow B_1^{\bot}, \ldots, \uparrow B_n^{\bot}, T$.

Table 1
Typing judgements for λ_{Aff}

$\frac{}{\Gamma;\Delta,x{:}T\vdash x{:}T}$ (Var)	$\frac{\Gamma;\Delta,x{:}S\vdash M{:}T}{\Gamma,x{:}S;\Delta\vdash M{:}T}$ (Der)
$\frac{\Gamma;\Delta,x{:}S\vdash M{:}T}{\Gamma;\Delta\vdash\lambda x.M{:}S\multimap T}$ $(\multimap -I)$	$\frac{\Gamma;\Delta\vdash N{:}S \quad \Gamma;\Delta'\vdash M{:}S\multimap T}{\Gamma;\Delta,\Delta'\vdash M\ N{:}T}$ $(\multimap -E)$
$\frac{\Gamma,x{:}S;\Delta\vdash M{:}T}{\Gamma;\Delta\vdash\lambda x.M{:}S\Rightarrow T}$ $(\Rightarrow -I)$	$\frac{\Gamma;\_\vdash N{:}S \quad \Gamma;\Delta\vdash M{:}S\Rightarrow T}{\Gamma;\Delta\vdash M\ N{:}T}$ $(\Rightarrow -E)$
$\frac{\Gamma;\Delta\vdash M{:}S \quad \Gamma;\Delta\vdash N{:}T}{\Gamma;\Delta\vdash\langle M,N\rangle{:}S\&T}$ $(\& -I)$	$\frac{\Gamma;\Delta\vdash M{:}T_1\&T_2}{\Gamma;\Delta\vdash\pi_i(M){:}T_i}$ $(\& -E)$

Terms-in-context of λ_{Aff} (Table 1) have the form $\Gamma;\Delta\vdash M : T$—i.e. there are two 'zones' (multisets of typed variables) to the left of the turnstyle, of which the first is intuitionistic, and the second is affine. The rules for the connective $\Rightarrow$ combine the familiar rules for ! and $\multimap$—i.e. $\Rightarrow$ elimination combines promotion with $\multimap$ elimination. The basic term-language itself is just the λ-calculus with pairing (to which we may add constants for manipulating the datatypes). Unlike [8], a single, standard notation for λ-abstraction and application is used for the introduction and elimination of both $\Rightarrow$ and $\multimap$, so the type which can be assigned to a term-in-context is not unique. Following [8], we may now exhibit the linear typing of the Plotkin-style linear CPS interpretation.

Definition 1.1. The linear CPS translation of simple types of the call-by-value λ-calculus is as follows:
$\overline{B} = B$, if B is an atomic type,
$\overline{S \Rightarrow T} = \overline{S} \Rightarrow (\overline{T} \Rightarrow \mathrm{R})\multimap\mathrm{R}$.

Proposition 1.2. *If $M : T$ is a λ_v term-in-context with free variables $x_1 : S_1, \ldots, x_n : S_n$ then its CPS translation may be typed as a λ_{Aff} term-in-context $x_1 : \overline{S_1}, \ldots, x_n : \overline{S_n}; \_\vdash \overline{M} : (\overline{T} \Rightarrow \mathrm{R})\multimap\mathrm{R}$.*

2. Game semantics

We shall now present a category of games in the style of Hyland and Ong [14], and Nickau [24], which will be used to define a semantics of λ_{Aff} (over an empty set of primitive datatypes) in which every compact strategy is definable as a term of λ_{Aff} extended with a constant for divergence.

We extend HO-arenas with a notion of *coherence* in the form of a symmetric relation on moves, and a corresponding refinement of the definition of legal sequence. A second significant feature is that we drop the "Opponent visibility" condition imposed in [14]; this actually cuts down the space of innocent and coherent strategies, allowing a universality result to be proved.

First, we shall briefly describe *underlying* arenas and innocent strategies, which will form a basis for both the model of λ_{Aff} (by adding a coherence relation) and for well-bracketed models of PCF, etc. (by adding a question/answer labelling to moves). These are essentially

the standard notions of HO-style game semantics, introduced in [14] and developed in [22,16], to which we refer for more detailed explanations.

Definition 2.1. An arena A is a pair $\langle M_A, \vdash_A \rangle$. M_A is a set of *moves*, and $\vdash_A \subseteq (M_A)_* \times M_A$ (where $(M_A)_* = M_A \cup \{*\}$, for a special token $* \notin M_A$) is an *enabling* relation, which allows a unique *polarity* to be *inferred* for each move, according to the following rules:
m is an O-move if it is initial (i.e. $* \vdash a$), or is enabled by some P-move.
m is a Player (P)-move if it is enabled by some O-move.

A *justified sequence* over an arena A is a sequence of elements of M_A in which each non-initial move a comes with a pointer to a preceding, enabling move $j_s(a)$. The set L_A of legal sequences on A consists of the justified sequences which are *alternating*—Player moves follow Opponent moves and vice versa. The product and function-space constructions on arenas are standard [14,22].

Definition 2.2. For arenas A_1, A_2, define:
- $A_1 \times A_2 = \langle M_{A_1} + M_{A_2}, \{\langle\langle m, i\rangle, \langle n, i\rangle\rangle \mid m \vdash_{A_i} n\} \cup \{\langle *, \langle m, i\rangle\rangle \mid * \vdash_{A_i} m\}\rangle$,
- $A_1 \to A_2 = \langle M_{A_1} + M_{A_2}, \{\langle\langle m, i\rangle, \langle n, i\rangle\rangle \mid m \vdash_{A_i} n\} \cup \{\langle\langle m, 2\rangle, \langle n, 1\rangle\rangle \mid * \vdash_{A_2} m \wedge * \vdash_{A_1} n\} \cup \{\langle *, \langle m, 2\rangle\rangle \mid * \vdash_{A_2} m\}\rangle$.

The empty arena (which has an empty set of moves) will be written **1**.

The behaviour of an innocent strategy is determined by the Player view of the current sequence [14,22].

Definition 2.3. The Player view of a non-empty justified sequence s is a subsequence $\ulcorner s \urcorner$ of s, defined by induction as follows:
$\ulcorner sa \urcorner = \ulcorner s \urcorner a$, if a is a Player move,
$\ulcorner sa \urcorner = a$, if a is an initial Opponent move,
$\ulcorner satb \urcorner = \ulcorner s \urcorner ab$, if b is an Opponent move justified by a.
The Player view of s is *legal* if every move in $\ulcorner s \urcorner$ points to a previous move in $\ulcorner s \urcorner$ — i.e. $\ulcorner s \urcorner$ is itself a legal sequence.

Strategies on HO-games are generally represented as even-prefix-closed sets of even-length legal sequences. We will simply represent an innocent strategy as its set of views, from which we can obtain a strategy represented in the former style by taking the *view-closure*.

Definition 2.4. An innocent strategy is a set of even-length legal P-views which is non-empty, even-prefix-closed and even-branching—i.e. $sab, sac \in \sigma$ implies $b = c$.
Given a strategy $\sigma : A$, the "view closure" $VC(\sigma)$ is defined to be the least set of legal sequences on A such that $\varepsilon \in VC(\sigma)$, and if $s \in VC(\sigma)$ and $\ulcorner sab \urcorner \in \sigma$, then $sab \in VC(\sigma)$.

Composition of strategies is by taking the views of the "parallel composition plus hiding" of their view-closures.

Definition 2.5. For $\sigma : A_1 \to A_2, \tau : A_2 \to A_3$: $\sigma;\tau = \{\ulcorner t \upharpoonright A_1, A_3 \urcorner \mid t \in L_{(A_1 \to A_2)\to A_3} \wedge t \upharpoonright A_1, A_3 \in L_{A_1 \to A_3} \wedge t \upharpoonright A_1, A_2 \in VC(\sigma) \wedge t \upharpoonright A_2, A_3 \in VC(\tau)\}$.

Composition defined in this way is *associative* (this is proved using the usual arguments for showing that the composition of innocent strategies (as sets of plays) is innocent [22], and that this operation is associative). Thus we may define a category $\mathcal{G}$ in which objects are arenas, and morphisms from A to B are strategies on $A \to B$. On each arena A, the identity $\mathsf{id}_A : A \to A$ is the set of views of the "copycat sequences" $s \in L_{A^- \to A^+}$ such that for all $t \sqsubseteq^{even} s, t \upharpoonright A^- = t \upharpoonright A^+$ (and every initial move in A^- is justified by the preceding move in A^+).

Proposition 2.6. $(\mathcal{G}, \mathbf{1}, \times)$ *is cartesian closed.*

Proof. This follows the constructions in [14,22]. Given strategies $\sigma_1 : A \to B_1$ and $\sigma_2 : A \to B_2$, we define $\langle \sigma_1, \sigma_2 \rangle : A \to B_1 \times B_2 = \{\mathtt{in}_i(s) \mid s \in \sigma_i, i \in \{1,2\}\}$ where $\mathtt{in}_i : L_{A \to B_i} \to L_{A \to B_1 \times B_2}$ is the obvious injection on coherent sequences, derived from their moves. Projection is a simple copycat strategy.

There are isomorphisms of arenas between $(A \times B) \to C$ and $A \to B \to C$ for all A, B, C with which we prove cartesian closure. □

Note that any cartesian closed category such as $\mathcal{G}$ yields a degenerate model of λ_{Aff} in which $\multimap = \Rightarrow$.

2.1. Coherence arenas

We now describe the addition of a simple coherence relation to arenas, which will allow us to give a non-degenerate interpretation of λ_{Aff}. This "linear decomposition" of HO games may be compared with that given by McCusker in his thesis [22]. The latter involves specifying the legal plays of a compound arena by projection onto the component arenas. This account prefigures many of the same issues which are encountered here (such as well-openedness) and does indeed yield a model of λ_{Aff}, but not a universal one. Moreover, the failure of universality can be directly ascribed to the projection condition—the example at the end of this section is intended to shed more light on this point.

Given an arena A, we say that moves $m, n \in M_A$ are *co-enabled* if $\exists l \in (M_A)_*.(l \vdash_A m) \wedge (l \vdash_A n)$.

Definition 2.7. A coherence arena (or C-arena) A is a pair $\langle |A|, \sim_A \rangle$ consisting of an underlying arena $|A|$ together with a symmetric relation [2] $\sim_A$ between co-enabled moves of $|A|$.

A coherent sequence of A is a legal sequence s of $|A|$ which satisfies the following conditions:

- All initial moves in s are coherent: if $ta, t'a' \sqsubseteq s$ and $* \vdash a, a'$ then $a \sim_A a'$.
- Any two non-initial moves in s with the same justifier are coherent: if $ta, t'a' \sqsubseteq s$ and $j_s(a) = j_s(a')$ then $a \sim_A a'$.

[2] We assume symmetry for the sake of convenience; by abandoning it we can construct a model of *ordered linear logic* as in [29].

