A Games Model of Bunched Implications

Guy McCusker[1] and David Pym[2]

[1] Department of Computer Science, University of Bath
Bath BA2 7AY
United Kingdom
G.A.McCusker@bath.ac.uk
[2] Hewlett-Packard Laboratories
Bristol BS34 8QZ
United Kingdom
david.pym@hp.com

Abstract. A game semantics of the $(-\!\!*, \rightarrow)$-fragment of the logic of bunched implications, **BI**, is presented. To date, categorical models of **BI** have been restricted to two kinds: functor category models; and the category **Cat** itself. The game model is not of this kind. Rather, it is based on Hyland-Ong-Nickau-style games and embodies a careful analysis of the notions of resource sharing and separation inherent in **BI**. The key to distinguishing between the additive and multiplicative connectives of **BI** is a semantic notion of separation. The main result of the paper is that the model is fully complete: every finite, total strategy in the model is the denotation of a term of the $\alpha\lambda$-calculus, the term language for the fragment of **BI** under consideration.

1 Introduction

The logic of bunched implications, **BI** [9–11], is a substructural logic which treats multiplicative and additive versions of its connectives on an equal footing, and in doing so gives an account of the notions of sharing and separation of resources. As a result, **BI** has clear applications to computer science: using **BI** as a type system gives rise to elegant approaches to interference-control in imperative programming [10] improving on Reynolds's *Syntactic Control of Interference* [13]; a particular model of **BI** has become known as *Separation Logic* [14] and is now a widely studied approach to local reasoning about programs which manipulate memory in challenging ways; and **BI** has also been developed into a Hennessy-Milner style logic for specification and reasoning about resource-sensitive properties of processes and systems [12].

From the outset, **BI** has enjoyed a rich semantic theory based on the notion of cartesian doubly-closed category: models of **BI** are categories possessing two distinct monoidal-closed structures, one of which is cartesian. To date the only known instances of this structure are functor category models and the category of categories, **Cat**.

Game semantics is an approach to modelling logics and programming languages which has seen considerable success in the last fifteen years. The first major results were the fully-abstract games models of PCF [1, 4, 7], and subsequently a wide variety of programming languages and logics have been modelled, very often with full abstraction or full completeness properties.

In this paper, we use game semantics to give a new model of a fragment of **BI**, in the form of the associated term-language, the $\alpha\lambda$-calculus [8–11]. Our model is a refinement of the Hyland-Ong-Nickau style model of the λ-calculus. We demonstrate that the model is fully complete, that is, that every finite, total element of the model is the denotation of a term of the language, i.e., a proof in **BI**.

There are many further directions for this work. As well as going on to model larger fragments of **BI**, we would like to combine the approach to resource-sensitivity introduced here with the games models of imperative programming [2], aiming to arrive at a semantic account of interference control extending that of [15]. The model given here incorporates a relational, rather than a syntactic, notion of separation, similar to the language λ_{sep} [3] and we conjecture that our model is also a model for that calculus.

2 Bunched Implications and the $\alpha\lambda$-calculus

We will not present **BI** directly but instead move straight to the associated term-language, the $\alpha\lambda$-calculus. The types of the $\alpha\lambda$-calculus are as follows:

$$A ::= \gamma \mid A \mathrel{-\!\!*} A \mid A \to A,$$

where γ ranges over a collection of ground types. Its terms are given by the grammar

$$M ::= x \mid \lambda x.M \mid MM \mid \alpha x.M \mid M \mathbin{@} M.$$

The standard β- and η- reduction apply to both kinds of abstraction and application, e.g. $(\alpha x.M) \mathbin{@} N \to M[N/x]$ and $(\lambda x.M)N \to M[N/x]$, with the usual notions of free and bound identifiers, capture-free substitution etc.

The typing rules are based on judgements of the form $\Gamma \vdash M : A$, where M is a term, A is a type, and Γ is a *bunch*: bunches are described by the grammar

$$\Gamma ::= I \mid x : A \mid \Gamma, \Gamma \mid \Gamma; \Gamma.$$

I is the empty bunch; $x : A$ is a singleton bunch; and there are two bunch-forming operations, comma and semicolon. The idea is that in a bunch Γ, Δ there is no resource sharing between Γ and Δ, while in $\Gamma; \Delta$ there may be. Formally, this is achieved by allowing contraction across semicolons but not across commas.

We write $\Gamma(\Delta)$ to indicate a bunch in which Δ appears as a subtree, and then $\Gamma(\Delta')$ indicates the similar tree where Δ is replaced by Δ'. Bunches are identified up to ***coherent equivalence***: $\equiv$ is the smallest equivalence relation on bunches including commutative monoid equations for I and ; , commutative monoid equations for I and , and congruence: if $\Delta \equiv \Delta'$ then $\Gamma(\Delta) \equiv \Gamma(\Delta')$. Note that in our presentation we do not have separate units for the two bunch-forming operations. This is because our version of the $\alpha\lambda$-calculus and our model will incorporate weakening for both connectives. That is, we treat the *affine* version of **BI**, so the units are identified.

The typing rules are as follows:

$$\frac{\Gamma \vdash M : A}{\Delta \vdash M : A}\;\Gamma \equiv \Delta \text{ (coherence)} \qquad \frac{}{x : A \vdash x : A}\text{(id)}$$

$$\frac{\Gamma(\Delta) \vdash M : A}{\Gamma(\Delta, \Delta') \vdash M : A}\text{(weakening for ,)} \qquad \frac{\Gamma(\Delta) \vdash M : A}{\Gamma(\Delta; \Delta') \vdash M : A}\text{(weakening for ;)}$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \mathrel{-\!\!*} B}\text{(}\lambda\text{-abstraction)} \qquad \frac{\Gamma \vdash M : A \mathrel{-\!\!*} B \quad \Delta \vdash N : A}{\Gamma, \Delta \vdash MN : B}\begin{pmatrix}\text{multiplicative}\\ \text{application}\end{pmatrix}$$

$$\frac{\Gamma; x : A \vdash M : B}{\Gamma \vdash \alpha x.M : A \to B}\text{(}\alpha\text{-abstraction)} \qquad \frac{\Gamma \vdash M : A \to B \quad \Delta \vdash N : A}{\Gamma; \Delta \vdash M \mathbin{@} N : B}\begin{pmatrix}\text{additive}\\ \text{application}\end{pmatrix}$$

$$\frac{\Gamma(\Delta; \Delta') \vdash M : B}{\Gamma(\Delta) \vdash M[\mathsf{idents}(\Delta)/\mathsf{idents}(\Delta')] : B}\;\Delta \cong \Delta' \text{ (contraction)}$$

In the final rule, $\Delta \cong \Delta'$ means that the bunches Δ and Δ' are the same up to renaming of identifiers, and the substitution $M[\mathsf{idents}(\Delta)/\mathsf{idents}(\Delta')]$ simultaneously replaces each identifier of Δ' appearing in M with the corresponding one from Δ.

