1 A Fully Abstract Game Semantics for Idealized Algol

To introduce our material, we will go back over some old ground, namely the fully abstract game semantics for Idealized Algol developed by Abramsky and McCusker in [AM96]. In keeping with the spirit of this thesis, we will aim to use category theoretic methods, and so our proofs of soundness and adequacy will depart from those given by Abramsky and McCusker, and will instead involve coalgebraic ideas developed by Laird in [Lai02] and [Lai].

1.1 Idealized Algol

The ground types of Idealized Algol are called `com`, `bool`, `nat` and `Var`. The first three are data types corresponding to the sets $\mathbb{C} = \{a\}$, $\mathbb{B} = \{\mathtt{t}, \mathtt{f}\}$ and $\mathbb{N} = \{0, 1, 2, \cdots\}$. `com` takes the role of a command or void type; typically, although the return value of a function $T \to \mathtt{com}$ will not convey any information, the function will have useful side effects.

The type `Var` is the type of a variable that holds elements of $\mathbb{N}$. It is best understood as corresponding to the following pseudo-Java 'interface'.

```
public interface Var
{
  nat read();
  com write(nat value);
}
```

We now present the typing rules for the language. Here, Γ will represent a *context*; i.e., a list $x_1 \colon T_1, \cdots, x_n \colon T_n$ of variable names together with their types.

First, we have the usual rules for the simply typed lambda calculus.

$$\frac{}{\Gamma, x \colon T \vdash x \colon T} \qquad \frac{\Gamma \vdash M \colon S \to T \qquad \Gamma \vdash N \colon S}{\Gamma \vdash MN \colon T} \qquad \frac{\Gamma, x \colon S \vdash M \colon T}{\Gamma \vdash \lambda x^S . M \colon S \to T}$$

We then have rules for each of the base types. At type `com` we have:

$$\frac{}{\Gamma \vdash \mathsf{skip} \colon \mathtt{com}} \qquad \frac{\Gamma \vdash M \colon \mathtt{com} \qquad \Gamma \vdash N \colon T}{\Gamma \vdash M; N \colon T}\, T \in \{\mathtt{com}, \mathtt{bool}, \mathtt{nat}\} \quad .$$

Here, `skip` is a generic command with no side-effects that returns the unique element of the singleton set $\mathbb{C}$. $M; N$ represents the sequential composition of M with N; i.e., the term that first evaluates M, performing any of its side-effects, and then evaluates N and returns the result.

At type `bool` we have true/false values and conditionals.

$$\frac{}{\Gamma \vdash \mathbb{t} \colon \texttt{bool}} \qquad \frac{}{\Gamma \vdash \mathbb{f} \colon \texttt{bool}}$$

$$\frac{\Gamma \vdash M \colon \texttt{bool} \qquad \Gamma \vdash N \colon T \qquad \Gamma \vdash P \colon T}{\Gamma \vdash \mathsf{If}\, M \,\mathsf{then}\, N \,\mathsf{else}\, P \colon T}\; T \in \{\texttt{com}, \texttt{bool}, \texttt{nat}\}$$

At type `nat` we have numerals, arithmetic operators and a conditional that tests whether a number is equal to 0 or not.

$$\frac{}{\Gamma \vdash n \colon \texttt{nat}} \qquad \frac{\Gamma \vdash M \colon \texttt{nat}}{\Gamma \vdash \mathsf{succ}\, M \colon \texttt{nat}} \qquad \frac{\Gamma \vdash M \colon \texttt{nat}}{\Gamma \vdash \mathsf{pred}\, M \colon \texttt{nat}}$$

$$\frac{\Gamma \vdash M \colon \texttt{nat} \qquad \Gamma \vdash N \colon T \qquad \Gamma \vdash P \colon T}{\Gamma \vdash \mathsf{If0}\, M \,\mathsf{then}\, N \,\mathsf{else}\, P \colon T}\; T \in \{\texttt{com}, \texttt{bool}, \texttt{nat}\}$$

At type `Var`, we have terms that call the read and write ‘methods’ to dereference the variable or to assign a new value to it.

$$\frac{\Gamma \vdash V \colon \texttt{Var}}{\Gamma \vdash !V \colon \texttt{nat}} \qquad \frac{\Gamma \vdash V \colon \texttt{Var} \qquad \Gamma \vdash E \colon \texttt{nat}}{\Gamma \vdash V \leftarrow E \colon \texttt{com}}$$

We also have the ability to create a new variable.

$$\frac{\Gamma, x \colon \texttt{Var} \vdash M \colon T}{\Gamma \vdash \mathsf{new}_T\, \lambda x.M \colon T}$$

The idea here is that M is a term that refers to some free variable x of type `Var`; then $\mathsf{new}\, \lambda x.M$ makes x behave like an actual storage cell (so, for instance, the result of the computation $\mathsf{new}_{\texttt{nat}}\, \lambda x.(x \leftarrow 5); !x$ will be 5).

We have another way of creating variables, using the mkvar keyword. If we think back to our illustration of the `Var` type as an interface, this becomes clearer. mkvar creates a new anonymous instance of the `Var` interface, using the ‘methods’ supplied through its arguments.

$$\frac{\Gamma \vdash M \colon \texttt{nat} \qquad \Gamma \vdash N \colon \texttt{nat} \to \texttt{com}}{\Gamma \vdash \mathsf{mkvar}\, MN \colon \texttt{Var}}$$

Lastly, we have fixpoint combinators at all types that we use to implement recursion.

$$\frac{\Gamma \vdash M \colon T \to T}{\Gamma \vdash \mathbf{Y}_T M \colon T}$$

1.2 Games and Strategies

We adopt the game semantics from [AM96]; these are based on the game semantics developed in [HO00], with a modification to make them into a linear category.

Definition 1.1. An *arena* is a tuple $A = (M_A, \lambda_A, \vdash_A)$, where

- M_A is a set of *moves*,
- $\lambda_A \colon M_A \to \{O, P\} \times \{Q, A\}$ is a function that identifies each move as either an *O-move* or a *P-move*, and as either a *question* or an *answer*, and
- $\vdash_A$ is a relation between $M_A + \{*\}$ and M_A such that
  - if $* \vdash_A a$, then $\lambda_A(a) = (O, Q)$, and if $b \vdash_A a$ then $b = *$,
  - if $a \vdash_A b$ and a is an answer, then b is a question, and
  - if $a \vdash_A b$ and $a \neq *$, then either a is an O-move and b a P-move, or the other way round.

If $* \vdash_A a$, then we say that a is an *initial move* in A. If $a \vdash_A b$, the we say that a *enables* b.

As a shorthand, we write $\lambda_A^{OP} \colon M_A \to \{O, P\}$ for $\mathrm{pr}_1 \circ \lambda_A$ and $\lambda_A^{QA} \colon M_A \to \{Q, A\}$ for $\mathrm{pr}_2 \circ \lambda_A$.

Definition 1.2. A *justified sequence* in an arena A is a finite sequence s of moves together with, for each non-initial move a occurring in s, a pointer back to some move b occurring earlier in s such that $b \vdash_A a$. We say that b *justifies* a or that b is the *justifier* of a.

Given such a justified sequence, we define the *P-view* $\ulcorner s \urcorner$ and *O-view* $\llcorner s \lrcorner$ of s inductively as follows.

$$\begin{aligned}
\ulcorner \epsilon \urcorner &= \epsilon \\
\ulcorner sa \urcorner &= \ulcorner s \urcorner a && \text{if } a \text{ is a } P\text{-move} \\
\ulcorner sa \urcorner &= a && \text{if } a \text{ is initial} \\
\ulcorner sbta \urcorner &= \ulcorner s \urcorner ba && \text{if } a \text{ is an } O\text{-move justified by } b
\end{aligned}$$

$$\begin{aligned}
\llcorner \epsilon \lrcorner &= \epsilon \\
\llcorner sa \lrcorner &= \llcorner s \lrcorner a && \text{if } a \text{ is an } O\text{-move} \\
\llcorner sbta \lrcorner &= \llcorner s \lrcorner ba && \text{if } a \text{ is a } P\text{-move justified by } b
\end{aligned}$$

A justified sequence s is *well-bracketed* if whenever a question q justifies some answer a, then any question q' occurring after q and before a must justify some answer a' occurring between q' and a, and moreover a is the only answer justified by q. We say that a justified sequence s is *alternating* if it alternates between O-moves and P-moves, and that it is *well-formed* if it is both well-bracketed and alternating.