So, in particular, a coherent sequence cannot contain repeated occurrences of self-incoherent moves with the same justifier. We can now define notions of additive and multiplicative product; both are based on the product of the underlying arenas, but they are differentiated by varying the coherence relations on initial moves. This distinction may be summarized as follows. In $A \otimes B$ every initial move of A is coherent with every initial move of B, whereas in $A\&B$ every initial move of A is incoherent with every initial move of B. Hence a coherent sequence of $A \otimes B$ consists of a pair of interleaved sequences of A and B, and a coherent sequence of $A\&B$ is a sequence wholly from A or wholly from B.

Definition 2.8. Given C-arenas A_1, A_2, define the following C-arenas:

Tensor-product: $A_1 \otimes A_2 = \langle |A_1| \times |A_2|, \sim_{A_1 \otimes A_2} \rangle$ where $\langle m, i \rangle \sim_{A_1 \otimes A_2} \langle n, j \rangle$ iff $i = j$ implies $m \sim_{A_i} n$.

Additive product: $A_1 \& A_2 = \langle |A_1| \times |A_2|, \sim_{A_1 \& A_2} \rangle$ where $\langle m, i \rangle \sim_{A_1 \& A_2} \langle n, j \rangle$ iff $i = j$ and $m \sim_{A_i} n$.

Linear function space: $A_1 \multimap A_2 = \langle |A_1| \to |A_2|, \sim_{A_1 \multimap A_2} \rangle$, where $\langle m, i \rangle \sim_{A_1 \multimap A_2} \langle n, j \rangle$ iff $i = j$ implies $m \sim_{A_i} n$.

We shall say that a strategy is *coherent* if it is never the first participant in a dialogue to violate the coherence condition.

Definition 2.9. For any C-arena A, an innocent strategy on A is coherent if whenever $sa \in VC(\sigma)$ and s is coherent then sa is coherent.

However, we cannot define a category of C-arenas in the standard fashion by taking morphisms from A to B to be coherent strategies on $A \multimap B$, as the composition of coherent strategies is not coherent in general. The problem arises when we have an arena B containing initial moves which are coherent, and an arena A containing initial moves which are incoherent. We may have a coherent strategy σ on $A \multimap B$ which plays two incoherent initial moves in A which are justified by two coherent initial moves in B, because the coherence condition applies only to moves with the same justifier. But if we compose σ with a strategy $\tau : B \to C$, which plays the same two coherent moves in B justified by a single initial move in C, the result is not coherent. To solve this problem, we place a further restriction on the coherent strategies which are permitted as morphisms.

Definition 2.10. A C-strategy from A to B is a coherent strategy on $A \multimap B$ such that if $s \in VC(\sigma)$ is coherent then $s \upharpoonright A$ is coherent.

Lemma 2.11. *Let $\sigma : A \to B$ and $\tau : B \to C$ be C-strategies. If $r \in L_{(A \multimap B) \multimap C}$ and $\exists s \in VC(\sigma), t \in VC(\tau)$ such that $r \upharpoonright A \multimap B \sqsubseteq s$ and $r \upharpoonright B \multimap C \sqsubseteq t$, then if $r \upharpoonright A \multimap C$ is coherent, $r \upharpoonright A \multimap B$ and $r \upharpoonright B \multimap C$ are both coherent.*

Proof. This is by induction on sequence length using the fact that every move in B is a Player move for either σ or τ, and hence cannot be the first move to violate coherence. □

Proposition 2.12. *The composition of innocent C-strategies is an innocent C-strategy.*

Proof. Note that for any $s \in L_{A \multimap B}$, if $s \upharpoonright A$ is coherent and $s \upharpoonright B$ is coherent then s is coherent. So to show that the composition of C-strategies $\sigma : A \to B, \tau : B \to C$ is a C-strategy it suffices to show that if $sa \in VC(\sigma; \tau)$ and s is coherent, then $sa \upharpoonright A$ and $sa \upharpoonright C$ are coherent.

By definition of composition, there exists $ta \in L_{(A \multimap B) \multimap C}$ such that $ta \upharpoonright A \multimap B \in VC(\sigma)$ and $ta \upharpoonright B \multimap C \in VC(\tau)$, and $sa = ta \upharpoonright A \multimap C$, and so $t \upharpoonright A \multimap C$ is coherent, and by Lemma 2.11, $t \upharpoonright A \multimap B$ and $t \upharpoonright B \multimap C$ are both coherent. Hence $ta \upharpoonright A \multimap B$ and $ta \upharpoonright B \multimap C$ are coherent, and so $sa \upharpoonright C = ta \upharpoonright C$ is coherent, and since σ is a C-strategy, $sa \upharpoonright A = (ta \upharpoonright A \multimap B) \upharpoonright A$ is coherent as required. □

Clearly, the identity strategy is always a C-strategy. Hence, we can define a category $\mathcal{CG}$ with C-arenas as objects and C-strategies from A to B as morphisms from A to B. Note that we have a functor $\mathcal{F} : \mathcal{CG} \to \mathcal{G}$ which acts on objects by forgetting the equivalence relation, and on strategies as the identity. We derive the action of both tensor and additive products from the product on the underlying category $\mathcal{G}$. (As a consequence of this, the tensor and additive units are the same, and hence the projections $\pi_i : A_1 \otimes A_2 \to A_i$ are C-strategies—i.e. we have an *affine* category).

Proposition 2.13. *$(\mathcal{CG}, \mathbf{1}, \otimes)$ is a symmetric monoidal category with cartesian products.*

Proof. Given C-strategies $\sigma_1 : A_1 \to B_1$ and $\sigma_2 : A_2 \to B_2$, the strategy $\sigma_1 \otimes \sigma_2 = \sigma_1 \times \sigma_2$ is a well-defined C-strategy from $A_1 \otimes A_2$ to $B_1 \otimes B_2$. The coherence isomorphisms for $\times$ in $\mathcal{G}$ are all C-strategies.

Similarly, the definitions of pairing and projection for the cartesian product & are given by pairing and projection in $\mathcal{G}$. □

The price we have paid for adopting C-strategies as morphisms is the loss of the symmetric monoidal *closed* structure—$\mathcal{CG}$ does not have all exponentials in the following sense.

Definition 2.14. Let A, B be objects in a SMC $(\mathcal{C}, I, \otimes)$. An exponential of B by A is an object B^A such that for all C in $\mathcal{C}$, there is an isomorphism: $\Lambda : \mathcal{C}(C \otimes A, B) \cong \mathcal{C}(C, B^A)$ which is natural in C.

We have an isomorphism of arenas—$(A \otimes B) \multimap C \cong A \multimap (B \multimap C)$—but not, in general $\mathcal{CG}(A \otimes B, C) \cong \mathcal{CG}(A, B \multimap C)$. For example, there is an obvious copycat coherent strategy on $A \multimap (A \otimes A)$ which is not a C-strategy from A to $A \otimes A$.

However, to model λ_{Aff}, we do not require that B^A exists for any arena B, but only for B within a specified collection of *well-opened* arenas, for which the notions of coherent strategy on $A \multimap B$ and C-strategy from A to B coincide.

Definition 2.15. Say that a C-arena A is *well-opened* if for any pair of initial moves $m, n \in M_A$ (not necessarily distinct), $m \not\asymp_A n$. We shall write $\mathcal{WO}$ for the full subcategory of $\mathcal{CG}$ consisting of well-opened C-arenas.

The proof of the following lemma is direct from the definitions.

Lemma 2.16. *If B is well-opened, then for any A, $A \multimap B$ is well-opened, and if A and B are well opened, then $A \& B$ is well-opened.*

Lemma 2.17. *If B is well-opened, then for any $\mathcal{C}$-arena A, $A \multimap B$ is an exponential of B by A. Moreover, the exponential structure of $\mathcal{CG}$ is preserved by the functor $\mathcal{F}$ into the cartesian closed structure of $\mathcal{G}$.*

Proof. If D is any C-arena, then *every* coherent strategy σ on $D \multimap B$ is a C-strategy from D to B, since by well-openedness of B, every coherent sequence on $D \multimap B$ contains at most one initial move, and so any two initial moves of D in s must have the same justifier, and hence be coherent. So for any arena C, $\mathcal{CG}(C \otimes A, B) \cong \mathcal{CG}(\mathbf{1}, (C \otimes A) \multimap B) \cong \mathcal{CG}(\mathbf{1}, C \multimap (A \multimap B)) \cong \mathcal{CG}(C, A \multimap B)$ as required. □

We can now use coherence to define a simple monoidal co-monad $! : \mathcal{CG} \to \mathcal{CG}$. The C-arena $!A$ has the same underlying arena as A, but all of the "incoherences" between the initial moves of A are removed.

Definition 2.18. For any C-arena A, define $!A = \langle |A|, \sim_{!A} \rangle$, where $m \sim_{!A} n$, if $m \sim_A n$ or $* \vdash m, n$.

Thus for a well-opened arena A, the coherent sequences of $!A$ consist of multiple interleaved sequences ("threads") of A. The action of $!$ on morphisms is simple: if $\sigma : !A \to B$ is a coherent strategy, then $\sigma^\dagger : !A \to !B$ has the same underlying strategy. For each C-arena A we have $\mathsf{der}_A : !A \to A$ which has the same underlying strategy as the identity. We also have equalities of arenas: $!(A \& B) = !A \otimes !B = !(A \otimes B)$, and thus morphisms $\mathsf{con}_A : !A \to !A \otimes !A$ for each A. The operation $(\_)^\dagger$ extends to send $f : !A_1 \otimes \ldots \otimes !A_n \to B$ to $f^\dagger : !A_1 \otimes \ldots \otimes !A_n \to !B$.

We have defined functors $\_ \& \_ : \mathcal{WO} \times \mathcal{WO} \to \mathcal{WO}$, $\_ \multimap \_ : \mathcal{WO}^{OP} \times \mathcal{WO} \to \mathcal{WO}$ and $\_ \Rightarrow \_ : \mathcal{WO}^{OP} \times \mathcal{WO} \to \mathcal{WO} = !\_ \multimap \_$ with which we interpret types in λ_{Aff}, having fixed an interpretation for atomic types. Terms in context $S_1, \ldots, S_m; T_1, \ldots, T_n \vdash M : U$ are interpreted as morphisms from $![\![S_1]\!] \otimes \ldots ![\![S_m]\!] \otimes [\![T_1]\!] \ldots \otimes [\![T_n]\!]$ to $[\![U]\!]$ according to Table 2. [3]

Definition 2.19. The equational theory of λ_{Aff}, $=_{\lambda_{\mathrm{Aff}}}$ is generated by the rules:

$$
\begin{array}{rl}
(\beta) & (\lambda x.M)\, N =_{\lambda_{\mathrm{Aff}}} M[N/x], \\
(\eta) & \lambda x.(M\, x) =_{\lambda_{\mathrm{Aff}}} M,\ x \notin FV(t), \\
(\pi) & \pi_i(\langle M_1, M_2 \rangle) =_{\lambda_{\mathrm{Aff}}} M_i,\ i = 1, 2, \\
(\pi_\eta) & \langle \pi_1(M), \pi_2(M) \rangle =_{\lambda_{\mathrm{Aff}}} M.
\end{array}
$$

[3] Or, more precisely, as $S_1, \ldots, S_m$ and $T_1, \ldots, T_n$ are *sequentializations* of multiset contexts, we interpret $\Gamma; \Delta \vdash U$ as a family containing a morphism from $![\![S_1]\!] \otimes \ldots ![\![S_m]\!] \otimes [\![T_1]\!] \ldots \otimes [\![T_n]\!]$ to $[\![U]\!]$ for each *sequentialization* of Γ as $S_1, \ldots, S_m$ and Δ as $T_1, \ldots, T_n$, closed under the coherence isomorphisms for $\otimes$, so that any one morphism determines the rest.

