Justified sequences in string diagrams: a comparison between two approaches to concurrent game semantics

Clovis Eberhart and Tom Hirschowitz

CNRS and Université Savoie Mont Blanc

Abstract. We compare two approaches to concurrent game semantics, one by Tsukada and Ong for a simply-typed λ-calculus and the other by the authors and collaborators for CCS and the π-calculus. Both approaches are obviously related, as they both define strategies as sheaves for a Grothendieck topology induced by the embedding of "views" into "plays". However, despite this superficial similarity, the notions of views and plays differ significantly: the former is based on standard justified sequences; the latter uses string diagrams.
In this paper, we relate both approaches at the level of plays. Specifically, we design a notion of play (resp. view) for the simply-typed λ-calculus, based on string diagrams as in our previous work, into which we fully embed Tsukada and Ong's plays (resp. views). We further provide a categorical explanation of why both notions yield essentially the same model, thus demonstrating that the difference is a matter of presentation.

1 Introduction

Two approaches to concurrent game semantics

Innocent game semantics, invented by Hyland and Ong [11], has led to fully abstract models for a variety of functional languages, where programs are interpreted as strategies in a game. Recent advances in concurrent game semantics have produced new games models for CCS and the π-calculus [9, 10, 6] and a non-deterministic, simply-typed λ-calculus [18]. These models are based on categories of innocent and concurrent strategies which share the feature of being defined as categories of *sheaves* over a site of plays.

The two models are obviously related, as they both define innocent strategies as sheaves for the Grothendieck topology induced by embedding views into plays. However, despite this superficial similarity, the notions of views and plays differ significantly. Indeed, Tsukada and Ong define them as *justified sequences* of moves satisfying additional conditions, as in standard Hyland-Ong/Nickau (HON) game-semantics [11, 15]. On the other hand, our plays [9, 10, 6] are based on *ad hoc* string diagrams, as originally suggested by Melliès in a different setting (*circa* 2008, published as [14]).

Since, in our approach, plays are not justified sequences, it is legitimate to wonder why we claim it is game semantics.

Comparing both approaches

In this paper, we relate both approaches not only at the superficial level described above, but also at the level of plays. Specifically, we design notions of play and view for the simply-typed λ-calculus, based on string diagrams as in our previous work, into which we embed plays and views as justified sequences, as defined in classical game semantics. We thus obtain (for each pair of *arenas* [11]) a commuting square of embeddings of categories as on the left below:

$$\begin{array}{ccc} \mathbb{V}_{A,B} & \xhookrightarrow{\mathsf{i}_{HON}} & \mathbb{P}_{A,B} \\ {\scriptstyle F^{\mathbb{V}}}\downarrow & & \downarrow{\scriptstyle F} \\ \mathbb{E}^{\mathbb{V}}(A \vdash B) & \xhookrightarrow[\mathsf{i}]{} & \mathbb{E}(A \vdash B) \end{array} \qquad \begin{array}{ccc} \widehat{\mathbb{V}_{A,B}} & \xhookrightarrow{\Pi_{\mathsf{i}_{HON}}} & \widehat{\mathbb{P}_{A,B}} \\ {\scriptstyle \Delta_{F^{\mathbb{V}}}}\uparrow & & \uparrow{\scriptstyle \Delta_F} \\ \mathbb{E}^{\mathbb{V}}\widehat{(A \vdash B)} & \xhookrightarrow[\Pi_{\mathsf{i}}]{} & \mathbb{E}\widehat{(A \vdash B)} \end{array} \tag{1}$$

where i_{HON} denotes the embedding of views into plays as they are classically defined in game semantics (and which we call HON-views and HON-plays), i denotes the embedding of our views into our plays, and $F^{\mathbb{V}}$ and F denote the constructed embeddings respectively from HON-views into views and from HON-plays into plays. Furthermore, we prove that all these embeddings are full and that $F^{\mathbb{V}}$ is an equivalence of categories (Theorem 1).

Using this and Guitart's theory of *exact squares* [8], we provide a categorical explanation of why both induced categories of innocent strategies coincide.