We say that a well-formed justified sequence is *visible* if whenever $ta \sqsubseteq s$, and a is a P-move, then the justifier of a occurs in the P-view of t, and if whenever $tb \sqsubseteq s$, and b is a non-initial O-move, then the justifier of b occurs in the O-view of t.

We say that a justified sequence s is *legal* if it is well-formed and visible, and write L_A for the set of legal sequences occurring in A.

Note that since every non-initial move in a justified sequence s must be justified by some previous move, then the first move in the sequence must be initial and therefore an O-question. If s is alternating, this means that s ends with an O-move if it has odd length and with a P-move if it has even length.

Definition 1.3. Given a legal sequence $s \in L_A$, and a move b in s, we say that a move a in s is *hereditarily justified by* b if there is a chain of justification pointers going back from a to b.

We write $s|_b$ for the subsequence of s given by all moves in s that are hereditarily justified by b. Given a set I of initial moves, we write $s|_I$ for the subsequence of s given by all moves that are hereditarily justified by some $b \in I$.

A *game* is given by a tuple $A = (M_A, \lambda_A, \vdash_A, P_A)$ where $(M_A, \lambda_A, \vdash_A)$ is an arena and P_A is a non-empty prefix-closed subset of L_A such that if $s \in P_A$ and I is a set of initial moves, then $s|_I \in P_A$.

We shall call an odd-length sequence $s \in P_A$ an *O-position* and an even-length sequence a *P-position*

Example 1.4 (Empty game). The *empty game* I is given by the tuple

$$(\varnothing, \varnothing, \varnothing, \{\epsilon\}) \, ,$$

where ϵ is the empty sequence.

Example 1.5 (Data-type games). Let X be some set. Then we have a game, which we shall also call X, given by:

- $M_X = \{q\} + X$,
- $\lambda_X(q) = (O, Q)$ and $\lambda_X(x) = (P, A)$ for all $x \in X$,
- $q \vdash_X x$ for each $x \in X$, and
- $P_X = \{\epsilon, q\} \cup \{qx : \; x \in X\}$, where the x in qx is justified by q.

In particular, we have games $\mathbb{C}$, $\mathbb{B}$ and $\mathbb{N}$, which we shall use to model the datatypes `com`, `bool` and `nat` of Idealized Algol.

Definition 1.6. Let A be a game. Then a *strategy* for A is a non-empty even-prefix-closed set $\sigma \subseteq P_A$ of P-positions in A such that if $sab, sac \in \sigma$ then $b = c$ and the justifier of b is the justifier of c.

Here, we have identified a strategy for a game with the set of P-positions that can occur when player P plays according to that strategy. So the condition we have given is one of *determinism*: in any O-position sa that can occur in the strategy, player P must have at most one reply.

Note that there may be O-positions for which player P has no reply at all; we use these to model non-terminating computations.

We write $\sigma \colon A$ to denote that σ is a strategy for the game A.

Definition 1.7. A strategy σ for a game A is called *innocent* if player P's moves only depend on the current P-view; i.e., if whenever $sab \in \sigma$, $t \in \sigma$ and $ta \in P_A$ such that $\ulcorner sa \urcorner = \ulcorner ta \urcorner$, then we have $tab \in \sigma$.

1.3 Connectives on Games

In the *product* $A \times B$ of games A and B, player O chooses either A or B on the first move and subsequent play is that game.

Definition 1.8. Given games A,B, define a game $A \times B$ by

- $M_{A\times B} = M_A + M_B$,
- $\lambda_{A\times B} = [\lambda_A, \lambda_B]$,
- $* \vdash_{A\times B} a$ if and only if $* \vdash_A a$ or $* \vdash_B a$ and $a \vdash_{A\times B} b$ if and only if $a \vdash_A b$ or $a \vdash_B b$, and
- $P_{A\times B} = \{s \in L_{A\times B} \;:\; s|_A \in P_A \text{ and } s|_B = \epsilon \text{ or } s|_A = \epsilon \text{ and } s|_B \in P_B\}$.

Here, we have written $s|_A$ for the subsequence of s consisting of all moves from M_A and $s|_B$ for the subsequence consisting of all moves from M_B.

In the *tensor product* $A \otimes B$ of games A and B, the games A and B are played in parallel, and player O may switch between games when it is his turn.

Definition 1.9. Given games A,B, define a game $A \otimes B$ by

- $M_{A\otimes B} = M_A + M_B$,
- $\lambda_{A\otimes B} = [\lambda_A, \lambda_B]$,
- $* \vdash_{A\otimes B} a$ if and only if $* \vdash_A a$ or $* \vdash_B a$ and $a \vdash_{A\otimes B} b$ if and only if $a \vdash_A b$ or $a \vdash_B b$, and
- $P_{A\otimes B} = \{s \in L_{A\otimes B} \;:\; s|_A \in P_A \text{ and } s|_B \in P_B\}$.

In the *linear implication* $A \multimap B$, the game B is played in parallel with a version of A in which the two players' roles have been switched around, and player P may switch between the two games when it is her turn.

Definition 1.10. Given games A,B, define a game $A \multimap B$ by

- $M_{A\multimap B} = M_A + M_B$,
- $\lambda_{A\multimap B} = [\neg \circ \lambda_A, \lambda_B]$,
- $* \vdash_{A\multimap B} a$ if and only if $* \vdash_B a$, and $a \vdash_{A\multimap B} b$ if and only if $a \vdash_A b$ or $a \vdash_B b$, or if a is initial in B and b is initial in a, and
- $P_{A\multimap B} = \{s \in L_{A\multimap B} \ : \ s|_A \in P_A \text{ and } s|_B \in P_B\}$.

Here, $\neg \colon \{O, P\} \times \{Q, A\} \to \{O, P\} \times \{Q, A\}$ is the function that reverses O and P, while leaving $\{Q, A\}$ unchanged.

In the *exponential* of a game A, infinitely many copies of A are played in parallel, and player O may switch between copies whenever it is his move.

Definition 1.11. Given a game A, define a game $!A$ by

- $M_{!A} = M_A$,
- $\lambda_{!A} = \lambda_A$,
- $\vdash_{!A} = \vdash_A$ and
- $P_{!A} = \{s \in L_{!A} \ : \ s|_b \in P_A \text{ for each initial move } b \text{ occurring in } s\}$.

Lastly, the *sequoid* $A \oslash B$ of two games A and B behaves like the tensor product $A \otimes B$, except that the opening move must take place in A.

Definition 1.12. Given games A, B, define a game $A \oslash B$ by

- $M_{A\oslash B} = M_{A\otimes B}$,
- $\lambda_{A\oslash B} = \lambda_{A\otimes B}$,
- $\vdash_{A\oslash B} = \vdash_{A\otimes B}$ and
- $P_{A\oslash B} = \{s \in P_{A\otimes B} \ : \ s = \epsilon \text{ or } s \text{ begins with a move from } A\}$.