Table 2
Interpretation of λ_{Aff} terms in context

$[\![\Gamma; \Delta, x : T \vdash x : T]\!] = \pi_r$	(Var)
$[\![\Gamma, x : S; \Delta \vdash M : T]\!] = (\mathsf{id}_{[\![\Gamma]\!]} \otimes \mathsf{der}_{[\![T]\!]} \otimes \mathsf{id}_{[\![\Delta]\!]}); [\![\Gamma; x : S, \Delta \vdash M : T]\!]$	(Der)
$[\![\Gamma; \Delta \vdash \lambda x.M : S \multimap T]\!] = \Lambda_{[\![S]\!]}([\![\Gamma; \Delta, x : S \vdash M : T]\!])$	$(\multimap - I)$
$[\![\Gamma; \Delta, \Delta' \vdash M\ N : T]\!] = (\mathsf{con}_{[\![\Gamma]\!]} \otimes \mathsf{id}_{[\![\Delta,\Delta']\!]}); [\![\Gamma; \Delta \vdash M : S \multimap T]\!] \otimes [\![\Gamma; \Delta' \vdash N : S]\!]; \mathsf{App}$	$(\multimap - E)$
$[\![\Gamma; \Delta \vdash \lambda x.M : S \Rightarrow T]\!] = \Lambda_{[\![S]\!]}([\![\Gamma, x : S; \Delta \vdash M : T]\!])$	$(\Rightarrow - I)$
$[\![\Gamma; \Delta \vdash M\ N : T]\!] = (\mathsf{con}_{[\![\Gamma]\!]} \otimes \mathsf{id}_{[\![\Delta]\!]}); ([\![\Gamma; \Delta \vdash M : S \Rightarrow T]\!] \otimes [\![\Gamma; \_ \vdash N : S]\!]^{\dagger}); \mathsf{App}$	$(\Rightarrow - E)$
$[\![\Gamma; \Delta \vdash \langle M, N\rangle : S \& T]\!] = \langle [\![\Gamma; \Delta \vdash M : S]\!], [\![\Gamma; \Delta \vdash N : T]\!]\rangle$	$(\& - I)$
$[\![\Gamma; \Delta \vdash \pi_i(M) : T_i]\!] = [\![\Gamma; \Delta \vdash M : T_1 \& T_2]\!]; \pi_i$	$(\& - E)$

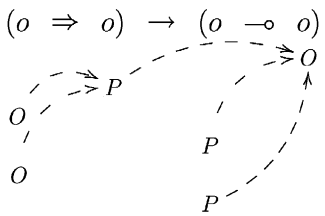


Fig. 1. Excluding the copycat $(o \Rightarrow o) \to (o \multimap o)$ by relaxing O-visibility.

To establish soundness of the interpretation with respect to this theory, we simply use our observation that the forgetful functor $\mathcal{F} : \mathcal{CG} \to \mathcal{G}$ preserves the structure of (both) products, and of exponentials.

Proposition 2.20. *For all terms-in-context,* $\mathcal{F}([\![\Gamma; \Delta \vdash M : T]\!]_{\mathcal{CG}}) = [\![\Gamma; \Delta \vdash M : T]\!]_{\mathcal{G}}$.

Corollary 2.21. *The model of* λ_{Aff} *in* $\mathcal{CG}$ *is sound with respect to* λ_{Aff}*-equivalence—i.e.* $M =_{\lambda_{\mathrm{Aff}}} N$ *implies* $[\![M]\!] = [\![N]\!]$.

Our semantics of linear CPS will be based on a specific interpretation of R as the smallest, non-empty well-opened C-arena.

Definition 2.22. Let o be the arena with one move which is not coherent with itself, i.e. $M_o = \{o\}$, $\vdash_o = \langle *, o\rangle$ and $o \not\asymp_o o$.

We may now present a brief example, which shows why Opponent is allowed to violate the visibility condition [14] in our model. Fig. 1 shows a play in which Player attempts to play copycat between the arenas $o \Rightarrow o$ and $o \multimap o$, which should not be possible. If we imposed a condition requiring the projection of play into each component game to be a valid play (which is a consequence of visibility) then Opponent would have no response to Player's second move, and the copycat strategy would be coherent. By violating projection/visibility, however, Opponent can repeat his second move, forcing Player to repeat his second move, and this violates coherence.

3. Universality for λ_{Aff}

We will now prove that our model of λ_{Aff} (with a single constant $\perp : \mathrm{R}$ for divergence,[4] interpreted as the empty strategy) is *universal* for all types generated from the atomic type R (i.e. we take $B = \emptyset$). That is, for every innocent C-strategy $\sigma : !\llbracket A_1 \rrbracket \otimes \ldots !\llbracket A_m \rrbracket \otimes \llbracket B_1 \rrbracket \otimes \ldots \llbracket B_n \rrbracket \to \llbracket T \rrbracket$ which is finite (as a set of views) there exists a λ_{Aff}-term-in-context $A_1, \ldots, A_m; B_1, \ldots, B_n \vdash M_\sigma : T$ such that $\sigma = \llbracket M_\sigma \rrbracket$. We will prove this by induction on $\#(\sigma)$ (the cardinality of σ as a set of views), using a decomposition argument similar to those used for PCF [6,14], and described axiomatically by Abramsky in [1]. This requires a series of "decomposition lemmas" stating that certain canonical operations in $\mathcal{CG}$ (for which we sometimes use λ_{Aff} notation as shorthand) have inverses. Axioms which have appeared in [1] (or their equivalents) are labelled as they appear there, and their proofs are omitted.

Lemma 3.1. *Given a finite innocent strategy $\sigma : (A_1 \multimap \ldots \multimap A_n \multimap o) \multimap o$, there exist strategies $\tau_1, \ldots, \tau_n$ such that $\#(\tau_i) < \#(\sigma)$ for each $i \leqslant n$, and $\sigma = \lambda \mathsf{x}.(((\mathsf{x}\ \tau_1) \ldots)\ \tau_n)$.*

Proof. By innocence, for each i, $\tau_i = \{s \upharpoonright A_i \mid s \in \sigma\}$ is a well-defined innocent strategy which satisfies the above requirements. □

Lemma 3.2 (*Bang lemma*). *For every strategy $\sigma : !B \to !A$, $\sigma = (\sigma; \mathsf{der}_A)^\dagger$.*

Definition 3.3. A strategy in $\mathcal{CG}(A, B)$ is *strict* if $\perp; \sigma = \perp$ (i.e. the first P-move in σ, if any, is in A). $\sigma \in \mathcal{CG}(A \otimes B, C)$ is *strict in A* if $\Lambda(\sigma) \in \mathcal{CG}(A, B \multimap C)$ is strict.

Note that *every* strategy in $\mathcal{CG}(A, o)$ is strict. The following lemma follows directly from determinacy of strategies.

Lemma 3.4. *If $\sigma \in \mathcal{CG}(A_1 \otimes \ldots \otimes A_n, o)$ is strict, then there is a unique i such that σ is strict in A_i.*

Lemma 3.5 (*Linearization of head occurrence*). *If $\sigma : !A \to B$ is strict, then there exists a strategy $\tau : A \otimes !A \to B$ which is strict in A, and such that $\#(\tau) = \#(\sigma)$ and $\sigma = \mathsf{con}_A; (\mathsf{der}_A \otimes \mathsf{id}_A); \tau$.*

Lemma 3.6. *Given a strict strategy $\sigma : (A_1 \multimap \ldots \multimap A_n \multimap o) \to (B \Rightarrow C)$, there exists a strict strategy $\tau : ((B \Rightarrow A_1) \multimap \ldots \to (B \Rightarrow A_n) \multimap o) \multimap C$ such that $\#(\tau) = \#(\sigma)$ and $\sigma(\mathsf{f}) = \lambda \mathsf{x}.\tau\,(\lambda \mathsf{y}_1 \ldots \mathsf{y}_n.\mathsf{f}\,(\mathsf{y}_1\,\mathsf{x}) \ldots (\mathsf{y}_n\,\mathsf{x}))$.*

Proof. By strictness, every non-empty view in σ contains the initial move in $A_1 \multimap \ldots \multimap A_n \multimap o$. Thus every view terminating in the initial move b in B contains an initial move a_i from precisely one of the A_i. Hence a unique τ can be derived from σ by adding a pointer from b to a_i. □

[4] We have a universal model of the pure calculus consisting of strategies which are *total* in the sense of Abramsky [1].

Definition 3.7. An arena is *π-atomic* if it is well-opened and has a unique initial move.

Lemma 3.8. *Given a strict* $\sigma : (A_1 \multimap \ldots \multimap A_n \multimap o) \to (B \multimap C)$, *where* C *is* π*-atomic, there exists* $1 \leqslant i \leqslant n$ *and a strict strategy* $\tau : (A_1 \multimap \ldots \multimap (B \multimap A_i) \multimap \ldots \multimap A_n) \multimap o) \to C$ *such that* $\#(\sigma) = \#(\tau)$ *and* $\sigma(\mathsf{f}) = \lambda \mathsf{x}.\tau\,(\lambda \mathsf{y}_1 \ldots \mathsf{y}_n.\mathsf{f}\,\mathsf{y}_1 \ldots (\mathsf{y}_i\,\mathsf{x}) \ldots \mathsf{y}_n)$.

Proof. This is as for Lemma 3.6, but in this case we note that b can occur as the final move in at most one P-view in σ (otherwise, Opponent can force Player to violate coherence). Hence b is associated with a unique a_i. □

Lemma 3.9 (π*-atomicity*). *If* C *is* π*-atomic and* $\sigma : A_1 \& A_2 \to C$ *is strict, then there exists* $i \in 1, 2$ *and* (*strict*) $\tau : A_i \to C$ *such that* $\sigma = \pi_i ; \tau$.

We define a corresponding notion of *π-atomic type* by the following grammar:
$P ::= \mathrm{R} \mid T \multimap P \mid T \Rightarrow P$
We shall write these types in the form $S_1 \Rightarrow \ldots \Rightarrow S_m \Rightarrow T_1 \multimap \ldots \multimap (T_n \multimap \mathrm{R})$, or $\vec{S} \Rightarrow \vec{T} \multimap \mathrm{R}$.