Terminology 1 *To be more precise, there are two notions of innocent strategies in standard game semantics: the first one is a prefix-closed set of views, the second one a prefix-closed set of plays verifying an extra condition called* innocence. *In both our approach and Tsukada and Ong's, the first notion generalises to presheaves on views, which we call respetively call* behaviours *and* TO-behaviours *(for Tsukada-Ong); the second notion generalises to sheaves on views, which we call respectively* innocent strategies *and* innocent TO-strategies. *Presheaves on plays correspond to possibly non-innocent strategies.*

Our second result (Corollary 1) states that the square of functors on the right of (1), induced by the left-hand side one, commutes up to isomorphism, where, $\prod_f$ denotes right Kan extension along f^{op}, and Δ_f denotes restriction along f^{op}. This entails:

- that $\Delta_{F^{\mathbb{V}}}$ is an equivalence of categories between TO-behaviours and behaviours;
- that Δ_F restricts to an equivalence of categories between innocent TO-strategies and innocent strategies.

Remark 1. Had we wanted to make F an equivalence of categories rather than a mere full embedding, we could easily have imposed an additional condition on our plays akin to alternation in a classical HON-game setting. The point is that we want to compare the purely diagrammatic notion of play with the classical one. And since we obtain an equivalence between both notions of innocent strategies anyway, we feel the result is in fact more convincing.

In summary, our main contribution is a clarification of the link between our plays based on string diagrams and the more classical justified sequences: the difference is essentially a matter of presentation.

Related Work

This paper compares Tsukada and Ong's model [18] to a model inspired by previous work by the authors and collaborators [9, 10, 6] and which builds on ideas from *presheaf models* [12], *causal models* [16], and *game semantics* [1]. Our notion of play, which is based on what we call *string diagrams*, is close in spirit to Melliès's work [14]. Let us also mention different approaches to concurrent game semantics [7, 13, 17]. In a submitted paper [5], we give a categorical reconstruction of the isomorphism between normal forms and certain innocent strategies. We also give a generic definition of the interpretation of terms in normal form as innocent strategies. We show that, in the example of the non-deterministic λ-calculus studied by Tsukada and Ong, our interpretation of terms coincides with theirs, which also shows that the two approaches are related.

Plan

We start by giving a brief recapitulation of game semantics and Tsukada and Ong's notion of strategy, after which we describe and define our notions of plays, views, and strategies. We then relate Tsukada and Ong's plays, views, and strategies to ours.

2 String Diagrams for HON Games

In this section, we design a new approach to HON games, based on *string diagrams*. We first recall classical notions of game semantics as well as Tsukada and Ong's work; whereafter we proceed to describe a sequent calculus (2) that gives some intuition about our model, and finally give a syntax for our plays and views in terms of proof trees in a sequent calculus that generalises (2).

2.1 Tsukada-Ong strategies

Let us start with a brief recapitulation on Tsukada and Ong's categories of views and plays, as well as their notion of strategy.

As usual in game semantics, games are based on *arenas*. An arena consists of a finite set of polarised moves, organised into a forest via a so-called *justification* relation. A compact definition is:

Definition 1. *An* arena *is a simple forest, i.e., a directed, simple graph in which all vertices are uniquely reachable from a unique root (= vertex without a parent).*

Vertices are called *moves*, and roots deemed *initial*. A move m is said to *justify* a move m' when m' is one of m's children.

Notation 1 *All forests considered in the sequel are simple, and we omit to mention it. We denote by $\sqrt{A}$ the set of roots of A. If A is an arena and m is in $\sqrt{A}$, then $A \cdot m$ is the forest strictly below m. The* ownership *of any vertex $m \in A$ is O (for* Opponent*) if the length of the unique path from a root to m is even, and P (for* Proponent*) otherwise. So, e.g., all roots have ownership O. We denote this map $M_A \to \{P, O\}$ by λ_A, where M_A is the set of moves of A.*

Example 1. The boolean arena $\mathbb{B}$ has a single root q, which is an Opponent move, and two Proponent moves `t` and `f`, both justified by q.