1.4 Composition of strategies

Definition 1.13. Let A, B, C be arenas. An *interaction sequence* between A, B, C is a justified sequence $\mathfrak{s}$ of moves drawn from M_A, M_B and M_C such that $\mathfrak{s}|_{A,B} \in L_{A\multimap B}$ and $\mathfrak{s}|_{B,C} \in L_{B\multimap C}$. Here, $\mathfrak{s}|_{A,B}$ is the subsequence of $\mathfrak{s}$ consisting of those moves from $\mathfrak{s}$ that occur in A or B, together with all justification pointers between moves in A and B, and $\mathfrak{s}|_{B,C}$ is defined similarly.

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

Given $\mathfrak{s} \in \mathrm{int}(A, B, C)$, we write $\mathfrak{s}|_{A,C}$ for the subsequence of $\mathfrak{s}$ consisting of those moves from $\mathfrak{s}$ that occur in A or B. A move b in $\mathfrak{s}|_{A,C}$ justifies a move a either if b justifies a in either the A or the C components, or if b justifies in $\mathfrak{s}$ some initial move c in B, which itself justifies a.

Definition 1.14. Let A, B, C be games, let σ be a strategy for $A \multimap B$ and let τ be a strategy for $B \multimap C$. We define $\sigma \| \tau$ to be given by the set

$$\{\mathfrak{s} \in \mathrm{int}(A, B, C) \,:\, \mathfrak{s}|_{A,B} \in \sigma \text{ and } \mathfrak{s}|_{B,C} \in \tau\}\,.$$

Then we define the *composition* $\sigma;\tau$ of σ and τ to be given by the set

$$\{\mathfrak{s}|_{A,C} \,:\, \mathfrak{s} \in \sigma \| \tau\}\,.$$

We need some small lemmata and definitions to help us show that this is a strategy.

Lemma 1.15. *We extend the function λ_A^{OP} to sequences of moves by*

- $\lambda_A^{OP}(\epsilon) = P$ *and*
- $\lambda_A^{OP}(sa) = \lambda_A(a)$.

If $s \in P_{A \multimap B}$, then $\lambda_{A \multimap B}^{OP}(s) = (\lambda_A^{OP}(s|_A) \Rightarrow \lambda_B^{OP}(s|_B))$, where $\Rightarrow$ is the binary operation on $\{O, P\}$ defined by

P	Q	$P \Rightarrow Q$
P	P	P
O	P	P
P	O	O
O	O	P

.

Moreover, if $\lambda_A^{OP}(s|_A) = O$ then $\lambda_A^{OP}(s|_B) = O$.

Proof. Induction on the length of s. If $s = \epsilon$, then $s|_A = s|_B = \epsilon$, and so $(\lambda_A^{OP}(s|_A) \Rightarrow \lambda_B^{OP}(s|_B)) = (P \Rightarrow P) = P = \lambda_{A \multimap B}^{OP}(s)$.

Suppose then that $s = ta$, and that $\lambda_{A \multimap B}^{OP}(t) = O$. This means that $\lambda_A^{OP}(t|_A) = P$ and $\lambda_B^{OP}(t|_B) = O$. Then, whether a is a move in A or a move in B, adding it will flip exactly one of these components – so $\lambda_{A \multimap B}(s|_A) = O$ and $\lambda_{A \multimap B}^{OP}(s|_B) = O$ if a is a move in A and $\lambda_{A \multimap B}(s|_A) = P$ and $\lambda_{A \multimap B}^{OP}(s|_B) = P$ if a is a move in C.

Suppose instead that $\lambda_{A \multimap B}^{OP}(t) = P$. By induction, this means that either $\lambda_A^{OP}(t|_A) = P$ and $\lambda_B^{OP}(t|_B) = P$ or that $\lambda_A^{OP}(t|_A) = O$ and $\lambda_B^{OP}(t|_B) = O$. In the first case, this means that either $t|_A$ is empty or its last move is a P-move in A (and therefore an O-move in $A \multimap B$), and so the move a must take place in C, meaning that $\lambda_A^{OP}(s|_A) = P$ and $\lambda_B^{OP}(s|_B) = O$.

Similarly, in the second case, the last move in $t|_C$ must be an O-move in B (and therefore an O-move in $A \multimap B$, and so the move a must take place in A, meaning that $\lambda_A^{OP}(s|_A) = P$ and $\lambda_B^{OP}(s|_B) = O$. $\square$

It follows that

Corollary 1.16 (Switching condition)**.** *Only player P may switch between games in $A \multimap B$; i.e., if $tab \in P_{A \multimap B}$, and a occurs in A and b in B, or if a occurs in B and b in A, then b is a P-move.*

Proof. Otherwise, $\lambda_{A \multimap B}(t) = O$, so $\lambda_A(t|_A) = P$ and $\lambda_B(t|_B) = O$. But we must also have $\lambda_{A \multimap B}(tab) = O$, so $\lambda_A(tab|_A) = P$ and $\lambda_B(tab|_B) = O$. But this is a contradiction, since $tab|_A$ and $tab|_B$ are both one move longer than the plays $t|_A$ and $t|_B$. $\square$

Definition 1.17 ([Har06, §3.1])**.** Given $\mathfrak{s} \in \mathrm{int}(A, B, C)$, we define the *$P$-view* $\ulcorner \mathfrak{s} \urcorner$ of $\mathfrak{s}$ inductively as follows.

$$\begin{aligned}
\ulcorner \epsilon \urcorner &= \epsilon \\
\ulcorner \mathfrak{s}a \urcorner &= \ulcorner \mathfrak{s} \urcorner a && \text{if } a \text{ is a move in } B, \text{ an } O\text{-move in } A \text{ or a } P\text{-move in } C \\
\ulcorner \mathfrak{s}c \urcorner &= c && \text{if } c \text{ is an initial move of } C \\
\ulcorner \mathfrak{s}b\mathfrak{t}a \urcorner &= \ulcorner \mathfrak{s} \urcorner ba && \text{if } a \text{ is a } P\text{-move of } A \text{ or an } O\text{-move of } C \text{ and is justified by } b
\end{aligned}$$

Lemma 1.18. *If $\mathfrak{s} \in \mathrm{int}(A, B, C)$, then $\ulcorner \mathfrak{s} \urcorner|_{A,C} = \ulcorner \mathfrak{s}|_{A,C} \urcorner$.*

Proof. Induction on the length of $\mathfrak{s}$. This is clear if $\mathfrak{s} = \epsilon$.

If a is an O-move in A or a P-move in C, then a is a P-move in $A \multimap C$. We have $\ulcorner \mathfrak{s}a \urcorner|_{A,C} = \ulcorner \mathfrak{s} \urcorner|_{A,C}a$, which by the inductive hypothesis is equal to $\ulcorner \mathfrak{s}|_{A,C} \urcorner a$, which is the same as $\ulcorner \mathfrak{s}a|_{A,C} \urcorner$. If b is a move in B, then $\ulcorner \mathfrak{s}b \urcorner|_{A,C} = \ulcorner \mathfrak{s} \urcorner b|_{A,C} = \ulcorner \mathfrak{s} \urcorner|_{A,C} = \ulcorner \mathfrak{s}|_{A,C} \urcorner = \ulcorner \mathfrak{s}b|_{A,C} \urcorner$, by the inductive hypothesis.

If c is initial in C, then $\ulcorner \mathfrak{s}c \urcorner|_{A,C} = c = \ulcorner \mathfrak{s}c|_{A,C} \urcorner$.