Proposition 3.10. *Every finite, innocent* C*-strategy* $\sigma \in [\![\Gamma ; \Delta \vdash T]\!]$ *is definable — i.e. there exists a* $\lambda_{\mathrm{Aff}}(\bot)$ *term-in-context* $\Gamma ; \Delta \vdash M_\sigma : T$ *such that* $\sigma = [\![M_\sigma]\!]$.

Proof. By induction on $\#(\sigma)$. If $\#(\sigma) = 0$ then σ is the empty strategy, which is definable for all types from $\bot : \mathrm{R}$. We prove the inductive case by analysis of a series of successively more general cases determined by the form of Γ, Δ and T.

(i) Suppose $T = \mathrm{R}$, $\Gamma = \{\}$ and $\Delta = \{\vec{A} \Rightarrow \vec{B} \multimap \mathrm{R}\}$. Then by Lemmas 3.1 and 3.2, and the induction hypothesis on $\#(\sigma)$, there are closed terms $\vec{M} : \vec{A}$ and $\vec{N} : \vec{B}$ such that $\sigma = [\![x : \vec{A} \Rightarrow \vec{B} \multimap \mathrm{R} \vdash ((x\ \vec{M})\ \vec{N}) : \mathrm{R}]\!]$.

(ii) Suppose T is π-atomic, $\Gamma = \{\}$, $\Delta = \{\vec{A} \Rightarrow \vec{B} \multimap \mathrm{R}\}$, and σ is strict.
We prove definability of σ by induction on T, for which the base case is (i). The induction cases are as follows:
- If $T = C \Rightarrow P$, then applying Lemma 3.6, we obtain a strategy τ which (by induction hypothesis on P) is definable as a term $\_ ; x : (C \Rightarrow \vec{A}) \Rightarrow (C \Rightarrow \vec{B}) \multimap \mathrm{R} \vdash M : P$ such that:
$\sigma = [\![y : \vec{A} \Rightarrow \vec{B} \multimap \mathrm{R} \vdash \lambda z.M[(\lambda \vec{u}\vec{v}.(y\ (\vec{u\ z})\ (\vec{v\ z})))/x] : C \Rightarrow P]\!]$.
- If $P_2 = C \multimap P$ then by Lemma 3.8 and induction on P, there exists $i \leqslant m$ and $x : \vec{A} \Rightarrow \vec{B'} \multimap \mathrm{R} \vdash M : P$ (where $B'_i = C \multimap B_i$ and $B'_j = B_j$ for $j \neq i$) such that $\sigma = [\![\_ ; y : \vec{A} \Rightarrow \vec{B} \multimap \mathrm{R} \vdash \lambda z.M[(\lambda \vec{u}.\lambda \vec{v}.y\,\vec{u}\,v_1 \ldots v_{i-1}(v_i\ z)v_{i+1} \ldots v_m)/x] : C \multimap P]\!]$.

(iii) Suppose T is π-atomic, $\Gamma = \{\}$, $\Delta = \{S\}$ and σ is strict.
This case is proved by induction on S. If S is π-atomic, then this is an instance of case (ii). Otherwise $S = \vec{A} \Rightarrow \vec{B} \multimap C_1 \& C_2$ and we can find $i \in \{1, 2\}$ and a term-in-context $x : \vec{A} \Rightarrow \vec{B} \multimap C_i \vdash M : P$ such that $\sigma = [\![y : \vec{A} \Rightarrow \vec{B} \multimap C_1 \& C_2 \vdash M[\lambda \vec{u}.\lambda \vec{v}.\pi_i\,(y\ \vec{u}\,\vec{v})/x] : T]\!]$.

(iv) Suppose $T = \mathrm{R}$, $\Gamma = \{A_1, \ldots, A_m\}$ and $\Delta = \{B_1, \ldots, B_n\}$. Then by Lemma 3.4, σ is strict in some unique $!\llbracket A_i \rrbracket$ or $\llbracket B_j \rrbracket$. If the latter, we obtain by carrying a strict strategy which is definable as a term $\_; y_j : B_j \vdash M : \vec{A} \Rightarrow B_1 \multimap \ldots \multimap B_{j-1} \multimap B_{j+1} \multimap \ldots \multimap B_n \multimap \mathrm{R}$ such that $\sigma = \llbracket x_1 : A_1, \ldots, x_m : A_m; y_1 : B_1, \ldots, y_n : B_n \vdash (M\ \vec{x})\ y_1 \ldots y_{j-1} y_{j+1} \ldots y_n : \mathrm{R} \rrbracket$.

If the former, then by linearization of head occurrence, we obtain a strategy $\sigma' : !\llbracket A_1 \rrbracket \otimes \ldots \otimes !\llbracket A_m \rrbracket \otimes \llbracket A_i \rrbracket \otimes \llbracket B_1 \rrbracket \otimes \ldots \otimes \llbracket B_n \rrbracket \to \llbracket T \rrbracket$ which is strict in (the linear occurrence of) A_i, and thus definable as above.

(v) The general case. This is proved by induction on T, for which the base case is item (iv). The inductive cases are simply dealt with by uncurrying and then currying or projection and pairing. □

4. Well-bracketing and linear CPS

We will now show that a variant of the original category of HO games with questions and answers [14] (extended by McCusker with lifted sums [22]) can be fully embedded in the category of coherence games.

Definition 4.1. A bracketed arena (or B-arena) is a pair $\langle |A|, \lambda_A \rangle$ consisting of an underlying arena $|A|$ (as per Definition 2.1), with a labelling function $\lambda_A : M_A \to \{\mathsf{Q}, \mathsf{A}\}$, which partitions the set of moves into *questions* and *answers*. Answers must be enabled (and may only be enabled) by questions.

The product and function-space of bracketed arenas are based on the corresponding constructions on the underlying arenas—i.e. $A \times B = \langle |A| \times |B|, [\lambda_A, \lambda_B] \rangle$ and $A \Rightarrow B = \langle |A| \to |B|, [\lambda_A, \lambda_B] \rangle$. The *bracketing condition* states that questions and answers (considered as opening and closing parentheses, respectively) must be well nested.

Definition 4.2. For each justified sequence s we define a prefix $\mathsf{pending}(s) \sqsubseteq s$ as follows:
$\mathsf{pending}(\varepsilon) = \varepsilon$,
$\mathsf{pending}(sq) = sq$ if q is a question,
$\mathsf{pending}(sqta) = \mathsf{pending}(s)$, if a is an answer justified by q.

The pending question of s is the final move in $\mathsf{pending}(s)$, if any.

Definition 4.3. An alternating justified sequence s on a bracketed arena is *well-bracketed* if every answer in s is justified by the pending question—i.e. if $rqta \sqsubseteq s$ and a is an answer justified by q, then $rq = \mathsf{pending}(rqt)$.
An innocent and *well-bracketed* strategy on a B-arena is an innocent strategy on the underlying arena such that whenever $sab \in VC(\sigma)$ and sa is well-bracketed then sab is well-bracketed.

The following lemma, which characterizes innocent and well-bracketed strategies in terms of their P-views is proved in [15,16].

Lemma 4.4. *An innocent strategy σ is well-bracketed if and only if for all $s \in \sigma$, s is well-bracketed.*

The composition of innocent and well-bracketed strategies (as defined in Definition 2.5) is well-bracketed [14,16] and thus we have a cartesian closed category $(\mathcal{WB}, \mathbf{1}, \times)$ with B-arenas as objects and innocent well-bracketed strategies on $A \Rightarrow B$ as morphisms from A to B [14,16].

Intuitively, one may think of the bracketing condition as corresponding to a "stack discipline" for moves. Asking a question pushes it onto the stack of open questions, and answering it pops the stack. The bracketing condition requires that only the top of the stack can be popped. It corresponds to the "stackability" of linear continuations [10], as the next section will show.

4.1. From well-bracketed games to coherence games

We now define a functor ϕ from $\mathcal{WB}$ to $\mathcal{CG}_!$ (the co-Kleisli category of ! on $\mathcal{CG}$). In fact, if we abuse notation a little by writing $\mathcal{WO}_!$ for the full subcategory of $\mathcal{CG}_!$ which has well-opened arenas as objects, then we may regard ϕ is a functor into $\mathcal{WO}_!$. We use the question/answer labelling on well-bracketed arenas to determine a coherence relation—(co-enabled) moves are incoherent if they are both initial or both answers to the same question, and coherent otherwise.

Definition 4.5. For each B-arena A we define a (well-opened) C-arena $\phi(A)$ as follows: $\phi(A) = \langle |A|, \sim_A \rangle$, where $m \not\sim_A n$ if $\lambda_A(m) = \lambda_A(n) = \mathsf{A}$ or $* \vdash m$ and $* \vdash n$, and $m \sim_A n$ otherwise.

So ϕ sends the B-arena $A \Rightarrow B$ to $\phi(A) \Rightarrow \phi(B)$, and thus we may define the action of ϕ on morphisms $\sigma : A \to B$ as the identity on the underlying innocent strategies. To show that this operation is well-defined, we need to show that any innocent and well-bracketed strategy on A is a coherent strategy on $\phi(A)$.

Lemma 4.6. *If sqt is a coherent sequence on $\phi(A)$ and q is a question such that* pending$(sqt) =$ pending(s)*, then q is answered in t.*

Proof. This is by induction on the length of sqt. In $sqtb$, *either* b is a question—in which case pending$(sqtb) \neq$ pending(s)—*or* b is an answer to a question q' in t—i.e. $t = t'q't''b$, where pending$(sqt') =$ pending(s) and hence q is answered in t' by induction hypothesis—or b is an answer to a question q' in s—i.e. $s = s'q't'$, and pending$(s'q't') =$ pending$(sqtb) =$ pending(s') and hence q' is already answered in t' which contradicts coherence of $sqtb$. □

Proposition 4.7. *For any B-arena A, if σ is a well-bracketed innocent strategy on A then it is a coherent strategy on $\phi(A)$.*

Proof. It is sufficient to show that if $sbc \in VC(\sigma)$ and sb is coherent, then the question pending(sb) has not already been answered and hence sbc is coherent. We show this by

induction on sequence length. Given an (odd-length) sequence sb, either b is a question—in which case it is the pending question, and unanswered—or b is an answer, and $sb = s'qtb$, where b answers q and so $\mathsf{pending}(sb) = \mathsf{pending}(s')$. The pending question in s' is not answered in s' (by inductive hypothesis) and so the only remaining possibility is that it is answered in t. In other words there is a prefix $s'qt'a \sqsubseteq s$, in which a answers $\mathsf{pending}(sb) = \mathsf{pending}(s')$. But by assumption we have $\mathsf{pending}(s'qt') = \mathsf{pending}(s')$ and hence by Lemma 4.6 q is answered in t', which contradicts coherence of sb. □

Functoriality and faithfulness are immediate by definition of ϕ, so it remains to show *fullness*. It is clearly not the case in general that if s is a coherent sequence on $\phi(A)$ then s is well-bracketed; the coherence condition prevents repetition of answers, but not answering a question prematurely. However, we can show that if σ is an innocent and coherent strategy on $\phi(A)$, then σ must be well-bracketed, as otherwise Opponent can force σ to violate coherence because of innocence, in the same way as depicted in Fig. 1. (Note that $o{-\circ}o$ is the image under ϕ of the B-arena with a single question and answer.)