Let us fix arenas A and B. Let $A \multimap B$ denote the simple graph obtained by adding to $A + B$ an edge $b \to a$ for all $b \in \sqrt{B}$ and $a \in \sqrt{A}$ (if B is non-empty, otherwise $A \multimap B = \varnothing$). The notion of ownership straightforwardly extends to $A \multimap B$ since all paths from any root to some vertex v have the same length. Concretely, ownership is left unchanged in B but reversed in A.

Remark 2. When B has a single root, $A \multimap B$ can be seen as an arena denoted $A \to B$. Moreover, in that case, we have $(A \to B) \cdot m = A + B \cdot m$ for the only root m of B.

Definition 2. *A* justified sequence *on (A, B) consists of a natural number $n \in \mathbb{N}$, equipped with maps $f : n \to M_A + M_B$ and $\varphi : n \to \{0\} \uplus n$ (here and later in the paper, we use n to denote the set $\{1, \dots, n\}$) such that, for all $i \in n$,*

- *$\varphi(i) < i$,*
- *if $\varphi(i) = 0$ then $f(i) \in \sqrt{B}$, and*
- *if $\varphi(i) \neq 0$, then $f(\varphi(i))$ is a parent of $f(i)$ in $A \multimap B$.*

For any $i \in n$, the *view* $\lceil (n, f, \varphi) \rceil_i$ *of i in* (n, f, φ) is the subset of n defined inductively by:

- $\lceil (n, f, \varphi) \rceil_i = \{i\}$ if i is an Opponent move with $\varphi(i) = 0$,
- $\lceil (n, f, \varphi) \rceil_i = \lceil (n, f, \varphi) \rceil_j \cup \{i\}$ if i is an Opponent move with $\varphi(i) = j > 0$,
- $\lceil (n, f, \varphi) \rceil_i = \lceil (n, f, \varphi) \rceil_{i-1} \cup \{i\}$ if i is a Proponent move.

A justified sequence $s = (n, f, \varphi)$ on (A, B) is *P-visible* when, for all Proponent moves i, $\varphi(i) \in \lceil s \rceil_i$. We further say that s is *alternating* when, for all $i \in n - 1$, $\lambda_{A \multimap B}(i) \neq \lambda_{A \multimap B}(i + 1)$.

Definition 3. *A* preplay *on the pair of arenas (A, B) is a P-visible, alternating, justified sequence on (A, B).*

A morphism of preplays $g : (n, f, \varphi) \to (n', f', \varphi')$ is an injective map $g : n \to n'$ such that:

- $f'(g(i)) = f(i)$ for all $i \in n$,
- $\varphi'(g(i)) = g(\varphi(i))$ for all $i \in n$ (with the convention that $g(0) = 0$),
- $g(2i) = g(2i - 1) + 1$ for all $i \in n/2$.

The last condition says that g should preserve blocks of an Opponent move and the next Proponent move (so-called *OP-blocks*).

Proposition 1. *Pre-plays and morphisms between them form a category $\mathbb{PP}_{A,B}$, with composition given by composition of underlying maps.*

Definition 4. *A* HON-play *is a preplay of even length. The category $\mathbb{P}_{A,B}$ is the full subcategory of $\mathbb{PP}_{A,B}$ spanning HON-plays.*

Example 2. The following justified sequence is a play on the arena pair $(\mathbb{B},\mathbb{B})$, where the justification pointers are drawn as arrows: $q_r \leftarrow q_l \leftarrow \mathsf{t}_l \leftarrow \mathsf{f}_r$. Here, we have written m_l when m is played in the left-hand copy of $\mathbb{B}$, and m_r when it is played in the right-hand one. This particular example shows a possible interaction between a function `f` of type $\mathbb{B} \to \mathbb{B}$ and its environment to compute a value `f(true)`: q_r represents the environment asking `f` to compute the result, q_l then represents `f` asking for its argument, t_l represents the environment telling `f` the value of its argument (`true`), and f_r finally represents `f` returning the value `f(true)` = `false` to the environment.