The simply-typed λ-calculus is of course the fragment of the $\alpha\lambda$-calculus where only ; is used in bunches, and only α and @ are used as term-formers. Even though our syntax for this fragment uses α as the binder, we will continue to call it the λ-calculus! (Note that α and λ are mnemonics for the αdditive and λinear abstractions respectively.)

We first establish some simple facts about the type system.

If Γ is a bunch containing identifiers x and y, then there is a unique sub-bunch of the form Γ_1, Γ_2 or $\Gamma_1; \Gamma_2$ such that x appears in Γ_1 and y in Γ_2. We say that x and y are ***separated*** in Γ if this sub-bunch has the form Γ_1, Γ_2. That is, x and y are in sub-bunches combined by the comma rather than the semicolon.

Lemma 1. *A term $\Gamma \vdash MN : B$ is typeable if and only if there are typeable terms $\Gamma \vdash M : A \mathrel{-\!\!*} B$ and $\Gamma \vdash N : A$, and all the free identifiers of M are separated from all the free identifiers of N in Γ.*

Proof The only-if direction is a straightforward structural induction on derivations. For the other direction, we show that Γ can be split as $\Gamma = \Gamma_1, \ldots, \Gamma_n$ where each Γ_i contains identifiers from at most one of M and N, and the result follows easily. □

The standard additive-style application rule for @ is admissible:

Lemma 2 (Additive application). *If $\Gamma \vdash M : A \to B$ and $\Gamma \vdash N : A$ then $\Gamma \vdash M \mathbin{@} N : B$.*

Proof Rename the variables in N to obtain $\Gamma' \vdash N' : A$ with $\Gamma \cong \Gamma'$, then use the application and contraction rules. □

3 A game semantics for the $\alpha\lambda$-calculus

In this section, we briefly recall the basic definitions and results in Hyland-Ong-Nickau style game semantics for the λ-calculus, including the full completeness argument, before going on to refine the model to handle the subtleties of the **BI**. The definitions we use are essentially those of [6], with one or two small presentational changes to facilitate our later refinements.

3.1 Arenas

A game has two participants: Player (P) and Opponent (O). A *play* of the game consists of a sequence of moves, alternately by O and P. In addition, each move is explicitly *justified* by an earlier move of the play, unless it is a special kind of move, called *initial*, which needs no justification. In the games we consider, O always moves first.

Before embarking on a formal definition, let us fix notation for sequences and operations on them. We use s, t, ... to range over sequences and a, b, ..., m, n, ... over elements of sequences. If s and t are sequences, then st or $s \cdot t$ is their concatenation; ε is the empty sequence. A move a will often be identified with the singleton sequence consisting just of a. The sequence $s_{<a}$ is the prefix of s up to, but not including, an element a, while $s_{\leq a}$ does include a.

An ***arena*** is specified by a triple $A = \langle M_A, \lambda_A, \vdash_A \rangle$ where:

- M_A is a set of ***moves***;
- $\lambda_A : M_A \to \{\mathsf{O}, \mathsf{P}\}$ is a ***labelling*** function which indicates whether a move is by Opponent (O) or Player (P). The function $\overline{\lambda_A}$ is λ_A with the O and P reversed. If $\lambda(a) = \mathsf{O}$, we call a an O-move; otherwise, a is a P-move;
- $\vdash_A$ is a relation between $M_A + \{\star\}$ and M_A, called ***enabling***, which satisfies
  - $\star \vdash_A a \Rightarrow [b \vdash_A a \iff b = \star]$, and
  - $a \vdash_A b \wedge a \neq \star \Rightarrow \lambda_A(a) \neq \lambda_A(b)$.

The enabling relation tells us either that a move a is ***initial*** and needs no justification ($\star \vdash_A a$), or that it can be justified by another move b, if b has been played ($b \vdash_A a$). An arena is called ***negative*** if all its initial moves belong to O.

A ***justified sequence*** s of moves in an arena A is a sequence of moves together with ***justification pointers***: for each move a in s which is not initial, there is a pointer to an earlier move b of s such that $b \vdash_A a$. We say the move b ***justifies*** a, and extend this terminology to say that a move b ***hereditarily justifies*** a if the chain of pointers back from a passes through b.

Given a justified sequence s, the ***view*** $\mathsf{view}(s)$ of s is defined as follows:

$$\mathsf{view}(\varepsilon) = \varepsilon \qquad \begin{array}{r}\mathsf{view}(s \cdot a) = a, \text{if } a \text{ is an initial move}\\ \mathsf{view}(s \cdot \overparen{a \cdot t \cdot b}) = \mathsf{view}(s) \cdot \overparen{a \cdot b}.\end{array}$$

If s is a justified sequence containing a move a, we say that a is ***visible at*** s if a appears in $\mathsf{view}(s)$.

A justified sequence s is a ***legal position*** iff:

- O moves first: if $s = as'$ then $\lambda(a) = \mathsf{O}$;

- s is ***alternating***: if $s = s_1 abs_2$ then $\lambda(a) \neq \lambda(b)$;
- The ***visibility condition*** holds: if $s = s_1 as_2$, and a is not initial, then the justifier of a is visible at s_1.

The set of all legal positions of an arena A is written L_A.

3.2 Strategies

A strategy for an arena A is a rule telling Player what move to make in a given position. Formally, this can be represented as a set σ of legal positions in which P has just moved, i.e., a nonempty set of even-length positions, such that

$$sab \in \sigma \Rightarrow s \in \sigma \qquad sab, sac \in \sigma \Rightarrow sab = sac.$$

For any arena A, the smallest possible strategy is $\{\varepsilon\}$, which never makes any response. It is called the ***empty strategy*** and denoted $\perp$.