Lemma 4.8. *For any C-strategy σ, if $smt \in \sigma$, where m is an O-move such that $m \sim m$, then for any (odd-length) $t' \sqsubseteq t$, there is a sequence of the form $smtmt' \in VC(\sigma)$.*

Proof. This is by induction on the length of t'. Suppose $t'b \subseteq t$ is odd-length. Then either $t' = \varepsilon$, or $t' = t''a$, where a is justified by the last move in t'' (as smt is a P-view). Hence $smtmt'$ is coherent, and $\ulcorner smtmt' \urcorner \sqsubseteq \ulcorner smt \urcorner$ and so $smtmt'b \in VC(\sigma)$. □

Proposition 4.9. *For any C-strategy $\sigma : \phi(A) \to \phi(B)$, σ is a well-bracketed strategy on $A \Rightarrow B$.*

Proof. By Lemma 4.4 it suffices to show that every $s \in \sigma$ is well-bracketed. Suppose for a contradiction that we have a minimal length sequence $sqta \in \sigma$ such that q is the pending question in sqt but a is not justified by q. By minimality of $sqta$, all of the questions in t have been answered, and so if a is an answer to any of them, it violates coherence. So suppose a is an answer to a question in s. Then by Lemma 4.8, we have a sequence of the form $sqtaqta$ in $VC(\sigma)$, where the justifier for each a is the same, and which therefore violates the coherence condition. □

Theorem 4.10. *The cartesian closed category $\mathcal{WB}$ of bracketed arenas and innocent, well-bracketed strategies embeds fully in the cartesian closed category $\mathcal{WC}_!$ of C-arenas and C-strategies.*

5. Relating well-bracketed and CPS semantics

We will now show how the functor ϕ can be used to relate models of programming languages (such as call-by-value PCF) based on well-bracketed games, to their linear CPS semantics. In order to do this we need to extend our interpretation of λ_{Aff} to include sum types, and in particular, primitive datatypes such as `nat`. To do this we will follow the

methodology used by Abramsky and McCusker in [3] to construct a category with coproducts from HO games (and hence to interpret $\mathrm{PCF_v}$). We define a category of indexed families of coherence games $\mathsf{Fam}(\mathcal{CG})$ corresponding to the completion of $\mathcal{CG}$ with countable coproducts. An alternative approach would be to model sum types more directly, as in the call-by-value game semantics of Honda and Yoshida [13], for example, or Laurent's model of polarized linear logic [20].

We will show that ϕ extends to a functor from $\mathsf{Fam}(\mathcal{WB})$ to $\mathsf{Fam}(\mathcal{CG}_!)$ which acts as a monad morphism sending the "lifted sum monad" on $\mathsf{Fam}(\mathcal{WB})$ to the "monad of linear continuations" $(\_ \Rightarrow \{o\})\multimap\{o\}$ on $\mathsf{Fam}(\mathcal{CG}_!)$. This result is then used to show that a games interpretation of call-by-value PCF via linear CPS translation is equivalent to the image under ϕ of Abramsky and McCusker's well-bracketed model.

Definition 5.1. For a category $\mathcal{C}$, let $\mathsf{Fam}(\mathcal{C})$ be the category of *set-indexed families* of objects of $\mathcal{C}$, which has as morphisms from $\{A_i \mid i \in I\}$ to $\{B_j \mid j \in J\}$, a pair $\langle f : I \to J, \{\psi_i : A_i \to B_{f(i)} \mid i \in I\}\rangle$, consisting of a re-indexing function and a family of morphisms in $\mathcal{C}$.

$\mathsf{Fam}(\mathcal{C})$ has all small co-products, given by the disjoint union of indexed families. It also inherits much of the structure of $\mathcal{C}$. For example, the symmetric monoid $\otimes$ on $\mathcal{CG}$ gives rise to a symmetric monoid on $\mathsf{Fam}(\mathcal{CG})$ (with unit $\{\mathbf{1}\}$) by setting $\{A_i \mid i \in I\} \otimes \{B_j \mid j \in j\} = \{A_i \otimes B_j \mid \langle i, j\rangle \in I \times J\}$. The cartesian product & on $\mathcal{CG}$ yields a cartesian product on $\mathsf{Fam}(\mathcal{CG})$ in the same way. As in $\mathcal{CG}$, we do not have symmetric monoidal closure, but exponentials by all well-opened objects (families of well-opened arenas)—we define $\{A_i \mid i \in I\}\multimap\{B_j \mid j \in J\} = \{\Pi_{i\in I}(A_i\multimap B_{f(i)}) \mid f \in J^I\}$. Similarly, the co-monad $! : \mathcal{CG} \to \mathcal{CG}$ extends to a co-monad on $\mathsf{Fam}(\mathcal{CG})$ by setting $!\{A_i \mid i \in I\} = \{!A_i \mid i \in I\}$. (Note that the categories $\mathsf{Fam}(\mathcal{CG}_!)$ and $\mathsf{Fam}(\mathcal{CG})_!$ are equivalent.)

Proposition 5.2. $\mathsf{Fam}(\mathcal{CG})$ *contains a sound model of* λ_{Aff} *into which the model in* $\mathcal{CG}$ *embeds fully and faithfully.*

Corresponding to linear CPS interpretation, we have a *strong monad of linear continuations*, which is simply a linear version of the standard continuations monad, and which we may express using λ_{Aff} notation.

Definition 5.3. The monad of linear continuations on $\mathsf{Fam}(\mathcal{CG}_!)$ is defined by the Kleisli triple $((\_ \Rightarrow \{o\})\multimap\{o\}, \eta, (\_)^*)$, where $\mathsf{f}^*(\mathsf{a}) = \lambda\mathsf{x}.\mathsf{a}(\lambda\mathsf{y}.\mathsf{x}\,\mathsf{f}(\mathsf{y}))$ and $\eta(\mathsf{a}) = \lambda\mathsf{x}.(\mathsf{x}\,\mathsf{a})$. The strength $t_{A,B} : A \times ((B \Rightarrow \{o\})\multimap\{o\}) \to ((A \times B) \Rightarrow \{o\})\multimap\{o\}$ is defined $\mathsf{t}(\mathsf{a}) = \lambda\mathsf{x}.(\pi_2(\mathsf{a})\,(\lambda\mathsf{y}.\mathsf{x}\,\langle\pi_1(\mathsf{a}), \mathsf{y}\rangle))$.

Thus, we have a model of the computational λ-calculus [23] corresponding to linear CPS translation into λ_{Aff} followed by interpretation in $\mathsf{Fam}(\mathcal{CG})$.

To construct a model of the computational λ-calculus in $\mathsf{Fam}(\mathcal{WB})$, we follow [3] in defining a lifting monad based on the lifted sum [22]. The latter takes a family of games $\{A_i \mid i \in I\}$ and adds an initial question with an answer a_i for each $i \in I$ which enables the initial moves of A_i.

Definition 5.4. For an indexed family of B-arenas $A = \{A_i \mid i \in I\}$, the lifted sum $\Sigma_{i\in I} A_i$ is defined as follows:

- $M_{\Sigma_{i\in I} A_i} = \{o\} + \coprod_{i\in I}(o + M_{A_i})$,
- $\vdash_{\Sigma_{i\in I} A_i} = \{\langle *, \mathtt{inl}(o)\rangle\} \cup \{\langle \mathtt{inl}(o), \mathtt{inr}(\mathtt{in}_i(\mathtt{inl}(o)))\rangle \mid i \in I\}$
  $\cup \{\langle \mathtt{inr}(\mathtt{in}_i(\mathtt{inl}(o))), \mathtt{inr}(\mathtt{in}_i(\mathtt{inr}(m)))\rangle \mid * \vdash_{A_i} m\}$
  $\cup \{\langle \mathtt{inr}(\mathtt{in}_i(\mathtt{inr}(m))), \mathtt{inr}(\mathtt{in}_i(\mathtt{inr}(n)))\rangle \mid m \vdash_{A_i} n\}$,
- $\lambda_{\Sigma_{i\in I} A_i}(\mathtt{inl}(o)) = \mathsf{Q}$, $\lambda_{\Sigma_{i\in I} A_i}(\mathtt{inr}(\mathtt{in}_i(\mathtt{inl}(o)))) = \mathsf{A}$,
  $\lambda_{\Sigma_{i\in I} A_i}(\mathtt{inr}(\mathtt{in}_i(\mathtt{inr}(m)))) = \lambda_{A_i}(m)$.

For example, the "flat" sum used to represent the type of natural numbers in the HO model of (call-by-name) PCF is equivalent to $\Sigma_{i\in\mathbb{N}}\mathbf{1}$. As shown in [22], the lifted sum is a weak co-product on $\mathcal{WB}$—there are morphisms $\mathtt{in}_i : A_i \to \Sigma_{i\in I} A_i$ for each i, an operation taking morphisms $\{f_i : A_i \to B \mid i \in I\}$ to $[f_i \mid i \in I] : \Sigma_{i\in I} \to B$ such that $\mathtt{in}_i; [f_i \mid i \in I] = f_i$, $[\mathtt{in}_i \mid i \in I] = \mathsf{id}$ and $[f_i \mid i \in I]; [g_j \mid j \in J] = [[f_i \mid i \in I]; g_j \mid j \in J]$. It is also distributive; we have a natural transformation $\mathrm{dist} : B \times \Sigma_{i\in I} A_i \to \Sigma_{i\in I}(B \times A_i)$ satisfying further naturality conditions [22].

Lemma 5.5. *The lifted sum $\Sigma A = \{\Sigma_{i\in I} A_i\}$ acts as a strong monad on* $\mathsf{Fam}(\mathcal{WB})$.

Proof. The monad has Kleisli triple $(\Sigma\_, [\_], \{\mathtt{in}_i \mid i \in I\})$ and monadic strength $\mathrm{dist} : A \times \Sigma B \to \Sigma(A \times B)$ deriving from the weak, distributive co-product structure of Σ. □

$\mathsf{Fam}(\mathcal{WB})$ has products, and exponentials by all pointed objects, and thus is a model of the computational λ-calculus.