Definition 5. *If $s = (n, f, \varphi)$ is a justified sequence, i and j are in n, and $\varphi(j) = 0$, we say that i is* hereditarily justified *by j if $i = j$ or $\varphi(i)$ is hereditarily justified by j.*

A thread *of s is a maximal sub-sequence of s where all moves have the same hereditary justifier. The pointers of a thread are inherited from s.*

Definition 6. *A* HON-view *on (A,B) is a non-empty HON-play $s = (n, f, \varphi)$ such that $\lceil s \rceil_n = s$. Let $\mathbb{V}_{A,B}$ denote the full subcategory of $\mathbb{P}_{A,B}$ spanning HON-views.*

Proposition 2. *A HON-play $s = (n, f, \varphi)$ is a HON-view iff for all odd $i \in n$, $\varphi(i) = i - 1$.*

The inclusion $\mathsf{i}_{HON} : \mathbb{V}_{A,B} \hookrightarrow \mathbb{P}_{A,B}$ induces in particular an adjunction

$$\widehat{\mathbb{P}_{A,B}} \underset{\prod_{\mathsf{i}_{HON}}}{\overset{\Delta_{\mathsf{i}_{HON}}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \widehat{\mathbb{V}_{A,B}}.$$

Definition 7. *We call $\widehat{\mathbb{V}_{A,B}}$ the category of* TO-behaviours*. We denote by $\mathrm{Sh}(\mathbb{P}_{A,B})$ the category of* innocent TO-strategies *on (A,B), which is the essential image of $\prod_{\mathsf{i}_{HON}}$.*

By construction, $\prod_{\mathsf{i}_{HON}}$ restricts to an equivalence $\mathrm{Sh}(\mathbb{P}_{A,B}) \simeq \widehat{\mathbb{V}_{A,B}}$.

2.2 Intuition and Informal Description

We start by describing a sequent calculus that gives some intuition about the model we construct in the next section, and describe that model informally.

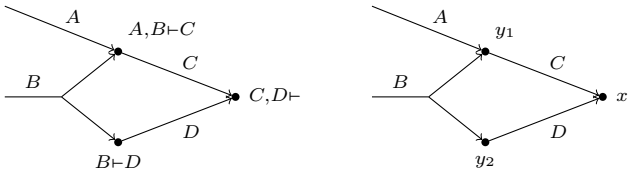
A sequent is a list of arenas, possibly with a distinguished arena, i.e., either of the form $A_1, \ldots, A_n \vdash$ or of the form $A_1, \ldots, A_n \vdash A$, where $A_1, \ldots, A_n, A$ are arenas. Our sequent calculus consists of the three rules below.

$$\frac{\ldots\ \Gamma, A \cdot m \vdash\ \ldots \qquad (\forall m \in \sqrt{A})}{\Gamma \vdash A}\ \text{RIGHT} \qquad \frac{\Gamma, A, \Delta \vdash A \cdot m}{\Gamma, A, \Delta \vdash}\ \text{LEFT} \qquad \frac{\Gamma \vdash A \qquad \Delta, A, \Delta' \vdash}{\Delta, \Gamma, \Delta' \vdash}\ \text{CUT} \tag{2}$$

Our game is inspired by concurrency, so it is intrinsically multi-party, and the topology of communication between the different agents is important. We will therefore need to describe positions, which represent the topology of communication between different agents, and how these positions evolve over time when agents communicate.

Positions A *position* in our game will be some sort of "interaction net" for the sequent calculus above. By this, we mean that a position will consist of a finite set of gates (which we will call *players*) labelled by sequents, and a finite set of wires (which we will call *channels*) labelled by arenas, and that each player labelled $A_1, \ldots, A_n \vdash A$ is connected to n incoming channels labelled $A_1, \ldots, A_n$, and one outgoing channel labelled A (and similarly, players labelled $A_1, \ldots, A_n \vdash$ are connected to n incoming channels labelled accordingly). Players will represent program fragments, while channels will represent ways for different program fragments to communicate. Channels may be connected to 0, 1, 2, or more players, which means that they can somehow be shared between players (they are therefore multi-edges).