A strategy is ***innocent*** if for all $sab, t \in \sigma$, if $ta \in L_A$ and $\mathsf{view}(sa) = \mathsf{view}(ta)$, then also $tab \in \sigma$, with b justified by the same element of $\mathsf{view}(ta) = \mathsf{view}(sa)$ as in sab.

3.3 Constructions on arenas

Given negative arenas A and B, the arenas $A \times B$ and $A \vdash B$ are defined as follows:

$$\begin{gathered}
M_{A\times B} = M_{A\vdash B} = M_A + M_B \text{ (disjoint union)} \\
\lambda_{A\times B} = [\lambda_A, \lambda_B] \qquad \lambda_{A\vdash B} = [\overline{\lambda}_A, \lambda_B] \\
\star \vdash_{A\times B} m \iff \star \vdash_{A\vdash B} m \iff \star \vdash_A m \vee \star \vdash_B m \\
m \vdash_{A\times B} n \iff m \vdash_{A\vdash B} n \iff m \vdash_A n \vee m \vdash_B n.
\end{gathered}$$

The idea here is that the games A and B are played in interleaved parallel fashion. In $A \times B$, labelling and enabling are inherited directly from A and B. In $A \vdash B$, the O/P roles in A are reversed; one impact this has is that a legal position in $A \vdash B$ always begins with a move from B. The unit for × is the empty arena $I = \langle\emptyset, \emptyset, \emptyset\rangle$. Note that $A \times B$ is negative if A and B are, while $A \vdash B$ is not. In fact the only arenas we will consider which are not negative are those of the form $A \vdash B$, where A and B are negative.

3.4 Composition of strategies

Let A, B and C be negative arenas. An ***interaction sequence*** on A, B, C is a justified sequence u of moves from $M_A + M_B + M_C$ such that

- $u \upharpoonright A, B \in L_{A\vdash B}$,
- $u \upharpoonright B, C \in L_{B\vdash C}$,
- $u \upharpoonright A, C$ is an alternating sequence of moves in $A \vdash C$, and
- there is at least one move between any A-move and any C-move in u.

We write $\mathsf{int}(A, B, C)$ for the set of all such sequences.

Given strategies σ on $A \vdash B$ and τ on $B \vdash C$ we define the composite strategy $\sigma\,;\tau$ on $A \vdash C$ by "parallel composition plus hiding".

$$\sigma\,;\tau = \{s \mid \exists u \in \mathsf{int}(A, B, C). u \upharpoonright A, B \in \sigma \wedge u \upharpoonright B, C \in \tau \wedge u \upharpoonright A, C = s\}.$$

Lemma 3. *Composition of strategies is well-defined (that is, $\sigma\,;\tau$ as defined above is indeed a strategy on $A \vdash C$). Moreover, it is associative and has identity given by the* **copycat strategy**

$$\mathsf{id}_A = \{s \in L_{A \vdash A'} \mid \forall t \sqsubseteq^{even} s.t \upharpoonright A = t \upharpoonright A'\}.$$

(Here A' is the same arena as A; we use the prime only to distinguish the two occurrences. The copycat strategy simply copies O's moves back and forth from one occurrence to the other.) Further, the composite of two innocent strategies is innocent and the identity is itself innocent.

We can now define two categories of games: $\mathcal{G}$ has negative arenas as objects and strategies on $A \vdash B$ as maps from A to B, with composition and identities as above; $\mathcal{G}_{\text{inn}}$ has the same objects but has only innocent strategies as morphisms.

The $\times$ operation on arenas gives a categorical product in $\mathcal{G}_{\text{inn}}$; the projections $A_1 \times A_2 \to A_i$ are given by copycat strategies.

The category $\mathcal{G}_{\text{inn}}$ is in fact cartesian closed: exponentials are given by the arena $A \Rightarrow B$ defined by

$$\begin{array}{ll} M_{A\Rightarrow B} = M_A + M_B & \star \vdash_{A\Rightarrow B} a \iff \star \vdash_B a \\ \lambda_{A\Rightarrow B} = [\overline{\lambda}_A, \lambda_B] & m \vdash_{A\Rightarrow B} n \iff m \vdash_A n \vee m \vdash_B n \vee [\star \vdash_B m \wedge \star \vdash_A n] \end{array}$$

Compare and contrast with $A \vdash B$: in creating $A \Rightarrow B$ we convert initial moves of A to non-initial moves, justified by initial B-moves. Note that this means $A \Rightarrow B$ is always negative if A and B are. Standard presentations use only negative arenas and use this definition for $A \vdash B$, but when we come to refine the category to model **BI** this will no longer suffice: there is an important distinction between morphisms and elements of the exponential type(s).

Now that we have a cartesian closed category, we can interpret the simply-typed λ-calculus in a standard fashion [5]:

- each type A is interpreted as an object $[\![A]\!]$, that is, as an arena, with $[\![A \to B]\!] = [\![A]\!] \Rightarrow [\![B]\!]$
- a context $\Gamma = x_1 : A_1; \ldots ; x_n : A_n$ is interpreted as the product $[\![A_1]\!] \times \cdots \times [\![A_n]\!]$
- each term $\Gamma \vdash M : A$ is interpreted as a map $[\![\Gamma \vdash M]\!]$ (often abbreviated to $[\![M]\!]$) with domain $[\![\Gamma]\!]$ and codomain $[\![A]\!]$.

The interpretation is fixed as soon as we determine an object $[\![\gamma]\!]$ for each ground type: this is done by taking $[\![\gamma]\!]$ to be a one-move arena, consisting of a single initial opponent move γ.

3.5 Full completeness for the λ-calculus

We now sketch the proof of the definability result for λ-calculus which forms the basis of, for example, Hyland, Ong and Nickau's fully abstract models of the programming language PCF [4, 7]. The key to the result is that an innocent strategy σ is determined by the set $\mathsf{branches}(\sigma)$ of legal positions $s \in \sigma$ such that every O-move in s is justified by the immediately preceding P-move, and that such sequences correspond to branches of Böhm-trees. Hence innocent strategies correspond to (partial, potentially infinite) Böhm-trees. Thus innocent strategies which are *total*, meaning they have a response to every move O can make, and *finite*, meaning the set $\mathsf{branches}(\sigma)$ is finite, correspond to ordinary Böhm-trees.