ϕ extends to a functor $\{\phi\} : \mathsf{Fam}(\mathcal{WB}) \to \mathsf{Fam}(\mathcal{CG}_!)$ in the obvious way, and preserves the structure of $\mathsf{Fam}(\mathcal{WB})$ as a model of the computational λ-calculus. The key point is that it sends the lifting monad on $\mathsf{Fam}(\mathcal{WB})$ to the linear continuations monad on $\mathsf{Fam}(\mathcal{CG})$.[5]

Lemma 5.6. *For any family of B-arenas, $\{A_i \mid i \in I\}$:*
$\phi(\Sigma_{i\in I} A_i) \cong (\Pi_{i\in I}(\phi(A) \Rightarrow o))\multimap o$.

Proof. This follows directly from the definition of the lifted sum. □

Proposition 5.7. $(\{\phi\}, \mathsf{id})$ *is a* (*strong*) *monad morphism.*

Proof. For any family $\{A_i \mid i \in I\}$, $(\{A_i \mid i \in I\} \Rightarrow \{o\})\multimap\{o\} = \{(\Pi_{i\in I}(A_i \Rightarrow o))\multimap o\}$ and thus $\{\phi(\Sigma_{i\in I} A_i)\} = (\{\phi(A_i) \mid i \in I\} \Rightarrow \{o\})\multimap\{o\}$. Moreover $\phi(\eta_A) = \eta_{\phi(A)}$ for all A, $\phi(f^*) = \phi(f)^*$ for all f, and $\phi(t_{A,B}) = t_{\phi(A),\phi(B)}$ for all A, B. □

[5] This may be seen as a consequence of the fact that they are both determined by the same universal property (up to ϕ embedding): $(\_ \Rightarrow \{o\})\multimap\{o\} : \mathsf{Fam}(\mathcal{CG}_!) \to \{\mathcal{WO}_s\}$ is right adjoint to the inclusion of $\{\mathcal{WO}_s\}$ (the category of (singletons of) well-opened games and strict linear maps) in $\mathsf{Fam}(\mathcal{CG}_!)$.

Table 3
Operational Semantics of $\mathrm{PCF_v}$

$$\frac{}{\lambda x.M \Downarrow \lambda x.M} \qquad \frac{}{\mathtt{0} \Downarrow \mathtt{0}}$$

$$\frac{M \Downarrow n}{\mathtt{succ}\ M \Downarrow \mathtt{succ\ n}} \qquad \frac{M \Downarrow \mathtt{succ\ n}}{\mathtt{pred}\ M \Downarrow \mathtt{n}}$$

$$\frac{M \Downarrow \mathtt{0}}{\mathtt{IF0}\ M \Downarrow \lambda xy.x} \qquad \frac{M \Downarrow \mathtt{succ\ n}}{\mathtt{IF0}\ M \Downarrow \lambda xy.y}$$

$$\frac{M \Downarrow \lambda x.M' \quad N \Downarrow V \quad M'[V/x] \Downarrow U}{M\ N \Downarrow U} \qquad \frac{M\ \lambda x.((\mathtt{Y}\ M)\ x) \Downarrow V}{\mathtt{Y}\ M \Downarrow V}$$

5.1. Linear CPS interpretation of call-by-value PCF

As an example of our approach, we will show that Abramsky and McCusker's well-bracketed model of call-by-value PCF [3] is equivalent to a linear continuation-passing interpretation.

$\mathrm{PCF_v}$ is the simply typed λ-calculus over the ground type $\mathtt{nat}$ with constants $\mathtt{0} : \mathtt{nat}$ and $\mathtt{succ}, \mathtt{pred} : \mathtt{nat} \Rightarrow \mathtt{nat}$, $\mathtt{IF0} : \mathtt{nat} \Rightarrow T \Rightarrow T \Rightarrow T$ for each T, and $\mathtt{Y} : (T \Rightarrow T) \Rightarrow T$ for $T = S_1 \Rightarrow S_2$. A "big-step" operational semantics of $\mathrm{PCF_v}$ is given in Table 3. We define a standard notion of observational equivalence for $\mathrm{PCF_v}$ terms.

Definition 5.8. $M \simeq_P N$ if and only if for all closing $\mathrm{PCF_v}$ contexts $C[\_] : \mathtt{nat}$, $C[M] \Downarrow$ if and only if $C[N] \Downarrow$.

As a target language for linear CPS translation of $\mathrm{PCF_v}$, we use $\lambda^{\mathrm{nat}}_{\mathrm{Aff}}$—that is λ_{Aff} over a base type nat of natural numbers, extended with constants corresponding to those of PCF—$0 : \mathrm{nat}$ and $\mathrm{succ}, \mathrm{pred} : \mathrm{nat} \multimap \mathrm{nat}$, $\mathrm{IF0} : \mathrm{nat} \multimap (\mathrm{R\&R}) \multimap \mathrm{R}$ and $\mathbf{Y} : (P \Rightarrow P) \Rightarrow P$ for each *pointed* type P. The pointed types are defined as in [8], by the grammar:

$$P, Q ::= \mathrm{R} \mid T \multimap P \mid P \& Q \mid T \Rightarrow P.$$

(This corresponds to the notion of pointedness in the semantics; general types are interpreted as families of games and pointed types as singleton families.)

Definition 5.9. We extend the equational theory of λ_{Aff} with the following axioms for the constants:

$$\mathrm{succ}\ (\mathrm{pred}\ n) =^{\mathrm{nat}}_{\lambda_{\mathrm{Aff}}} n \qquad \mathrm{pred}\ (\mathrm{succ}\ n) =^{\mathrm{nat}}_{\lambda_{\mathrm{Aff}}} n,$$
$$\mathrm{IF0}\ 0 =^{\mathrm{nat}}_{\lambda_{\mathrm{Aff}}} \lambda x.\pi_1(x)\ \ \mathrm{IF0}\ (\mathrm{succ}\ n) =^{\mathrm{nat}}_{\lambda_{\mathrm{Aff}}} \lambda x.\pi_2(x),$$
$$\mathbf{Y}\ M =^{\mathrm{nat}}_{\lambda_{\mathrm{Aff}}} M\ (\mathbf{Y}\ M).$$

Definition 5.10. We extend the linear CPS translation of λ_v *to* $\mathrm{PCF_v}$ by setting $\overline{\mathtt{nat}} = \mathrm{nat}$ and adding translations of the constants as follows:

- $\overline{\mathtt{0}} = \lambda k.(k\ 0)$,
- $\overline{\mathtt{succ}} = \lambda k.(k\ (\lambda x.\lambda s.s\ (\mathrm{succ}\ x)))$,
- $\overline{\mathtt{pred}} = \lambda k.(k\ (\lambda x.\lambda s.(\mathrm{IF0}\ x)\ \langle \Omega, s\ (\mathrm{pred}\ x)\rangle))$,

- $\overline{\mathtt{IF0}} = \lambda k.(k\ \lambda m.\lambda a.(a\ \lambda n.\lambda b.(b\ \lambda l.\lambda c.(\mathrm{IF0}\ m)\langle n\ c, l\ c\rangle)))$,
- $\overline{\mathtt{Y}} = \lambda k.(k\ \lambda f.\mathbf{Y}\ \lambda y.\lambda z.f\ \lambda m.y\ (\lambda a.(m\ a)\ z))$.

Lemma 5.11. *For any PCF program M, $M \Downarrow V$ implies $\overline{M} =^{\mathrm{nat}}_{\lambda_{\mathrm{Aff}}} \overline{V}$.*

Proof. We first show that if $\overline{V} = \lambda k.k\ M$, then $\overline{N[V/x]} = \overline{N}[M/x]$. The lemma then follows by a straightforward consideration of cases. □

We interpret nat in $\mathsf{Fam}(\mathcal{CG})$ as the family $\{\mathbf{1}_i \mid i \in \mathbb{N}\}$, giving obvious denotations of the constants 0, succ, pred. IF0 is interpreted as a family of strategies $\{\sigma_i : o\&o \multimap o\}$ where $\sigma_0 = \pi_l$ and $\sigma_i = \pi_r$ for all $i > 0$.

Since strategies on pointed type-objects are ordered (continuously) by inclusion, we may interpret the combinator $\mathbf{Y} : (P \Rightarrow P) \Rightarrow P$ (for each pointed type P) in standard fashion as the least fixed point of $\lambda \mathsf{fg}.\mathsf{g}\ (\mathsf{f}\ \mathsf{g})$. *Soundness* of the interpretation with respect to axioms of Definition 5.9 is straightforward.

Lemma 5.12. *If $M =^{\mathrm{nat}}_{\lambda_{\mathrm{Aff}}} N$, then $[\![M]\!] = [\![N]\!]$.*

Since $\mathsf{Fam}_{\Sigma}(\mathcal{WB})$ is a model of the computational λ-calculus [23] we may interpret $\mathrm{PCF_v}$ by setting $[\![\mathtt{nat}]\!] = \{\mathbf{1}_i \mid i \in \mathbb{N}\}$, interpreting the constants as in [3].

Proposition 5.13. *For every PCF-type T, $\phi([\![T]\!]_{\mathcal{WB}}) = [\![\overline{T}]\!]_{\mathcal{CG}}$ and for every term $M : T$, $\phi([\![M]\!]_{\mathcal{WB}}) = [\![\overline{M}]\!]_{\mathcal{CG}}$.*

Proof. This is by structural induction on M using Theorem 4.10 and Proposition 5.7 together with case-by-case verification for the constants. An example induction case is:

$$\begin{aligned}
&\phi([\![\Gamma \vdash \lambda x.M]\!]_{\mathcal{WB}}) = \phi(\Lambda([\![\Gamma, x \vdash M]\!]_{\mathcal{WB}}); \eta)\\
&\quad = \Lambda(\phi([\![\Gamma, x \vdash M]\!])); \phi(\eta) \text{ by Theorem 4.10}\\
&\quad = \Lambda([\![\overline{\Gamma} \vdash \overline{M}]\!]); \phi(\eta) \text{ by inductive hypothesis}\\
&\quad = [\![\overline{\Gamma} \vdash \lambda k.k\ \lambda x.\overline{M}]\!] \text{ by Proposition 5.7.} \qquad \square
\end{aligned}$$

We may now use this result, together with definability of compact terms in the well-bracketed model [3], to prove a full abstraction result for the linear CPS translation.

Given a term $M : (\mathbf{T} \Rightarrow R) \multimap \mathrm{R}$, we may write $M \Downarrow$ if M has a head-normal form—i.e. $M =^{\mathrm{nat}}_{\lambda_{\mathrm{Aff}}} \lambda k.k\ M'$ for some M'. Let $\simeq_L$ be contextual equivalence of $\lambda^{\mathrm{nat}}_{\mathrm{Aff}}$ terms defined with respect this notion of convergence [6] —i.e. if $M, N : T$ are closed terms, then $M \simeq_L N$ if for all compatible contexts $C[\_] : (S \Rightarrow \mathrm{R}) \multimap \mathrm{R}$, $C[M] \Downarrow$ if and only if $C[N] \Downarrow$.