Suppose a is a P-move of A or an O-move of C – so a is an O-move in $A \multimap C$ – and suppose that a is justified by b in the sequence $\mathfrak{s}b\mathfrak{t}a$. Since a cannot be an initial move in A, b must occur in the same game as a, and in particular must not occur in B. Then we have $\ulcorner \mathfrak{s}b\mathfrak{t}a \urcorner|_{A,C} = \ulcorner \mathfrak{s} \urcorner ba|_{A,C} = \ulcorner \mathfrak{s} \urcorner|_{A,C}ba$, which by the inductive hypothesis is equal to $\ulcorner \mathfrak{s}|_{A,C} \urcorner ba = \ulcorner \mathfrak{s}ba|_{A,C} \urcorner$. $\square$

Lemma 1.19 ([Har06, §3.1])**.** *Let $\mathfrak{s}a \in \mathrm{int}(A, B, C)$ (so, in particular, $\mathfrak{s}|_{A,B}$ and $\mathfrak{s}|_{B,C}$ satisfy the visibility condition). If a is a move in B, an O-move in A or a P-move in C, then $\ulcorner \mathfrak{s}a \urcorner \in \mathrm{int}(A, B, C)$.*

Proof. Induction on the length of $\mathfrak{s}$. If $\mathfrak{s} = \epsilon$, then this is clear. Otherwise, suppose that $\mathfrak{s}$ is non-empty.

First, we claim that $\ulcorner \mathfrak{s} \urcorner \in \mathrm{int}(A, B, C)$. If $\mathfrak{s}$ ends with a move in B, an O-move in A or a P-move in C, then this follows immediately from the inductive hypothesis. Otherwise, suppose that $\mathfrak{s}$ ends with a P-move in A or an O-move in C. If this last move is initial, then $\ulcorner \mathfrak{s} \urcorner$ is a single move, so the claim is trivial. Otherwise, write $\mathfrak{s} = \mathfrak{t}p\mathfrak{u}r$, where p justifies r. By the inductive hypothesis, we have $\ulcorner \mathfrak{t}p \urcorner \in \mathrm{int}(A, B, C)$, and then $\ulcorner s \urcorner = \ulcorner \mathfrak{t}p\mathfrak{u}r \urcorner = \ulcorner \mathfrak{t} \urcorner pr = \ulcorner \mathfrak{t}p \urcorner r \in \mathrm{int}(A, B, C)$.

Now, since a is a P-move in $A \multimap B$ or in $B \multimap C$, its predecessor b is an O-move and has some justifier c contained in $\ulcorner \mathfrak{s}|_X \urcorner$, where $X \in \{A \multimap B, B \multimap C\}$ is that component in which a is a P-move. Then this c is preceded by some other O-move b', which is necessarily also contained in $\ulcorner \mathfrak{s} \urcorner$, and so has some justifier c', contained in $\ulcorner \mathfrak{s}|_X \urcorner$ by visibility. Continuing in this way until we reach an initial move, we build up the whole of the sequence $\ulcorner \mathfrak{s}|_X \urcorner$ as a subsequence of $\ulcorner \mathfrak{s} \urcorner$. Therefore, the justifier of a must be contained in $\ulcorner \mathfrak{s} \urcorner$, and so $\ulcorner \mathfrak{s}a \urcorner = \ulcorner \mathfrak{s} \urcorner a \in \mathrm{int}(A, B, C)$. ☐

Lemma 1.20 (O-views in the linear implication, [HO00, 4.2,4.3])**.** *Let A, B be games, and let bs be a non-empty play in $A \multimap B$ beginning with an initial move b in B.*

i) If bs ends with a P-move in B, then $\llcorner bs \lrcorner_{A \multimap B} = \llcorner bs|_B \lrcorner_B$.

ii) If bs ends with a P-move in A, then $\llcorner bs \lrcorner_{A \multimap B} = b \ulcorner s|_A \urcorner^A$.

Proof. Induction on the length of s. If $s = \epsilon$, then bs ends with an O-move in B, and we have $\ulcorner b \urcorner^{A \multimap B} = b = \ulcorner b \urcorner^B$.

Otherwise, suppose that bs ends with a P-move c in B. Let d be the justifier of c. Then d must be an O-move in B. Write $bs = tduc$, where t, u are sequences. Then $\llcorner tduc \lrcorner_{A \multimap B} = \llcorner t \lrcorner_{A \multimap B} dc$ and $\llcorner tduc|_B \lrcorner_B = \llcorner t|_B \lrcorner_B dc$. By Corollary 1.16, t must end with a P-move in B, or be empty, so by the inductive hypothesis we have $\llcorner t \lrcorner_{A \multimap B} = \llcorner t|_B \lrcorner_B$. Therefore, $\llcorner bs \lrcorner_{A \multimap B} = \llcorner tduc \lrcorner_{A \multimap B} = \llcorner t \lrcorner_B dc = \llcorner tduc|_B \lrcorner_B = \llcorner bs|_B \lrcorner_B$.

Next, suppose that bs ends with a P-move a in A. Let c be the justifier of a. Then c must be an O-move in A. Write $s = tcua$, where t, u are sequences. Then $\llcorner btcua \lrcorner_{A \multimap B} = \llcorner bt \lrcorner_{A \multimap B} ca$ and $\ulcorner tcua|_A \urcorner^A = \ulcorner t|_A \urcorner^A ca$, since the roles are reversed in A. By Corollary 1.16, t must end in a P-move in A, or be empty, so by the inductive hypothesis we have $\llcorner bt \lrcorner_{A \multimap B} = b \ulcorner t|_A \urcorner^A$. Therefore, $\llcorner bs \lrcorner_{A \multimap B} = \llcorner btcua \lrcorner_{A \multimap B} = \llcorner bt \lrcorner_{A \multimap B} ca = b \ulcorner t|_A \urcorner^A ca = b \ulcorner tcua|_A \urcorner^A = b \ulcorner s|_A \urcorner^A$. ☐

Proposition 1.21. *$\sigma; \tau$ is a strategy for $A \multimap C$.*

Proof. First, we claim that $\mathfrak{s}|_{A,C} \in P_{A \multimap B}$ for any $\mathfrak{s} \in \sigma \| \tau$. Since we certainly have $\mathfrak{s}|_{A,C}|_A = \mathfrak{s}|_{A,B}|_A \in P_A$ and $\mathfrak{s}|_{A,C}|_C = \mathfrak{s}|_C = \mathfrak{s}|_{B,C}|_C \in P_C$, it suffices to show that $\mathfrak{s}|_{A,C} \in L_{A \multimap C}$.

Suppose that $ta \sqsubseteq \mathfrak{s}|_{A,C}$. We claim that $\lambda_{A \multimap C}(t) = \neg\lambda_{A \multimap C}(a)$. By Lemma 1.15, we are in one of the following configurations.

$\lambda_A^{OP}(t\|_A)$	$\lambda_B^{OP}(t\|_B)$	$\lambda_C^{OP}(t\|_C)$	$\lambda_{A \multimap B}^{OP}(t\|_{A,B})$	$\lambda_{B \multimap C}^{OP}(t\|_{B,C})$	$\lambda_{A \multimap C}^{OP}(t\|_{A,C})$
P	P	P	P	P	P
P	P	O	P	O	O
P	O	O	O	P	O
O	O	O	P	P	P

In the configuration PPP, the move a cannot be a move in A, since that would leave $ta|_{A \multimap B}$ in the configuration OP, which is impossible by Lemma 1.15. Therefore, it must be a move in C, and must therefore be an O-move in C and hence an O-move in $A \multimap C$.

In the configuration PPO, once again the move a cannot take place in A, since this would leave $ta|_{A \multimap B}$ in an illegal configuration. Therefore, it must occur in C, and must be a P-move in C and hence a P-move in $A \multimap C$.

In the configuration POO, the move a cannot take place in C, or it would leave $ta|_{B,C}$ in the illegal configuration OP, so the move a takes place in A. Therefore, it must be an O-move in A and hence a P-move in $A \multimap C$.

Lastly, in the configuration OOO, the move a cannot occur in C, or it would leave $ta|_{B,C}$ in the configuration OP, and so it must take place in A. Therefore, it must be a P-move in A, and hence an O-move in $A \multimap C$.