We consider a type A built from a single ground type γ using the $\rightarrow$ constructor, and a total, finite innocent strategy σ on the arena $[\![A]\!]$. Our task is to find a closed Böhm-tree M such that $[\![M]\!] = \sigma$. Using the uncurrying isomorphisms, we may consider an arena of the form $A_1 \times \cdots \times A_n \vdash \gamma$ and search for a term of the form $x_1 : A_1, \ldots, x_n : A_n \vdash M : \gamma$ to denote our strategy.

Let $A_i = A_{i,1} \rightarrow \cdots \rightarrow A_{i,k_i} \rightarrow \gamma_i$, where we use the subscript on γ simply to distinguish it from other occurrences of the ground type. Since σ is total, it has a response to the initial move γ, so there is some sequence $\gamma\gamma_i \in \mathsf{branches}(\sigma)$. The term denoting σ will then take the form $x_i M_1 \ldots M_{k_i}$ for some terms M_j.

Every longer sequence in $\mathsf{branches}(\sigma)$ has the form $\gamma\gamma_i\gamma_{i,j}t$ where $\gamma_{i,j}$ is the initial move of $[\![A_{i,j}]\!]$ for some j. The set $\{\gamma_{i,j}t \mid \gamma\gamma_i\gamma_{i,j}t \in S\}$ then determines (the branches of) a finite, total innocent strategy τ_j on the arena $A_1 \times \cdots \times A_n \vdash A_{i,j}$. This requires relabelling the moves a little: moves of the sequences t which are hereditarily justified by $\gamma_{i,j}$ become moves in the right-hand arena $A_{i,j}$, while all others remain in the left-hand side.

The set $\mathsf{branches}(\tau_j)$ is strictly smaller than $\mathsf{branches}(\sigma)$, so we can use an inductive argument to find a term M_j such that $[\![M_j]\!]$ is this strategy. We can then verify that $\sigma = [\![x_i M_1 \ldots M_{k_i}]\!]$ thus completing the full completeness argument.

The correspondence between branches of Böhm-trees and plays in $\mathsf{branches}(\sigma)$ motivates our forthcoming definitions of the model of the $\alpha\lambda$-calculus, so we expand on the accompanying intuition:

- An O-move corresponds to the selection of a subterm to investigate: the initial move γ commences investigation of the term, and after P responds with γ_i, O can choose any of the $\gamma_{i,j}$, which begins investigation of the subterm M_j;
- A P-move corresponds to the choice of head-variable for the subterm under investigation;
- The justifier of a P move is the O-move corresponding to the subterm where the chosen head-variable was introduced by abstraction. Initial P-moves correspond to free variables;
- Suppose we have an O move m and a later P-move n whose justifier appears before m. The subterm M corresponding to m contains the use of a variable x corresponding to n, and this variable appears free in M, since it was abstracted (justified) outside M.

To illustrate, consider the λ-term

$$\lambda f.\lambda x.f(\lambda g.gx) : (((\gamma_1 \rightarrow \gamma_2) \rightarrow \gamma_3) \rightarrow \gamma_4) \rightarrow \gamma_5 \rightarrow \gamma_6.$$

A typical play in the strategy interpreting this term is

$$\begin{tikzcd}[column sep=small] \gamma_6 \cdot & \gamma_4 \arrow[l, bend right] & \gamma_3 \cdot \arrow[l] & \gamma_2 \arrow[l, bend right] & \gamma_1 \cdot \arrow[l] & \gamma_5. \arrow[lllll, bend right] \end{tikzcd}$$

The P-moves correspond to head-variables: γ_4 picks f; γ_2 picks g; and γ_5 picks x. Note also that move γ_5's justifier is before those of γ_1 and γ_3, since x is free in the subterms being investigated, which are x and $\lambda g.gx$ respectively.

3.6 Refining the model

Armed with intuition about the correspondence between strategies and Böhm-trees in the λ-calculus, let us investigate how these ideas might be refined to reflect the $\alpha\lambda$-calculus.

Let $\Gamma \vdash M$ be a Böhm-tree in the λ-calculus. If we replace some of the ; constructors in Γ with , so that Γ becomes a nontrivial bunch, and replace some of the $\rightarrow$ type constructors with $\mathbin{-\!\!*}$, what constraints do the typing rules of the $\alpha\lambda$-calculus place upon M, and how might these be reflected in the games model?

For a term $fM_1 \ldots M_n$ to remain typeable after this refinement of the typing, we require at least that

- if f appears free in one of the M_i, then M_i is an argument to a $\rightarrow$-function, and
- if M_i and M_{i+j} share a free variable x, then M_{i+j} is an argument to a $\rightarrow$-function.

For instance, if f has type $A \mathbin{-\!\!*} B \rightarrow C$, then in a term $(fM) @ N$, f may appear in N but not M; and M and N may share free variables. However, if f has type $A \rightarrow B \mathbin{-\!\!*} C$ then in a term $(f @ M)N$, f may appear free in M but not N, and M and N's free variables must be distinct.

The constraints run deeper than this: in a subterm $fM_1 \ldots M_n$ of a closed Böhm-tree, if M_{i+j} is an argument to a $\mathbin{-\!\!*}$-function, then the free variables of M_i and M_{i+j} must be separated (Lemma 1). What this means for the term is that, if M_i contains x and M_{i+j} contains y:

- whichever of x and y was introduced by abstraction deeper in the term, must have been introduced by λ-abstraction rather than α-abstraction.

To capture all of this in the games model, we will need:

- a semantic account of the notion of "variable appearing free in a subterm";
- a semantic account of the distinction between arguments to $\rightarrow$-functions and $\mathbin{-\!\!*}$-functions;
- a semantic account of the idea that "the deeper abstraction must be a λ-abstraction".

We shall now formalize these ideas in the games setting.

We shall now extend our games model with the appropriate structure to allow us to incorporate the above ideas. In the next few paragraphs, remarks appearing in brackets $[\ldots]$ indicate the intuition corresponding to our definitions.

Given moves a and b in a legal position s, with a occurring before b and $\lambda(a) \neq \lambda(b)$, we say b is an ***external move*** to a, writing $b\,\mathbf{ext}\,a$, if the justifier of b occurs before a (or b is initial), and a is visible at $s_{<b}$. [This gives us a semantic account of variables appearing free in subterms.]