Theorem 5.14. *The linear CPS translation from $\mathrm{PCF_v}$ to $\lambda^{\mathrm{nat}}_{\mathrm{Aff}}$ is fully abstract, i.e. for all closed PCF terms $M, N : T$, $M \simeq_P N$ if and only if $\overline{M} \simeq_L \overline{N}$.*

[6] The separating contexts must be of linear type to obtain a full abstraction result, otherwise we may make observations in $\lambda^{\mathrm{nat}}_{\mathrm{Aff}}$ not available in PCF.

Proof. To prove soundness, suppose $M \not\simeq_P N$. Let $U = \lambda z.M$ and $V = \lambda z.N$ where $z \notin FV(M) \cup FV(N)$. Then (w.l.o.g.) there exists $C[\_]$ such that $C[U] \Downarrow$ and $C[V] \not\Downarrow$, and hence $(\lambda x.C[x])\ U \Downarrow$ and $(\lambda x.C[x])\ V \not\Downarrow$. Hence $\overline{(\lambda x.C[x])\ U} \Downarrow$ by Lemma 5.12, and $\{\phi\}(\llbracket(\lambda x.C[x])\ V\rrbracket) = \llbracket\overline{(\lambda x.C[x])\ V}\rrbracket = \bot$ by adequacy of the well-bracketed model of $\mathrm{PCF_v}$ [3], and so $\overline{(\lambda x.C[x])\ V} \not\Downarrow$, and hence $\overline{M} \not\simeq_L \overline{N}$, as required.

For the converse, suppose $\overline{M} \not\simeq_L \overline{N}$. Then without loss of generality there exists $C[\_] : (\mathrm{nat} \Rightarrow \mathrm{R})\multimap\mathrm{R}$ such that $C[\overline{M}] \Downarrow$ and $C[\overline{N}] \not\Downarrow$, and so $\llbracket(\lambda x.C[x]\ \overline{M}\rrbracket \neq \bot$ and $\llbracket\lambda x.C[x]\ \overline{N}\rrbracket = \bot$. By Proposition 4.9, $\llbracket\lambda x.C[x]\rrbracket$ is a well-bracketed strategy on $\llbracket T \Rightarrow \mathtt{nat}\rrbracket_{\mathcal{WB}}$, and by continuity, it has a compact approximant ρ such that $\langle\rho, \llbracket M\rrbracket_{\mathcal{WB}}\rangle; \mathsf{App} \neq \bot$ and $\langle\rho, \llbracket N\rrbracket_{\mathcal{WB}}\rangle = \bot$. By definability of compact strategies in the well-bracketed model [3], ρ is definable as a value $V : T \Rightarrow \mathtt{nat}$ such that $\llbracket V\rrbracket = \rho; \eta$ and hence by adequacy of the well-bracketed model, $V\ M \Downarrow$ and $V\ N \not\Downarrow$. □

Neither of the models of $\mathrm{PCF_v}$ are *universal*, because they are not effectively presented, and so there are strategies on $\llbracket\mathtt{nat} \Rightarrow \mathtt{nat}\rrbracket$, for example, which don't correspond to computable functions. However, it would seem to be largely a formality to obtain a universality result for an effectively presented version of the model in $\mathsf{Fam}(\mathcal{WB})$ using the methodology for call-by-name PCF [6,14]. This would give us an easy proof that the linear CPS translation is universal—i.e. for every $\lambda^{\mathrm{nat}}_{\mathrm{Aff}}$ term at a translated type $M : \overline{T}$, there exists a PCF term $N : T$ such that $\overline{N} \simeq_L M$.

6. Further directions

We will now sketch some directions in which the semantics of linear CPS may be developed, to model languages with different features such as recursive types, call-by-name evaluation, state and limited control features.

6.1. Recursive types

We describe briefly how to interpret the addition of recursive types to λ_{Aff}, and how this allows us to analyse the completeness properties of untyped calculi. Our motivating example is the linear CPS translation of the untyped version of λ_v (as in [8]).

We add recursive types to λ_{Aff} by the addition of type-variables and a fixpoint operator on pointed types, which are thus given by the grammar:

$$P ::= X \mid \mathrm{R} \mid T\multimap P \mid P\&P \mid A \Rightarrow P \mid \mu X.P.$$

We extend the typing rules for λ_{Aff} terms:

$$\frac{\Gamma; \Delta \vdash M : \mu X.P}{\Gamma; \Delta \vdash M : P[\mu X.P/X]} \quad \frac{\Gamma; \Delta \vdash M : P[\mu X.P/X]}{\Gamma; \Delta \vdash M : \mu X.P}.$$

The CPS translation of untyped λ_v may be typed in λ_{Aff} by setting $D = \mu X.(X \Rightarrow (X \Rightarrow \mathrm{R})\multimap\mathrm{R})$ and translating each term M with free variables $x_1, \ldots x_n$ as a λ_{Aff} term-in-context $\overline{x_1} : D, \ldots, \overline{x_n} : D \vdash \overline{M} : (D \Rightarrow \mathrm{R})\multimap\mathrm{R}$ [8].

To interpret recursive types in $\mathcal{CG}$, we use the "information-system-like" approach studied in depth in [22], which allows domain equations to be solved up to equality. First, we define an inclusion order on C-arenas.

Definition 6.1. $A_1 \trianglelefteq A_2$ if:
- $M_{A_1} \subseteq M_{A_2}$,
- $\vdash_{A_1} = \vdash_{A_2} \cap ((M_{A_1})_* \times M_{A_1})$,
- $\sim_{A_1} = \sim_{A_2} \cap (M_{A_1} \times M_{A_1})$.

This order is inherited in $\mathsf{Fam}(\mathcal{CG})$: $\{A_i \mid i \in I\} \quad \{B_j \mid j \in J\}$ if $I = J$ and $A_i \quad B_i$ for each $i \in I$.

If $A \trianglelefteq B$ then there are obvious inclusion and projection morphisms $A \to^{\mathtt{in}} B \to^{\mathtt{out}} A$ such that $\mathtt{in}; \mathtt{out} = \mathsf{id}_A$ and $\mathtt{out}; \mathtt{in} \subseteq \mathsf{id}_B$.

Proposition 6.2. *C-Arenas form a large cpo, ordered by* $\trianglelefteq$*, and each of the functors* &, $\multimap$ *and* ! *are continuous with respect to* $\trianglelefteq$.

Thus we can define a least fixed point operator by iteration.

Definition 6.3. Given a continuous functor $F : \mathsf{Fam}(\mathcal{CG})^{OP} \times \mathsf{Fam}(\mathcal{CG}) \to \{\mathcal{WO}\}$, let $\Delta(F) \in \{\mathcal{WO}\} = \bigsqcup_{i \in \omega} F^i(\mathbf{1})$.

Then $F(\Delta(F), \Delta(F)) = \Delta(F)$ (moreover, this is a *minimal invariant* for F [26]) and so we have a sound model of λ_{Aff}, based on the interpretation of types with n free variables as mixed-variance functors from $(\mathsf{Fam}(\mathcal{CG})^{OP} \times \mathsf{Fam}(\mathcal{WO}))^n$ to $\{\mathcal{WO}\}$ via the functors &, $\multimap$ and $\Rightarrow$, and minimal invariant Δ.

As for PCF, we can show that linear CPS interpretation of the untyped λ_v-calculus in $\mathcal{CG}$ is equivalent, via ϕ embedding, to the model in $\mathcal{WB}$ obtained by solving the domain equation $D = D \Rightarrow D_\perp$ [22], and thereby prove full abstraction for the translation. However, allowing only pointed recursive types is too restrictive to allow the linear CPS of PCF to be extended to a metalanguage with recursive types such as FPC [28], which contain unpointed recursive types such as $\mu X.X + 1$. We require a notion of recursive type for λ_{Aff} which admits this, but not e.g. $\mu X.(X \Rightarrow (1 + 1))$.

6.2. Call-by-name CPS interpretation

There are two different CPS interpretations for call-by-name languages which one might wish to "linearize"; the "extensional" CPS interpretation of Streicher and Reus [31], and Plotkin's original call-by-name CPS translation [27]. The linear CPS semantics of the former is a fairly straightforward development from call-by-value. For example we may interpret the continuations at call-by-name PCF types: $[\![\mathtt{nat}]\!]_c = \Pi_{i \in I} o$, and $[\![A \Rightarrow B]\!]_c = !([\![A]\!]_c \multimap o) \otimes [\![B]\!]_c$.[7]

Terms in context $x_1 : S_1, \ldots x_n : S_n \vdash M : T$ are then interpreted as morphisms from $!([\![S_1]\!]_c \multimap o) \otimes \ldots \otimes !([\![S_n]\!]_c \multimap o)$ to $[\![T]\!]_c \multimap o$, following Streicher and Reus [31]. As expected, this is equivalent to the original Hyland–Ong interpretation in $\mathcal{WB}$, via embedding in $\mathcal{WO}_!$.

[7] Note that this does not correspond to a type of λ_{Aff}—we require a different source language to describe the corresponding call-by-name linear cps translation.

The non-extensional (Plotkin-style) linear CPS interpretation of call-by-name presents some more subtle problems. The corresponding translation may be presented as follows:

- $\overline{x} = x$,
- $\overline{\lambda x.M} = \lambda k.(k\ \lambda x.M)$,
- $\overline{M\ N} = \lambda k.(M\ (\lambda m.((m\ N)\ k)))$.

We have four possible translations for the lazy function type $S \Rightarrow T$ in λ_{Aff}, corresponding to four different representations of lifting: $((\overline{S} \Rightarrow \overline{T}) \multimap \mathrm{R}) \multimap \mathrm{R}$, $((\overline{S} \Rightarrow \overline{T}) \Rightarrow \mathrm{R}) \multimap \mathrm{R}$, $((\overline{S} \Rightarrow \overline{T}) \multimap \mathrm{R}) \Rightarrow \mathrm{R}$, and $((\overline{S} \Rightarrow \overline{T}) \Rightarrow \mathrm{R}) \Rightarrow \mathrm{R}$.

If we adopt the second of these (which corresponds to the call-by-value linear CPS translation in that lifting is represented as $(\_ \Rightarrow \mathrm{R}) \multimap \mathrm{R}$) then the translation is not fully abstract. This does not introduce first-class continuations, but the translation of "sequential composition" (which tests its first argument for convergence to a value and then evaluates the second):

$$\overline{M; N} = \lambda k.\overline{M}\ \lambda m.\overline{N}\ k$$

can be typed. The presence of sequential composition is sufficient to change the observational equivalence in the language [4], and thus to break full abstraction for the translation into λ_{Aff}.[8] Conversely, we may adopt the third typing, which admits **call**/**cc**, but not sequential composition (this approach is used by Laurent and Regnier [21] to give a CPS translation of the $\lambda\mu$-calculus), or the fourth, which admits both.