Having established that $\mathfrak{s}|_{A,C}$ is alternating, we show that it is well-bracketed. Suppose that a question move q in $\mathfrak{s}|_{A,C}$ justifies some answer move a. q and a must occur in the same component, since the only case in which a move from one of A and C can justify a move in the other is when both moves are initial, and hence questions. Suppose first that q and a both occur in the game C. Suppose that some other question move q' occurs between q and a in $\mathfrak{s}|_{A,C}$. If q' occurs in C, then it must be answered by some a' occurring between q' and a, since $\mathfrak{s}|_C$ is a well-bracketed sequence. Otherwise, suppose that q' occurs in A.

By examining the table above, we see that there must be some move in B occurring between q and q' in $\mathfrak{s}$, since moves in A move between configurations OOO and POO, while moves in C move us between configurations PPP and PPO. Let b be the earliest such move. Then b must be a question; indeed, if it is an answer, then it is non-initial and so can only be justified by questions in B. But such a question must occur earlier in $\mathfrak{s}|_{B,C}$ than q, which would mean that q was an unanswered question when the move b was played, contradicting well-bracketedness of $\mathfrak{s}|_{B,C}$. Since b is a question, it must be answered by some a'' occurring between b and a. Therefore, since $\mathfrak{s}|_{A,B}$ is well-bracketed, the move q' must be answered by some a' occurring between a' and a'' in $\mathfrak{s}|_{A,B}$, and therefore between a' and a in $\mathfrak{s}|_{A,C}$.

The case when q and a both occur in A is similar.

Lastly, we need to show that $\mathfrak{s}|_{A,C}$ satisfies the visibility condition. Let $ta \sqsubseteq \mathfrak{s}|_{A,C}$. Choose some $\mathfrak{t} \sqsubseteq \mathfrak{s}$ such that $\mathfrak{t}|_{A,C} = t$.

Suppose a is a P-move. Then by Lemma 1.19, $\ulcorner \mathfrak{t}a \urcorner \in \text{int}(A, B, C)$. By Lemma 1.18, $\ulcorner t \urcorner a = \ulcorner ta \urcorner = \ulcorner \mathfrak{t}a \urcorner|_{A,C}$, and therefore that the justifier of a must be inside $\ulcorner t \urcorner$.

Secondly, suppose that a is an O-move. If a is an O-move in C, then either it is initial or t ends with some P-move in B, and therefore $\llcorner t \lrcorner_{A \multimap C} = \llcorner t|_B \lrcorner_B = \llcorner \mathfrak{t}|_{B,C} \lrcorner_B$. Therefore, since $t|_{B,C}$ satisfies visibility, the justifier of a must lie in $\llcorner t \lrcorner_{A \multimap C}$. If a is an O-move in A, then write $t = cu$ and $\mathfrak{t} = c\mathfrak{u}$, where c is the starting move in C. We have $\llcorner cua \lrcorner_{A \multimap C} = c \ulcorner u|_A a \urcorner^A = \llcorner c\mathfrak{u}|_{A,B} \lrcorner_{A \multimap B}$. Therefore, the justifier of a must lie in $\llcorner t \lrcorner_{A \multimap C}$.

Therefore, $\mathfrak{s}|_{A,C} \in L_{A \multimap C}$, so $\mathfrak{s}|_{A,C} \in P_{A \multimap C}$.

It is fairly clear that $\sigma;\tau$ is even-prefix closed, since σ and τ are. Indeed, if $\mathfrak{s}|_{A,C} \in \sigma;\tau$ and $t \sqsubseteq \mathfrak{s}|_{A,C}$, then we may choose some prefix $\mathfrak{t}$ of $\mathfrak{s}$ such that $t = \mathfrak{t}|_{A,C}$. Then $\mathfrak{t}|_{A,B} \sqsubseteq \mathfrak{s}|_{A,B} \in \sigma$ and $\mathfrak{t}|_{B,C} \sqsubseteq \mathfrak{s}|_{B,C} \in \tau$, so $\mathfrak{t} \in \sigma \| \tau$.

We claim that every sequence in $\sigma;\tau$ has even length. Indeed, if $\mathfrak{s}|_{A,B} \in \sigma$ and $\mathfrak{s}|_{B,C} \in \tau$, then both $\mathfrak{s}|_{A,B}$ and $\mathfrak{s}|_{B,C}$ must have even length, so must be in configuration OO or PP. This means that $\mathfrak{s}$ as a whole must be in configuration OOO or PPP, and so $\mathfrak{s}|_{A,C}$ must be in configuration OO or PP, so must have even length.

Lastly, we need to show that $\sigma;\tau$ is deterministic. Suppose that $sab, sac \in \sigma;\tau$, and suppose that $b \neq c$. Suppose that $\mathfrak{s}|_{A,C} = sab$ and $\mathfrak{t}|_{A,C} = sac$, for $\mathfrak{s}, \mathfrak{t} \in \sigma \| \tau$, and let $\mathfrak{u}$ be the longest common prefix of $\mathfrak{s}, \mathfrak{t}$. $\mathfrak{s}$ and $\mathfrak{t}$ are certainly incomparable under the prefix ordering, since $\mathfrak{s}|_{A,C}$ and $\mathfrak{t}|_{A,C}$ are, so we have $\mathfrak{u}p \sqsubseteq \mathfrak{s}$ and $\mathfrak{u}q \sqsubseteq \mathfrak{t}$, where $p \neq q$. Now p and q cannot be O-moves in A, P-moves in C or moves in B, or they would have to be equal by determinism of σ and τ. Therefore, they are P-moves in A or O-moves in C, but this contradicts $\mathfrak{s}|_{A,C} = sab$ and $\mathfrak{t}|_{A,C} = sac$.

Therefore, the composition $\sigma;\tau$ is a strategy. $\square$

We also want to show that the composition of innocent strategies is innocent. We follow the proof given in [Har06]. First, we use a lemma.

Lemma 1.22 ([Har06, 3.3.3])**.** *Let* $\mathfrak{s}a \in \text{int}(A, B, C)$.

i) If a is a P-move of A or an O-move of B, then $\ulcorner \mathfrak{s}a|_{A,B} \urcorner = \ulcorner \ulcorner \mathfrak{s}a \urcorner|_{A,B} \urcorner$.

ii) If a is a P-move of B or an O-move of C, then $\ulcorner \mathfrak{s}a|_{B,C} \urcorner = \ulcorner \ulcorner \mathfrak{s}a \urcorner|_{B,C} \urcorner$.

Proof. Induction on the length of $\mathfrak{s}$. We prove (i); the proof of (ii) is exactly the same.

If a is a P-move of A or an O-move of B, then it is an O-move of $A \multimap B$. If a is an initial move of $A \multimap B$, then we have $\ulcorner \mathfrak{s}|_{A,B} a \urcorner = a = \ulcorner a \urcorner|_{A,B} = \ulcorner\ulcorner \mathfrak{s} a \urcorner\urcorner|_{A,B}$. Otherwise, write $\mathfrak{s} = \mathfrak{t} b \mathfrak{u}$, where b justifies a. Then $\ulcorner \mathfrak{s} a|_{A,B} \urcorner = \ulcorner \mathfrak{t}|_{A,B} b \mathfrak{u}|_{A,B} a \urcorner = \ulcorner t|_{A,B} \urcorner b a$, which by the inductive hypothesis is equal to $\ulcorner\ulcorner \mathfrak{t} \urcorner|_{A,B} \urcorner b a$, which is equal to $\ulcorner\ulcorner \mathfrak{t} b \mathfrak{u} a \urcorner|_{A,B} \urcorner = \ulcorner\ulcorner \mathfrak{s} a \urcorner|_{A,B} \urcorner$. $\square$

Proposition 1.23. *If $\sigma\colon A \multimap B$ and $\tau\colon B \multimap C$ are innocent strategies, then $\sigma;\tau\colon A \multimap C$ is innocent.*

Proof. Suppose there are $sab, t \in \sigma;\tau$ such that $ta \in P_{A \multimap C}$, $\ulcorner sa \urcorner = \ulcorner ta \urcorner$. Let $\mathfrak{s}'b$ be such that $\mathfrak{s}'b|_{A,C} = sab$ and choose the minimal prefix $\mathfrak{s} \sqsubseteq \mathfrak{s}'$ such that $\mathfrak{s}a|_{A,C} = sa$.