Given an arena A, a ***separation relation*** on A is an irreflexive, symmetric binary relation $\#_A$ on the moves of A, subject to the condition that if $a \#_A b$ then either $a \vdash_A b$, $b \vdash_A a$ or there is some c such that $c \vdash a, b$. That is, separation exists only between moves which enable one another, or which share a justifier. [Separation will allow us to distinguish between $-\!\!*$ and $\to$: the arenas $A -\!\!* B$ and $A \to B$ will differ only in that initial moves from A are separated from those of B in $A -\!\!* B$.]

We refer to an arena A together with a separation relation $\#_A$ as a ***sep-arena***, and often refer to it just as A rather than the pair $(A, \#_A)$.

If s is a legal position in a sep-arena containing two moves a and b, we write $a \#_s b$ if a and b have the same justifier in s (or both are initial) and also $a \#_A b$. [In the case where these are O-moves, $a \#_s b$ tells us that the two subterms being investigated may not share resources.]

We write $a *_s b$ if any of the following conditions holds:

- $a \#_s b$;
- a is justified by a', $b\,\mathbf{ext}\,a'$ and $a \#_A a'$; or
- b is justified by b', $a\,\mathbf{ext}\,b'$ and $b \#_A b'$.

[In the case where these are P-moves, $a *_s b$ tells us that the variables being chosen do not share resources: in a closed term, the more deeply abstracted of the two variables was λ-abstracted.]

Let A be a sep-arena. A legal position s is ***separation safe*** if

- for any O-moves $a, b \in s$ with $a \#_s b$, if $a_1\,\mathbf{ext}\,a$ and $b_1\,\mathbf{ext}\,b$ then $a_1 *_s b_1$,
- for any P-move a such that $a\,\mathbf{ext}\,b$ in s, if b is justified by b' and $b \#_A b'$ then $a *_s b'$.

Example Supposing we give **BI**-types to the example term at the end of Section 3.5. In the example play, we have $\gamma_5\,\mathbf{ext}\,\gamma_1$, and if g is a $-\!\!*$-function, we will have $\gamma_1 \# \gamma_2$, so separation safety will require $\gamma_5 * \gamma_2$. This will hold if $\gamma_2 \# \gamma_3$, which will be the case if f is a $-\!\!*$-function, so that g is λ-abstracted rather than α-abstracted.

Given negative sep-arenas A and B, we can define a separation relation on $A \vdash B$ as the disjoint union of $\#_A$ and $\#_B$. A strategy σ on $A \vdash B$ is called separation safe if every $s \in \sigma$ is separation safe. [Separation safe strategies are those which adhere to the typing constraints of the $\alpha\lambda$-calculus.]

The main technical effort of the paper consists in showing that separation-safety is preserved by composition of strategies. This involves a detailed analysis of interaction sequences.

A move m in an interaction sequence u necessarily belongs to at least one of $u \upharpoonright A, B$ and $u \upharpoonright B, C$. In the case where m is in B, we say that $u \upharpoonright A, B$ is the **P-*component*** of m if m is a P-move there; otherwise it is the O-component. Similarly for $u \upharpoonright B, C$.

The view $\mathsf{view}(u)$ of an interaction sequence is defined exactly as the view of an ordinary legal position. We note that $\mathsf{view}(u)$ is the same as the view of the O-component of the last move of u. This is because the justifier of a B-move which is an O-move in A, B must be a P-move in A, B, and the move immediately preceding it in u must again be an O-move coming from A, B.

We say that moves m and n in $\mathsf{view}(u)$ have opposite polarity if there is an even number of moves strictly between them in $\mathsf{view}(u)$.

Given an interaction sequence u containing moves m and n, with m occurring before n, we say n is an external move to m, $n \, \mathbf{ext}_u \, m$, if m is visible at $u_{<n}$, m and n have opposite polarity and n is initial or has its justifier before m. This is equivalent to saying that $n \, \mathbf{ext} \, m$ in the P-component of n.

The following lemma relates the $\mathbf{ext}$ relation in $u \restriction A, C$ to those in the A, B and B, C components.

Lemma 4. *Let $u \in \mathsf{int}(A, B, C)$. Let n be a P-move in $u \restriction A, C$ and m an O-move in $u \restriction A, C$ such that $n \, \mathbf{ext} \, m$ in $u \restriction A, C$. Then there exist moves $n_1, \ldots, n_k$ in u, all coming from B, such that $n \, \mathbf{ext}_u \, n_1 \, \mathbf{ext}_u \cdots \mathbf{ext}_u \, n_k \, \mathbf{ext} \, m$.*

We omit the proof, which involves a careful analysis of views in interaction sequences, similar to the proof that innocence is preserved by composition of strategies given in [6].

Having lifted the definition of $\mathbf{ext}$ to interaction sequences, the definitions of # and $*$ on legal positions lift directly to interaction sequences, and we have $m * n$ in u iff $m * n$ in the P-component of m.

The following lemma is key to our proof that separation-safety is preserved by composition.

Lemma 5. *Let A, B and C be negative sep-arenas. Let $u \in \mathsf{int}(A, B, C)$ be such that $u \restriction A, B$ and $u \restriction B, C$ are separation-safe. Let m_1 and m_2 be moves in u such that either m_1 and m_2 are O-moves in $u \restriction A, C$ with $m_1 \# m_2$, or they are P-moves in some component with $m_1 * m_2$. Suppose there are moves $n_{1,1}, \ldots, n_{1,k_1}$, $n_{2,1}, \ldots, n_{2,k_2}$ such that*

- *$n_{i,1} \, \mathbf{ext} \, n_{i,2} \, \mathbf{ext} \cdots \mathbf{ext} \, n_{i,k_i} \, \mathbf{ext} \, m_i$, for $i = 1, 2$,*
- *$n_{1,1}$ and $n_{2,1}$ are P-moves in $u \restriction A, C$,*
- *the intervening moves $n_{i,2}, \ldots, n_{i,k_i}$ are in B,*
- *the justifier of $n_{1,1}$ is visible at $u_{<n_{2,1}} \restriction A, C$ and vice versa.*

Then $n_{1,1} * n_{2,1}$ in $u \restriction A, C$.