The minimal typing of the translation of the lazy function type in λ_{Aff} is $\overline{S \Rightarrow T} = (\overline{S} \Rightarrow \overline{T}) \multimap \mathrm{R}) \multimap \mathrm{R}$. Note, however, that we can still type the translation of sequential composition in λ_{Aff}; in order to obtain a fully abstract translation, we need to restrict the weakening rule in our target calculus to non-linear variables. Therefore to give an accurate semantics of the linear CPS translation of the lazy λ-calculus, we require a model of the linear/non-linear λ-calculus which does not allow weakening. We can construct such a semantics based on coherence games and *complete* plays.

Definition 6.4. A (Player) *complete* justified sequence over a C-arena A is a coherent sequence s such that for every O-move a in s, if b is a P-move enabled by a such that $b \not\frown b$, then there is a P-move c in s such that c is enabled by a, and $c \not\frown b$.

The linear CPS transform corresponds to a functor into our linear category from the category of well-bracketed and *persistent* strategies used by di Gianantonio and Ong [25] to give a definability result for the lazy λ-calculus. Representing strategies via complete plays introduces a significant amount of technical complication, and so this work will be presented elsewhere.

[8] In fact, a satisfactory categorical account of the semantics of the pure lazy λ-calculus using a lifted function type has proved elusive, because sequential composition is derivable from the functoriality of lifting acting on a CCC. The linear CPS interpretation resolves this problem by separating the linear category (on which lifting acts as a functor) from the cartesian closed one.

6.3. Semantics of backwards jumps

As we have already observed in our examples, jumping behaviour in the game semantics corresponds to allowing structural rules (duplication and promotion) at only limited types such as the response type for continuations. We may admit such rules whilst retaining the property that continuations are called only once during evaluation (provided that general references are not available, otherwise jumps are likely to be sufficient to express first-class continuations). We may use this observation to give linear CPS semantics of functional languages without general first-class continuations but with backwards jumps. This makes many of the advantages of non-local control flow available, whilst avoiding the more complex behaviour which arises when continuations may be called multiple times.

In the setting of call-by-value PCF, for example, we may type the CPS translation of first-order $\mathbf{call/cc} : ((\mathtt{nat} \Rightarrow T) \Rightarrow \mathtt{nat}) \Rightarrow \mathtt{nat}$

$$\overline{\mathbf{call/cc}\ M} = \lambda k.\overline{M}\ (\lambda m.(m\ \lambda ab.(k\ a))\ k)$$

provided we allow contraction at the answer type $\mathtt{nat} \Rightarrow \mathrm{R}$: i.e. we add a structural rule:

$$\frac{\Gamma, x : (\mathtt{nat} \Rightarrow \mathrm{R}); \Delta \vdash M : T}{\Gamma; \Delta, x : (\mathtt{nat} \Rightarrow \mathrm{R}) \vdash M : T}.$$

From a syntactic perspective, this translation is not very satisfactory—it is clearly necessary to heavily restrict β-reduction in the calculus in order to preserve subject reduction. However, the semantic interpretation is very simple—we have already observed that forcing Opponent to obey the visibility condition means that there is a copycat between $o \Rightarrow o$ and $o \multimap o$. Since $[\![\mathtt{nat} \Rightarrow \mathrm{R}]\!] = \{\Pi_{i\in\omega} o\}$, we may interpret the above structural rule (as a copycat strategy from $(\Pi_{i\in\omega} o) \Rightarrow o$ to $(\Pi_{i\in\omega} o) \multimap o$) in a category of games in which both participants are subject to visibility.

6.4. Linear CPS translation and state

One of the most significant developments in game semantics has been in modelling languages with state, such as Idealized Algol [2], and ML-style higher-order references [5]. It is natural to consider whether the game semantics of linear CPS can be extended to these models: can we give a game semantics of an imperative version of λ_{Aff}, so that linear CPS interpretation via such a language corresponds to the direct (well-bracketed) models of state?

The key feature of the games model of Idealized Algol is that the "knowing" strategies used to interpret stateful programs are not dependent only on the Player view but on the whole history of play. Moreover, the correspondence between knowing behaviour and the use of state is captured by a *factorization* result, used to prove definability of compact strategies and hence full abstraction: all knowing strategies can be obtained as the result of interaction between an innocent strategy and a reference cell, represented as a strategy $\mathsf{cell} : I \to \mathsf{var}$ (where var is the interpretation of the type of natural number references).

To give a model of imperative λ_{Aff} we need to characterize those strategies which are the result of interaction between a *coherent* innocent strategy and a reference cell.

Definition 6.5. A knowing strategy is coherent if whenever $s, t \in \sigma$, and sa is coherent, and $\ulcorner sab \urcorner = \ulcorner t \urcorner$, then sab is coherent.

(Knowing C-strategies are defined in the same way as innocent C-strategies.) We then have the following factorization property: a knowing strategy $\sigma : I \to A$ is coherent if and only if there exists an innocent coherent strategy $\widehat{\sigma} : \mathsf{var} \to A$ such that $\mathsf{cell}; \widehat{\sigma} = \sigma$.

Thus knowing and coherent strategies form a category (with the same coherence structure as $\mathcal{CG}$) in which we may give an interpretation of λ_{Aff} extended with ground-type reference variables, with a universality result for compact strategies. As in the innocent case, we have a full and faithful functor into our category from a category of well-bracketed games and knowing strategies, which sends the model of RML described in [3] to a linear CPS model.

Any extension to general references would seem more problematic. Modelling references at higher types requires the condition of Player-visibility to be dropped, and so there does not seem to be any way of recovering the bracketing condition from coherence. Thus, whilst we may construct a coherence games model of λ_{Aff} with references, it does not correspond via linear CPS translation to an intuitionistically typed source language, but to one which already has first-class linear continuations. These may, however, be useful in the control of interference between references [32].

Acknowledgements

I would like to thank the referees for a correction and several helpful comments.

References

[1] S. Abramsky, Axioms for full abstraction and full completeness, in: G. Plotkin, M. Fofte, C. Stirling (Eds.), Essays in Honour of Robin Milner, MIT Press, Cambridge, MA, 1997.
[2] S. Abramsky, G. McCusker, Linearity, sharing and state: a fully abstract game semantics for idealized algol with active expressions, in: P.W. O'Hearn, R. Tennent (Eds.), Algol-like Languages, Birkhauser, Basel, 1997.
[3] S. Abramsky, G. McCusker, Call-by-value games, in: M. Neilsen, W. Thomas (Eds.), Proc. CSL '97, Springer, Berlin, 1998, pp. 1–17.
[4] S. Abramsky, C.-H.L. Ong, Full abstraction in the lazy λ-calculus, Inform. Comput. 105 (1993) 159–267.
[5] S. Abramsky, K. Honda, G. McCusker, A fully abstract games semantics for general references, in: Proc. 13th Annu. Symp. on Logic in Computer Science, LICS '98, 1998.
[6] S. Abramsky, R. Jagadeesan, P. Malacaria, Full abstraction for PCF, Inform. Comput. 163 (2000) 409–470.
[7] A. Barber, Dual intuitionistic linear logic, Technical Report ECS-LFCS-96-347, LFCS, University of Edinburgh, 1996.
[8] J. Berdine, P. O'Hearn, U. Reddy, H. Thielecke, Linear continuation-passing, Higher Order Symbol. Comput. 15 (2/3) (2002) 181–208.
[9] J. Berdine, P.W. O'Hearn, H. Thielecke, On affine typing and completeness of CPS, Draft, December 2000.
[10] O. Danvy, F. Pfenning, The occurrence of continuation parameters in CPS terms, Technical Report CMU-CS-95-121, Department of Computer Science, Carnegie Mellon University, 1995.
[11] A. Filinski, Linear continuations, in: Proc. 19th Annu. ACM Symp. on Principles of Programming Languages, 1992.
[12] M. Hasegawa, Linearly used effects: monadic and CPS transformations into the linear lambda calculus, in: Proc. 6th Internat. Symp. on Functional and Logic Programming (FLOPS2002), Aizu, Lecture Notes in Computer Science, Vol. 2441, Springer, Berlin, 2002, pp. 167–182.

[13] K. Honda, N. Yoshida, Game theoretic analysis of call-by-value computation, in: Proc. ICALP '97, Lecture Notes in Computer Science, Vol. 1256, Springer, Berlin, 1997.
[14] J.M.E. Hyland, C.-H.L. Ong, On full abstraction for PCF: I, II and III, Inform. Comput. 163 (2000) 285–408.
[15] J. Laird, Full abstraction for functional languages with control, in: Proc. 12th Internat. Symp. on Logic in Computer Science, LICS '97, IEEE Computer Society Press, Silver Spring, MD, 1997.
[16] J. Laird, A Semantic Analysis of Control, Ph.D. Thesis, Department of Computer Science, University of Edinburgh, 1998.
[17] J. Laird, Finite models and full completeness, in: Proc. CSL '00, Lecture Notes in Computer Science, Vol. 1862, Springer, Berlin, 2000.
[18] J. Laird, A fully abstract game semantics of local exceptions, in: Proceedings of LICS '01, IEEE Computer Society Press, Silver Spring, MD, 2001.
[19] O. Laurent, Étude de la polarisation en logique, Ph.D. Thesis, Université d'Aix-Marseille II, 2002.
[20] O. Laurent, Polarized games, in: Proc. 17th Internat. Symp. on Logic in Computer Science, LICS '02, 2002.
[21] O. Laurent, L. Regnier, About translations of classical logic into polarized linear logic, in: Proc. LICS '03, 2003, pp. 11–20.
[22] G. McCusker, Games and full abstraction for a functional metalanguage with recursive types, Ph.D. Thesis, Imperial College London, 1996, Cambridge University Press, Cambridge.
[23] E. Moggi, Notions of computation and monads, Inform. Comput. 93 (1) (1991).
[24] H. Nickau, Hereditarily sequential functionals, in: Proc. Symp. on Logical Foundations of Computer Science: Logic at St. Petersburg, Lecture Notes in Computer Science, Springer, Berlin, 1994.
[25] C.-H.L. Ong, P. di Gianantonio, Games characterizing Levy–Longo trees, in: Proc. ICALP '02, 2002, pp. 476–487.
[26] A.M. Pitts, Relational properties of domains, Inform. Comput. 127 (1996) 66–90.
[27] G. Plotkin, Call-by-name, call-by-value and the λ-calculus, Theoret. Comput. Sci. 1 (1975) 125–159.
[28] G. Plotkin, Lectures on predomains and partial functions, Notes for a course given at the Center for the study of Language and Information, Stanford, 1985.
[29] J. Polakow, F. Pfenning, Properties of terms in continuation-passing style in an ordered logical framework, in: Workshop on Logical Frameworks and Metalanguages, Santa Barbera, 2000.
[30] P. Selinger, Control categories and duality: on the categorical semantics of the lambda-mu calculus, Math. Struct. Comput. Sci. 11 (2001) 207–260.
[31] T. Streicher, B. Reus, Classical logic: continuation semantics and abstract machines, J. Funct. Programming 8 (6) (1998) 543–572.
[32] S. Zdancewic, A.C. Myers, Secure information flow via linear continuations, Higher Order Symbol. Comput. 15 (2002).