Let $\mathfrak{t}a$ be such that $\mathfrak{t}a|_{A,C} = ta$. Since $\ulcorner sa \urcorner = \ulcorner ta \urcorner$, we have $\ulcorner \mathfrak{s}a \urcorner|_{A,C} = \ulcorner \mathfrak{s}a|_{A,C} \urcorner = \ulcorner sa \urcorner = \ulcorner ta \urcorner = \ulcorner \mathfrak{t}a|_{A,C} \urcorner = \ulcorner \mathfrak{t}a \urcorner|_{A,C}$ by Lemma 1.18. Let $\mathfrak{u}$ be the longest common prefix of $\ulcorner \mathfrak{s}a \urcorner$ and $\ulcorner \mathfrak{t}a \urcorner$. If $\mathfrak{s}a$ and $\mathfrak{t}a$ are not equal, then without loss of generality there is some $\mathfrak{u}p \sqsubseteq \mathfrak{s}$, where $\mathfrak{u}p \not\sqsubseteq \mathfrak{t}$. Then, by determinism of σ and τ, this p cannot be a P-move in either $A \multimap B$ or $B \multimap C$, so it must be a P-move in A or an O-move in C, and is therefore preceded by another move in A or C, which contradicts $\ulcorner \mathfrak{s}a \urcorner|_{A,C} = \ulcorner \mathfrak{t}a \urcorner|_{A,C}$. Therefore, $\ulcorner \mathfrak{s}a \urcorner = \ulcorner \mathfrak{t}a \urcorner$.

Now write $\mathfrak{s}' = \mathfrak{s}ab_1 \cdots b_n b$, where each b_i is a move in B. We show by induction that $\mathfrak{t}ab_1 \cdots b_j \in \sigma\|\tau$. Indeed, if $\mathfrak{t}ab_1 \cdots b_{j-1} \in \sigma\|\tau$, then b_j (or b) is a P-move in either $A \multimap B$ or $B \multimap C$, and b_{j-1} is an O-move in that same component. Write X for the component ($A \multimap B$ or $B \multimap C$) in which b_j is a P-move. Repeating the argument above, we see that $\ulcorner \mathfrak{t}ab_1 \cdots b_{j-1} \urcorner = \ulcorner \mathfrak{s}ab_1 \cdots b_{j-1} \urcorner$, and so we have that $\ulcorner \mathfrak{t}ab_1 \cdots b_{j-1}|_X \urcorner = \ulcorner \mathfrak{s}ab_1 \cdots b_{j-1}|_X \urcorner$ by Lemma 1.22. Therefore, by innocence of σ (if $X = A \multimap B$) or τ (if $X = B \multimap C$), we see that $\mathfrak{t}ab_1 \cdots b_j \in \sigma\|\tau$. It follows that $\mathfrak{t}ab_1 \cdots b_n b \in \sigma\|\tau$, and therefore that $tab \in \sigma;\tau$. $\square$

1.5 Associativity of composition

In this section, we will prove that composition is associative; i.e., that if $\sigma\colon A \multimap B$, $\tau\colon B \multimap C$ and $\upsilon\colon C \multimap D$ are strategies, then $(\sigma;\tau);\upsilon = \sigma;(\tau;\upsilon)$. To do this, if A, B, C, D are arenas, we define the set $\mathrm{int}(A, B, C, D)$ to be the set of all sequences $\mathfrak{u}$ of moves such that $\mathfrak{u}|_{A,B} \in L_{A \multimap B}$, $\mathfrak{u}|_{B,C} \in L_{B \multimap C}$ and $\mathfrak{u}|_{C,D} \in L_{C \multimap D}$. Given such a sequence $\mathfrak{u}$, we define $\mathfrak{u}|_{A,D}$ as before; i.e., we take all moves from $\mathfrak{u}$ occurring in A and D, together with justification pointers within these games, and if an initial move in A is justified by an initial move in B, which is justified by an initial move in C, which is justified by an initial move in D, then we add a justification pointer from that move in A to that move in D.

Given strategies σ, τ, υ as above, we define $\sigma\|\tau\|\upsilon$ to be the set of all $\mathfrak{u} \in \mathrm{int}(A, B, C, D)$ such that $\mathfrak{u}|_{A,B} \in \sigma$, $\mathfrak{u}|_{B,C} \in \tau$ and $\mathfrak{u}|_{C,D} \in \upsilon$. We then claim that:

Lemma 1.24.

$$(\sigma;\tau);\upsilon = \{\mathfrak{u}|_{A,C} \,:\, \mathfrak{u} \in \sigma\|\tau\|\upsilon\} = \sigma;(\tau;\upsilon)\,.$$

Proof. Firstly, if $\mathfrak{u} \in \sigma\|\tau\|\upsilon$, then it is clear to see that $\mathfrak{u}|_{A,B,C} \in \sigma\|\tau$ and that $\mathfrak{u}|_{B,C,D} \in \tau\|\upsilon$, and therefore that $\{\mathfrak{u}|_{A,C} \,:\, \mathfrak{u} \in \sigma\|\tau\|\upsilon\} \subseteq (\sigma;\tau);\upsilon$ and $\{\mathfrak{u}|_{A,C} \,:\, \mathfrak{u} \in \sigma\|\tau\|\upsilon\} \subseteq \sigma;(\tau;\upsilon)$.

Conversely, suppose that $\mathfrak{t} \in (\sigma;\tau)\|\upsilon$, so that $\mathfrak{t}|_{A,C} \in \sigma;\tau$ and $\mathfrak{t}|_{C,D} \in \upsilon$, and choose some $\mathfrak{s} \in \sigma\|\tau$ such that $\mathfrak{s}|_{A,C} = \mathfrak{t}|_{A,C}$. We may write

$$\mathfrak{s} = \mathbf{c}_1\mathbf{b}_1\mathbf{a}_1 \cdots \mathbf{c}_n\mathbf{b}_n\mathbf{a}_n$$

for some (possibly empty) sequences of moves $\mathbf{a}_i$ from A, $\mathbf{b}_i$ from B and $\mathbf{c}_i$ from C. We may then write

$$\mathfrak{t} = \mathbf{d}_1\mathbf{c}_1\mathbf{a}_1 \cdots \mathbf{d}_n\mathbf{c}_n\mathbf{a}_n$$

(for the same $\mathbf{a}_i$, $\mathbf{c}_i$), and we can therefore interleave these sequences into the sequence

$$\mathfrak{u} = \mathbf{d}_1\mathbf{c}_1\mathbf{b}_1\mathbf{a}_1 \cdots \mathbf{d}_n\mathbf{c}_n\mathbf{b}_n\mathbf{a}_n\,,$$

which is in $\sigma\|\tau\|\upsilon$. Then we have $\mathfrak{u}|_{A,D} = \mathfrak{t}|_{A,D}$, and it follows that $(\sigma;\tau);\upsilon \subseteq \{\mathfrak{u}|_{A,C} \,:\, \mathfrak{u} \in \sigma\|\tau\|\upsilon\}$, and the case for $\sigma;(\tau;\upsilon)$ is identical. □

1.6 Copycat strategies

Definition 1.25. Let A, B be games. Then a *structural isomorphism* from A to B is a bijection $f\colon M_A \to M_B$ such that $a \vdash_A b$ if and only if $f(a) \vdash_B f(b)$ and $* \vdash_A a$ if and only if $* \vdash_B f(a)$, and such that the plays of B are precisely the plays of A, after applying the function f pointwise to the moves (and keeping the justification indices as they are).