Proof We require an auxiliary definition: the ***pivot*** of separation of $m_1 * m_2$. According to the definition of $*$, there are four possibilities:

- m_1 and m_2 are both initial and $m_1 \# m_2$: then there is no pivot;
- m_1 and m_2 have the same justifier j and $m_1 \# m_2$: then j is the pivot;
- m_1 is justified by some j, $m_1 \# j$ and $m_2 \, \mathbf{ext} \, j$: then j is the pivot;
- as above with m_1 and m_2 exchanged: again j is the pivot.

We approach our proof by induction on the length of $u_{\leq j}$, where j is the pivot of $m_1 * m_2$; by convention this is zero if there is no pivot.

The base case, therefore, is the case in which the m_i are initial and $m_1 \# m_2$. If they are initial C-moves, the moves n_{i,k_i} must be initial B-moves and by separation safety of $u \restriction B, C$, $n_{1,k_1} * n_{2,k_2}$, which is to say $n_{1,k_1} \# n_{2,k_2}$. The only external moves of initial B-moves are initial A-moves, so we have that n_{1,k_1-1} and n_{2,k_2-1} are initial A-moves and $n_{1,k_1-1} * n_{2,k_2-1}$ by separation safety in $u \restriction A, B$. It must be the case that $n_{1,1} = n_{1,k_1-1}$ and $n_{2,1} = n_{2,k_2-1}$ completing the argument in this case.

If the m_i are initial B-moves, the second half of the above argument applies.

The inductive step considers the cases where a pivot j exists.

Suppose first that j is the shared justifier of m_1 and m_2, and $m_1 \# m_2$. We first deal with the degenerate case in which one of the k_i (wlog k_1) is zero, i.e., $n_{1,1} = m_1$. In this case the m_i are both P-moves in A or C; suppose wlog they are in A. But then it must also be that $k_2 = 0$ since no B-move can be an external of m_2 by definition. Thus we need only show that $m_1 * m_2$ which is true by hypothesis.

If both k_1 and k_2 are non-zero, we have $n_{i,k_i} \,\mathbf{ext}\, m_i$ for $i = 1, 2$ and $m_1 \# m_2$ in $u \restriction A, B$ or $u \restriction B, C$. Hence by separation safety in this component, we have $n_{1,k_1} * n_{2,k_2}$, with pivot appearing earlier in u than j. We can therefore apply the inductive hypothesis to conclude.

Suppose finally that the pivot j is the justifier of m_1, where $m_1 \# j$ and $m_2 \,\mathbf{ext}\, j$. (The case with the m_i exchanged is handled symmetrically.) If $k_1 = 0$, then j is in the view at $u_{<n_{2,1}} \restriction A, C$ and $n_{2,1}$'s justifier appears before that of m_2 and hence before j, so $n_{2,1} \,\mathbf{ext}\, j$. Then by definition, $n_{2,1} * m_1 = n_{1,1}$. Otherwise, $k_1 \neq 0$ so we have $n_{1,k_1} \,\mathbf{ext}\, m_1$, and separation safety in the P-component of n_{1,k_1} ensures that $n_{1,k_1} * j$. The pivot for this separation must appear before j, so we can apply the inductive hypothesis to this pair to complete the proof. □

Given negative sep-arenas A and B, the sep-arenas $A \times B$ and $A * B$ are defined as follows. Both have the same underlying arena as $A \times B$ defined for ordinary arenas; the difference is in the separation relations:

$$\begin{aligned} m \#_{A \times B} n &\iff m \#_A n \vee m \#_B n \\ m \#_{A * B} n &\iff m \#_A n \vee m \#_B n \vee [\star \vdash_A m \wedge \star \vdash_B n] \end{aligned}$$

plus a clause symmetric in m and n. That is, in $A * B$, the initial moves of A are separated from those of B, which is not the case in $A \times B$.

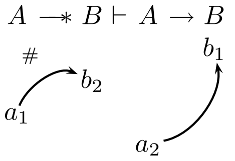
We can also define $A \to B$ and $A {-\!\!*} B$ on negative sep-arenas. The underlying arenas are as for $A \Rightarrow B$. The separation relations are defined as follows:

$$\begin{aligned} m \#_{A \to B} n &\iff m \#_A n \vee m \#_B n \vee [\star \vdash_A m \wedge \star \vdash_B b \vdash_B n \wedge n \#_B b] \\ m \#_{A {-\!*} B} n &\iff m \#_A n \vee m \#_B n \vee [\star \vdash_A m \wedge \star \vdash_B n] \\ &\quad \vee \quad \star \vdash_A m \wedge \star \vdash_B b \vdash_B n \wedge n \#_B b \end{aligned}$$

plus clauses symmetric in m and n. That is, in both $A \to B$ and $A {-\!\!*} B$, initial A moves are separated from moves of B which are enabled by and separated from initial moves. Furthermore in $A {-\!\!*} B$, initial A-moves are separated from the initial B-moves that enable them. Thus for example in $A \to (B {-\!\!*} C)$, initial A-moves are separated from initial B-moves.

To illustrate the impact of separation safety, consider the sep-arenas $A \times B$ and $A * B$. The identity strategy on the underlying arena is separation-safe on $A * B \vdash A \times B$, but not on $A \times B \vdash A * B$. In the latter case, the initial moves on the right are related by # but are copied to the initial moves on the left, which are not. Since the initial moves on the left are external to those on the right, this violates separation safety.

Similarly, the identity strategy on the underlying arena gives us a valid strategy on $A \to B \vdash A \multimap\!\!* B$ but not the other way around: to see why consider a typical play



Here we have $a_2 \,\mathbf{ext}\, a_1$, and a_1 is justified by b_2 with $a_1 \# b_2$, so separation safety would require $a_2 * b_2$ which is not the case here. On the other hand if we had $a_2 \# b_1$, which we would if the $\to$ were $\multimap\!\!*$, then indeed $a_2 * b_2$.

Proposition 1. *The identity strategy on a sep-arena is separation safe.*

Proof Any P-move played by the identity strategy is external only to the immediately preceding O-move, of which it is a copy. Thus if a_1 **ext** b_1 and a_2 **ext** b_2 with $b_1 \# b_2$, we immediately have a_1 **ext** a_2. Similarly if a_1 **ext** b_1 and b_1 is justified by some b_2 with $b_1 \# b_2$, then a_1 is a copy of b_1 and hence is justified by an a_2 with $a_1 \# a_2$. It must also be the case that b_2 is the copy of a_2, so b_2 **ext** a_2 and hence $b_2 * a_1$ as required. □

Proposition 2. *The composition of separation-safe innocent strategies is itself a separation-safe strategy.*

Proof If $\sigma : A \vdash B$ and $\tau : B \vdash C$ are separation-safe strategies and $s \in \sigma ; \tau$ then there exists some $u \in \mathsf{int}(A, B, C)$ such that $u \restriction A, B \in \sigma$, $u \restriction B, C \in \tau$ and $u \restriction A, C = s$.