Given a structural isomorphism f from A to B, the *copycat strategy* $\mathrm{cc}_f\colon A \to B$ is given by

$$\mathrm{cc}_f = \{s \in P^{even}_{A \multimap B} \,:\, \text{for all even-length } t \sqsubseteq s,\ t|_B = f^*(t|_A)\}\,,$$

where f^* denotes f applied pointwise.

Proposition 1.26. cc_f *is an innocent strategy for* $A \multimap B$*. Moreover, if* $\sigma\colon C \multimap A$ *is a strategy, then*

$$\sigma;\mathrm{cc}_f = \{[\mathrm{id}_{M_C}, f]^*(s) \,:\, s \in \sigma\}\,,$$

and if $\tau\colon B \multimap D$ *is a strategy, then*

$$\mathrm{cc}_f;\tau = \{[f^{-1}, \mathrm{id}_{M_D}]^*(s) \,:\, s \in \tau\}\,.$$

Proof. cc_f is clearly prefix-closed by definition. Suppose that $sab, sac \in \mathrm{cc}_f$; then $s|_B = f^*(s|_A)$ and $sab|_B = f^*(sab|_A) = f^*(s|_A)f^*(ab|_A) = s|_B f^*(ab|_A)$. It follows that either $ab|_B = f^*(ab|_A)$, so either a is a move in A and $b = f(a)$ or a is a move in b and $a = f(b)$. Since the same applies to c, we have $b = c$.

This argument shows that cc_f is *history-free* – i.e., that its reply to an O-position is entirely determined by the last O-move – and therefore it is certainly innocent.

Now let $\sigma\colon C \multimap A$ be a strategy. Suppose that $\mathfrak{s} \in \sigma \| \mathrm{cc}_f$. Then $\mathfrak{s}|_{C,A} \in \sigma$ and $\mathfrak{s}|_B = f^*(\mathfrak{s}|_A)$. It follows that $\mathfrak{s}|_{C,B} = [\mathrm{id}_{M_C}, f]^*(\mathfrak{s}|_{C,A})$.

Conversely, given $s \in \sigma$, for each P-move a in s occurring in the component A, insert the move $f(a)$ immediately after it, and for each O-move b in s occurring in the component A, insert the move $f(b)$ immediately before it. Let these extra moves in B be justified according to the original moves in A. Then the resulting sequence $\mathfrak{s}$ is contained in $\sigma \| \mathrm{cc}_f$, and $\mathfrak{s}|_{A,C} = [\mathrm{id}_{M_C}, f]^*(s)$.

The case for composition in the other direction is exactly the same. □

An easy corollary of this fact is that composition of copycat strategies respects composition of the underlying structural isomorphisms.

Corollary 1.27. *Let f be a structural isomorphism from A to B and let g be a structural isomorphism from B to C. Then $g \circ f$ is a structural isomorphism from A to C and $\mathrm{cc}_{g\circ f} = \mathrm{cc}_f ; \mathrm{cc}_g$.*

It is also easy to see from Proposition 1.26 that the identity function $\mathrm{id}\colon M_A \to M_A$ is a structural isomorphism from A to itself, and that the resulting copycat strategy $\mathrm{cc}_{\mathrm{id}}$ is an identity for composition. Combining this with our result for associativity in the previous section, we get that

Theorem 1.28. *The collection of games forms a category $\mathcal{G}$, where the morphisms $A \to B$ are strategies for $A \multimap B$, composition is as above and the identity morphisms are the copycat strategies induced from the identity functions on moves.*

In this setting, Proposition 1.26 tells us that structural isomorphism gives rise to an isomorphism in $\mathcal{G}$.

Proposition 1.29. *Let f be a structural isomorphism from a game A to a game B. Then cc_f is an isomorphim in $\mathcal{G}$ from A to B.*

1.7 Symmetric Monoidal Closed Structure of $\mathcal{G}$

We now claim that the tensor product connective $\otimes$ makes $\mathcal{G}$ into a symmetric monoidal closed category, with inner hom given by $\multimap$.

Definition 1.30. Let $\sigma\colon A \multimap B$ and $\tau\colon C \multimap D$ be strategies. We define a strategy $\sigma \otimes \tau\colon (A \otimes C) \multimap (B \otimes D)$ by

$$\sigma \otimes \tau = \{s \in P_{(A\otimes C)\multimap(B\otimes D)} \;:\; s|_{A,B} \in \sigma \text{ and } s|_{C,D} \in \tau\}\,.$$

To prove that this is a strategy, we prove a lemma analogous to our Lemma 1.15.

Lemma 1.31. *Let $s \in P_{A\otimes B}$. Then $\lambda^{OP}_{A\otimes B}(s) = \lambda^{OP}_A(s|_A) \wedge \lambda^{OP}_B(s|_B)$, where $\wedge$ is the binary operator on $\{O, P\}$ given by*

p	q	$p \wedge q$
P	P	P
O	P	O
P	O	O
O	O	O

.

Moreover, either $\lambda^{OP}_A(s|_A) = P$ or $\lambda^{OP}_B(s|_B) = P$.

Proof. Mutual induction on the length of s. This is obvious if s is empty. Suppose that $sa \in P_{A\otimes B}$, where a is an O-move. By induction, since $\lambda_{A\otimes B}(s) = P$, we must have $\lambda_{A\otimes B}(s|_A) = P$ and $\lambda_{A\otimes B}(s|_B) = P$. Therefore, depending on which game a is played in, either $\lambda_A(sa|_A) = O$ and $\lambda_B(sa|_B) = P$ or $\lambda_A(sa|_A) = P$ and $\lambda_B(sa|_B) = O$.

If $sb \in P_{A\otimes B}$, where b is a P-move, then by induction either $\lambda_A(s|_A) = O$ and $\lambda_B(s|_B) = P$ or $\lambda_A(s|_A) = P$ and $\lambda_B(s|_B) = O$. In either case, player P must play in whichever game is currently in an O-position, returning us to configuration PP. □

The above proof gives us the following result analogous to Corollary 1.16.

Corollary 1.32 (Switching condition for $\otimes$)**.** *Player O switches games in $A\otimes B$; i.e., if $sab \in P_{A\otimes B}$, where a and b take place in different games (i.e., a in A and b in B or a in B and b in A), then b is an O-move.*

Proposition 1.33. *$\sigma \otimes \tau$ is a strategy for $(A \otimes C) \multimap (B \otimes D)$.*

Proof. $\sigma \otimes \tau$ is certainly an even-prefix-closed subset of $P^{even}_{(A\otimes C)\multimap(B\otimes D)}$.

Let s be a play of $P_{(A\otimes B)\multimap(C\otimes D)}$. We consider the possible configurations of s; i.e., the tuples $(\lambda_A(s|_A), \lambda_B(s|_B), \lambda_C(s|_C), \lambda_D(s|_D))$.

By Lemma 1.15 we must avoid the overall configuration OP for the linear implication, and by Lemma 1.31 we must avoid the configuration OO inside either tensor product, so we end up with the following possibilities.

$\lambda_A(s\|_A)$	$\lambda_C(s\|_C)$	$\lambda_B(s\|_B)$	$\lambda_D(s\|_D)$	$\lambda_{A\otimes C}(s\|_{A,C})$	$\lambda_{B\otimes D}(s\|_{B,C})$	$\lambda_{(A\otimes C)\multimap(B\otimes D)}(s)$
P	P	P	P	P	P	P
P	P	P	O	P	O	O
P	P	O	P	P	O	O
P	O	P	O	O	O	P
O	P	O	P	O	O	P
P	O	O	P	O	O	P
O	P	P	O	O	O	P

Now, if $s \in \sigma \otimes \tau$, or an odd-length sequence formed by adding an O-move to the end of a sequence in $\sigma \otimes \tau$, then we also know that $s|_{A,B} \in \sigma \subseteq P_{A\multimap B}$ and that $s|_{C,D} \in \tau \subseteq P_{C\multimap D}$. This means that we can discount the last two configurations in the table above, since one contains the illegal configuration OP in $C \multimap D$ and the other contains the illegal configuration OP in $A \multimap B$.