Suppose $m_1 \#_s m_2$ and $n_i \,\mathbf{ext}_s\, m_i$ for $i = 1, 2$. By Lemma 4, there are moves $n_{i,1}, \ldots, n_{i,k_i}$ in u coming from B such that n_i **ext** $n_{i,1}$ **ext** $\cdots$ **ext** n_{i,k_i} **ext** m_i. Note also that, since m_i is visible at n_i and m_1 and m_2 share a justifier j, the justifier of each n_i appears in the view $\mathsf{view}(s_{<j})$. Thus n_1's justifier is visible at $s_{<n_2}$ and vice versa. Thus by Lemma 5, $n_1 * n_2$ as required.

Finally suppose m is justified by j with $m \# j$, and some $n \,\mathbf{ext}_s\, m$. Since m is in $\mathsf{view}(s_{<n})$, by the visibility condition the justifier of j is in $\mathsf{view}(s_{<n})$ and the justifier of n is in $\mathsf{view}(s_{<j})$. Again by Lemma 4 we have a sequence of B-moves in u such that $n \,\mathbf{ext}\, n_1 \,\mathbf{ext} \cdots \mathbf{ext}\, n_k \,\mathbf{ext}\, m$. By separation safety of σ and τ we obtain $n_k * j$, and then Lemma 5 tells us that $n * j$ as required. □

We can now define a category $\mathcal{G}_{\text{sep}}$ with negative sep-arenas as objects and separation safe strategies on $A \vdash B$ as maps from A to B.

3.7 Interpreting the $\alpha\lambda$-calculus

In order to give a semantics of the $\alpha\lambda$-calculus, we must show that $\mathcal{G}_{\mathrm{sep}}$ has enough structure to interpret both ; and , in contexts, and both $\rightarrow$ and $-\!\!*$ in types.

Proposition 3. *The category $\mathcal{G}_{\mathrm{sep}}$ is cartesian closed, with product given by $\times$ and exponential by $\rightarrow$. A second symmetric monoidal structure is provided by $*$ and, if B is a sep-arena with only one initial move and A any sep-arena, then $A -\!\!* B$ is an exponential with respect to $*$.*

Proof It is not difficult to check that $\times$ is categorical product, just as the corresponding construct is on $\mathcal{G}_{\mathrm{inn}}$. Similarly it is easy to check that $*$ is a monoidal structure: all the structural isomorphisms are given by copycat strategies as usual, and one just has to verify that they are separation safe. The exponentials are where the novelty lies.

In $\mathcal{G}_{\mathrm{inn}}$, the fact that $\Rightarrow$ is the exponential with respect to $\times$ follows from the fact that there is a clear 1-1 correspondence between the legal positions of $A \times B \vdash C$ and $A \vdash B \Rightarrow C$: both consist of moves from A, B and C, and to turn a play in the former arena into one in the latter, one simply alters the justification structure so that initial B-moves are justified by the unique visible initial C-move. This lifts to a natural isomorphism of strategies and is standard in game semantics. We shall argue that the same isomporphism serves for the two exponentials in $\mathcal{G}_{\mathrm{sep}}$.

Our only additional obligation is to show that separation safety is preserved when moving across this isomorphism. Let a_0, b_0, c_0 range over initial moves of A, B and C respectively, and c_1 over moves of C such that some $c_0 \vdash_C c_1$ with $c_0 \#_C c_1$. The only differences between the separation relations on $A \times B \vdash C$ and $A \vdash B \rightarrow C$ is that in the latter, $b_0 \# c_1$ which is not the case in the former. But if s is a position of $A \vdash B \rightarrow C$ containing b_0 and c_1 with the same justifier c_0, so that $b_0 \#_s c_1$, the corresponding position of $A \times B \vdash C$ has b_o **ext** c_0 and hence $b_0 * c_1$. A similar argument shows that all instances of $*$ in positions of $A \vdash B \Rightarrow C$ are retained when moving to $A \times B \vdash C$, so separation safety is preserved by the isomorphism of positions.

For the correspondence between $A * B \vdash C$ and $A \vdash B -\!\!* C$ there are two differences in $\#$: first, as above, $b_0 \# c_1$ in the latter but not the former, but this is handled just as above; second, $a_o \# b_0$ in the former but not the latter. For the second, observe that in any position of $A * B \vdash C$ in which $a_0 \# b_0$ is required for separation safety, the initial move c_0 in $\mathsf{view}(s_{<a_0})$ is the same as that in $\mathsf{view}(s_{<b_0})$; this is because C has only one initial move by hypothesis. Therefore when moving across the isomorphism we have a_0 **ext** c_0 and b_0 justified by c_0 with $b_0 \# c_0$, hence $a_0 * b_0$. A similar analysis shows that moving in the other direction across the isomorphism also preserves separation safety. □

The category $\mathcal{G}_{\mathrm{sep}}$ is therefore *almost* a cartesian doubly-closed category, the notion of categorical model for the $\alpha\lambda$-calculus defined in [9–11]. $\mathcal{G}_{\mathrm{sep}}$ lacks certain exponentials, such as those of the form $A -\!\!* (B * C)$, whose right-hand side arena has multiple initial moves. However, those are not necessary for the interpretation of the $\alpha\lambda$-calculus, so we can define the semantics of the language exactly as in [10] using just the structure we have. We briefly sketch the interpretation below; there are no surprises.

Types are interpreted as sep-arenas:

- $[\![\gamma]\!]$ is the arena with a single, initial opponent move γ and empty separation relation;
- $[\![A \multimap\!\!* B]\!] = [\![A]\!] \multimap\!\!* [\![B]\!]$ and $[\![A \to B]\!] = [\![A]\!] \to [\![B]\!]$.

Bunches are also interpreted as sep-arenas:

$$[\![I]\!] = I \qquad [\![\Gamma ; \Delta]\!] = [\![\Gamma]\!] \times [\![\Delta]\!] \qquad [\![\Gamma, \Delta]\!] = [\![\Gamma]\!] * [\![\Delta]\!].$$

The coherence of symmetric monoidal categories ensures that if $\Gamma \equiv \Delta$ there is a canonical isomorphism between $[\![\Gamma]\!]$ and $[\![\Delta]\!]$. This allows us to interpret the coherence rule. If $\Gamma \cong \Delta$ then $[\![\Gamma]\!] = [\![\Delta]\!]$, so the diagonal map $\Gamma \to \Gamma \times \Gamma$ can be used to interpret contraction. Since I is terminal and is the unit for both $\times$ and $*$, there is a canonical projection from any $[\![\Gamma(\Delta;\Delta')]\!]$ or $[\![\Gamma(\Delta,\Delta')]\!]$ to $[\![\Gamma(\Delta)]\!]$ which can be used to interpret weakening. Finally, the two abstraction and application rules are interpreted using the currying isomorphisms and the evaluation maps of the two exponentials.

4 Definability

Theorem 1. *Let A be a type of the $\alpha\lambda$-calculus and Γ a bunch, both built over a single ground type γ. Let σ be a finite, total separation-safe innocent strategy on $[\![\Gamma]\!] \vdash [\![A]\!]$. Then there is a term $\Gamma \vdash M : A$ such that $[\![M]\!] = \sigma$.*

Proof Forgetting the separation relations, we can treat σ as a strategy on the underlying arena and using t he argument for full completeness in the λ-calculus given in Section 3.5, find a term M such that $[\![M]\!] = \sigma$ in the model of ordinary λ-calculus. We must show that this term is typeable in the $\alpha\lambda$-calculus.

It is straightforward to establish that, if we start with a separation-safe strategy σ, the substrategies which give rise to argument terms M_i are themselves separation safe. Thus the inductive hypothesis gives us terms $\Gamma \vdash M_i : A_i$ and a candidate application

$$\Gamma \vdash x_i \bullet M_1 \bullet \cdots \bullet M_k$$

where each $\bullet$ may be a linear or additive application, according to the type of x_i. We must show that this application is typeable in the $\alpha\lambda$-calculus. We proceed by induction on the number of applications. The base case, that of a variable by itself, is trivial. For the inductive step, by inductive hypothesis we have $\Gamma \vdash M_k$ and

$$\Gamma \vdash (x_i \bullet M_1 \bullet \cdots \bullet M_{k-1}).$$

If the final application is additive, the result follows from Lemma 2. In the multiplicative case, by Lemma 1 it suffices to show that the free identifiers of M_k are separated from those of x_i, $M_1, \ldots, M_{k-1}$. In the terminology of Section 3.5, each free identifier of M_k corresponds to a move m in σ such that $m\,\mathbf{ext}\,\gamma_{i_k}$. The

x_i corresponds to the move γ_i, while the free identifiers of each M_j corresponds to a move n such that $n\,\mathbf{ext}\,\gamma_{i,j}$. By definition, each $\gamma_{i,j} \# \gamma_{i,k}$ and $\gamma_{i,k} \# \gamma_i$, so by separation safety, $m * n$ and $m * \gamma_i$. All these moves are initial moves in $[\![\Gamma]\!]$, so $*$ can only arise from $\#$. Hence the free identifiers of M_k are separated from those of the M_j and x_i as required. □

References

1. S. Abramsky, R. Jagadeesan, and P. Malacaria. Full abstraction for PCF. *Information and Computation*, 162(2):409–470, 2000.
2. S. Abramsky and G. McCusker. Linearity, sharing and state: a fully abstract game semantics for Idealized Algol with active expressions (extended abstract). In *Proceedings of 1996 Workshop on Linear Logic*, volume 3 of *Electronic notes in Theoretical Computer Science*. Elsevier, 1996.
3. R. Atkey. A λ-calculus for resource separation. In *Proceedings, 31st International Colloquium on Automata, Languages and Programming (ICALP 2004)*, volume 3142 of *Lecture Notes in Computer Science*, pages 158–170. Springer-Verlag, 2004.
4. J. M. E. Hyland and C.-H. L. Ong. On full abstraction for PCF: I, II and III. *Information and Computation*, 162(2):285–408, 2000.
5. J. Lambek and P. J. Scott. *Introduction to Higher Order Categorical Logic*. Cambridge University Press, 1986.
6. G. McCusker. *Games and Full Abstraction for a Functional Metalanguage with Recursive Types*. Distinguished Dissertations in Computer Science. Springer-Verlag, 1998.
7. H. Nickau. Hereditarily sequential functionals. In *Proceedings of the Symposium on Logical Foundations of Computer Science: Logic at St. Petersburg*, Lecture notes in Computer Science. Springer, 1994.
8. P. W. O'Hearn. Resource interpretations, bunched implications and the $\alpha - \lambda$-calculus. In J.-Y. Girard, editor, *Proceedings, Typed Lambda-Calculi and Applications, L'Aquila, Italy, April 1999*, volume 1581 of *LNCS*, pages 258–279. Springer-Verlag, 1999.
9. P. W. O'Hearn and D. J. Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2):215–244, June 1999.
10. P. W. O'Hearn. On bunched typing. *Journal of Functional Programming*, 13(4):747–796, July 2003.
11. D. J. Pym. *The Semantics and Proof Theory of the Logic of Bunched Implications*. Kluwer Academic, 2002. Errata available at `http://www.cs.bath.ac.uk/~pym/BI-monograph-errata.pdf`.
12. D. J. Pym and C. Tofts. Systems modelling via resources and processes: Philosophy, calculus, semantics, and logic. *Electronic Notes in Theoretical Computer Science*, 172:545–587, 2007. Errata available at `http://www.cs.bath.ac.uk/~pym/pym-tofts-fac-errata.pdf`.
13. J. C. Reynolds. Syntactic control of interference. In *Conf. Record 5th ACM Symposium on Principles of Programming Languages*, pages 39–46, 1978.
14. J. C. Reynolds. Separation logic: a logic for shared mutable data structures. In *Seventeenth Annual IEEE Symposium on Logic in Computer Science (LICS 2002)*. IEEE Computer Society Press, 2002. Invited paper.
15. M. Wall. *Games for Syntactic Control of Interference*. PhD thesis, University of Sussex, 2005.