Now suppose that $sab, sac \in \sigma \otimes \tau$. Then sa is an O-position in $P_{(A\otimes C)\multimap(B\otimes D)}$, and is therefore in configuration $PPPO$ or $PPOP$. By inspecting the table above, we see that if sa is in configuration $PPPO$, then b and c must both occur either in C or in D, and that if sa is in configuration $PPOP$, then b and c must both occur either in A or in B. In either case, we must have $b = c$, by determinism of τ (in the first case) or of σ (in the second case). □

We need a lemma to prove that the tensor product of two innocent strategies is innocent.

Lemma 1.34. *Let $s \in \sigma \otimes \tau$.*

i) If s ends with a move in A or B, then $\ulcorner s \urcorner^{(A\otimes C)\multimap(B\otimes D)} = \ulcorner s|_{A,B} \urcorner^{A\multimap B}$.

ii) If s ends with a move in C or D, then $\ulcorner s \urcorner^{(A\otimes C)\multimap(B\otimes D)} = \ulcorner s|_{C,D} \urcorner^{C\multimap D}$.

Proof. Induction on the length of s. We prove (i); (ii) is exactly the same.

If a is a P-move, then we have $\ulcorner sa \urcorner = \ulcorner s \urcorner a$. By our analysis in the proof of Proposition 1.33, player P only switches moves between A and B, and between C and D, so s must end with a move from A or B. Therefore, by the inductive hypothesis, $\ulcorner s \urcorner = \ulcorner s|_{A,B} \urcorner$. Then $\ulcorner sa \urcorner = \ulcorner s \urcorner a = \ulcorner s|_{A,B} \urcorner a = \ulcorner sa|_{A,B} \urcorner$.

If a is an initial move, then $\ulcorner sa \urcorner = a = \ulcorner sa|_{A,B} \urcorner$.

If a is an O-move justified by b in $sbta$, then $\ulcorner sbta \urcorner = \ulcorner s \urcorner ba$. Then b is a P-move, so s must end with a move in A or B, as before. Therefore, by the inductive hypothesis, $\ulcorner s \urcorner = \ulcorner s|_{A,B} \urcorner$. Then $\ulcorner sbta \urcorner = \ulcorner s \urcorner ba = \ulcorner s|_{A,B} \urcorner ba = \ulcorner sbta|_{A,B} \urcorner$. □

Proposition 1.35. *Let $\sigma\colon A \to B$, $\tau\colon C \to D$ be innocent strategies. Then $\sigma \otimes \tau$ is innocent.*

Proof. Suppose $sab, t \in \sigma \otimes \tau$ such that $ta \in P_{(A\otimes C)\multimap(B\otimes D)}$ and $\ulcorner sa \urcorner = \ulcorner ta \urcorner$. Suppose without loss of generality that a is a move in A or B. Then $\ulcorner sa \urcorner = \ulcorner sa|_{A,B} \urcorner$ and $\ulcorner ta \urcorner = \ulcorner ta|_{A,B} \urcorner$, and therefore $tab|_{A,B} \in \tau$ by innocence of τ, and so $tab \in \sigma \otimes \tau$. $\square$

The most important thing we need to prove is that $\otimes$ is a functor.

Proposition 1.36. *Let $\sigma' \colon A'' \multimap A'$, $\sigma \colon A' \multimap A$, $\tau' \colon B'' \multimap B'$ and $\tau \colon B' \multimap B$ be strategies. Then $(\sigma' \otimes \tau');(\sigma \otimes \tau) = (\sigma';\sigma) \otimes (\tau';\tau)$.*

Moreover, if A', A, B', B are games, f is a structural isomorphism from A' to A and g is a structural isomorphism from B' to B, then $\mathrm{cc}_f \otimes \mathrm{cc}_g = \mathrm{cc}_{[f,g]}$. In particular, if A and B are games, then $\mathrm{id}_A \otimes \mathrm{id}_B = \mathrm{id}_{A\otimes B}$.

Proof. First suppose that $s \in (\sigma' \otimes \tau');(\sigma \otimes \tau)$; so $s = \mathfrak{s}|_{A'',B'',A,B}$, where $\mathfrak{s} \in (\sigma' \otimes \tau') \| (\sigma \otimes \tau)$. Then $\mathfrak{s}|_{A'',A'} \in \sigma'$ and $\mathfrak{s}|_{A',A} \in \sigma$, so $\mathfrak{s}|_{A'',A',A} \in \sigma' \| \sigma$ and therefore $s|_{A'',A} = \mathfrak{s}|_{A'',A} \in \sigma';\sigma$. Similarly, $s|_{B'',B} \in \tau';\tau$, and therefore $s \in (\sigma';\sigma) \otimes (\tau';\tau)$.

Conversely, suppose that $s \in (\sigma';\sigma) \otimes (\tau';\tau)$. Choose some $\mathfrak{s} \in \sigma' \| \sigma$, $\mathfrak{t} \in \tau' \| \tau$ such that $s|_{A'',A} = \mathfrak{s}|_{A'',A}$ and $s|_{B'',B} = \mathfrak{s}|_{B'',B}$. By our analysis, the only time we switch from the A'', A-component to the B'', B component in s, or *vice versa*, is when player O switches between the games A and B. Thus, we may divide s up into blocks, each starting and ending with a move in the outer component $A \otimes B$. This then gives us a way to divide up $\mathfrak{s}$ and $\mathfrak{t}$ into blocks, such that each block of $\mathfrak{s}$ or $\mathfrak{t}$ projects on to a block of s. Lastly, we can string these blocks together to give us some $\mathfrak{u} \in (\sigma' \otimes \tau') \| (\sigma \otimes \tau)$ such that $\mathfrak{u}|_{A'',B'',A,B} = s$.

For the second part, let A', A, B', B be games, let f be a structural isomorphism from A' to A and let g be a structural isomorphism from B' to B. Then we have

$$\begin{aligned} \mathrm{cc}_f \otimes \mathrm{cc}_g &= \{s \in P_{(A'\otimes B')\multimap(A\otimes B)} \;:\; s|_{A',A} \in \mathrm{cc}_f \text{ and } s|_{B',B} \in \mathrm{cc}_g\} \\ &= \left\{ s \in P_{(A'\otimes B')\multimap(A\otimes B)} \;\middle|\; \begin{array}{l} \text{for all even } t \sqsubseteq s|_{A',A},\ u \sqsubseteq s|_{B',B}, \\ t|_A = f^*(t|_{A'}) \text{ and } u|_B = g^*(u|_{B'}) \end{array} \right\} \end{aligned}$$

$\square$

References

[AJ94] Samson Abramsky and Radha Jagadeesan. Games and full completeness for multiplicative linear logic. *The Journal of Symbolic Logic*, 59(2):543–574, 1994.

[AM96] Samson Abramsky and Guy McCusker. Linearity, sharing and state: a fully abstract game semantics for Idealized Algol with active expressions: Extended abstract. *Electronic Notes in Theoretical Computer Science*, 3:2 – 14, 1996. Linear Logic 96 Tokyo Meeting.

[Har06] Russell Harmer. Innocent game semantics. 2006.

[HO00] J.M.E. Hyland and C.-H.L. Ong. On full abstraction for PCF: I, II, and III. *Information and Computation*, 163(2):285 – 408, 2000.

[Lai] James Laird. Functional programs as coroutines: A semantic analysis. *Logical methods in computer science*. To appear.

[Lai02] J. Laird. A categorical semantics of higher-order store. In *Proceedings of CTCS '02*, number 69 in ENTCS. Elsevier, 2002.