A simple way to describe these positions is to draw them. For example, the left-hand side drawing below describes a position with four channels labelled respectively A, B, C, and D, and three players, labelled respectively $(A, B \vdash C)$, $(B \vdash D)$, and $(C, D \vdash)$, and where the first two players share their B channel, and the last player shares both its channels with the first and second player respectively. Since the labelling of players is redundant with that of edges, we will omit it and draw the position as below right.



(3)

Moves Now that we have defined the positions of the game, we should define its "moves", i.e., how positions evolve over time. Channels labelled A represent communication channels on which program fragments can communicate values of "type" A. Therefore, two players can only interact when they are linked by a CUT rule, i.e., when the first player's outgoing channel is one of the other's incoming channels. For example, in the position (3), both y_1 and y_2 can interact

with x (on C and D respectively), but there can be no interaction between y_1 and y_2.

Players labelled $\Gamma \vdash A$ are "passive" players waiting for a signal on their A channel. Players labelled $\Gamma \vdash$ are "active" players who can choose to send a signal on any of their channels. Passive players represent program fragments that are not currently computing anything, either because they are waiting to be called on, or because they have called another program fragment and are waiting for them to return a value to continue their computations. On the other hand, active players represent program fragments that are currently computing something and may call other program fragments or return a value to the program fragment that called them.

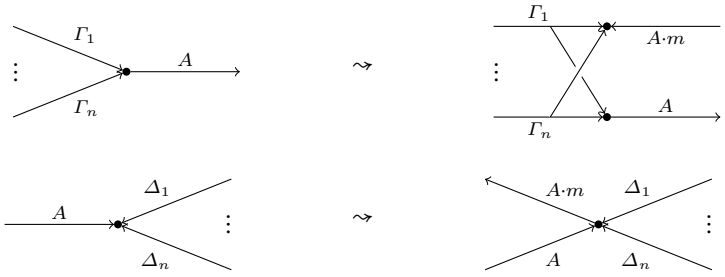
When an active player calls a passive one, their roles are exchanged: the active one becomes passive and *vice versa*. Moreover, a player should not "forget" about program fragments they knew of, because they may reuse them during execution; so they must keep all their incoming edges. Furthermore, the callee must have a way to return the value it computed to its caller, which is modelled by the fact that the interaction creates a new channel shared by both players. This leads us to design the interaction this way: two players labelled $\Gamma \vdash A$ and $\Delta, A, \Delta' \vdash$ can interact on A, and their labels become $\Gamma, B \vdash$ and $\Delta, A, \Delta' \vdash B$ respectively for some well-chosen arena B. In our setting, the interaction will be given by choosing a *root* of A, so B will be $A \cdot m$ for some $m \in \sqrt{A}$. This agrees with classical game semantics in the sense that interaction between program fragments is given by moves in an arena; but it is also slightly finer in the sense that we only allow roots to be played, where classical game semantics allows any move (as long as the whole interaction is a play). In our model, moves that are deeper than roots will be enabled by the fact that interactions between two players on an arena A spawns new arenas that are sub-arenas of A, whose roots can in turn be played as part of a new interaction, and so on. Note that the interaction described here somehow corresponds for the active player to the LEFT rule of our sequent calculus, and for the passive one to a variant of the RIGHT one, with a single $A \cdot m$ chosen.

There is however a last subtlety: we said that program fragments should not forget about other program fragments they knew of. This means in particular that $\Delta, A, \Delta' \vdash A \cdot m$ must remember $\Gamma \vdash A$, since they can call it again once they become active again. This means that $\Gamma \vdash A$ should still exist after the interaction.

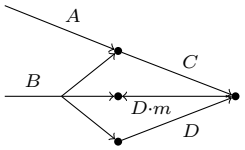
Therefore, the whole interaction can be seen as some rewriting of the position on the left below into the one on the right.

Furthermore, in order to define strategies, we must assume that players can interact with the environment, i.e., players that are not present in the position.

This means that we must allow players to evolve on their own, without necessarily the presence of another connected player. In order to allow this, we must fragment the move above into moves that "only concern a single player", which gives rise to the two rewriting rules below.

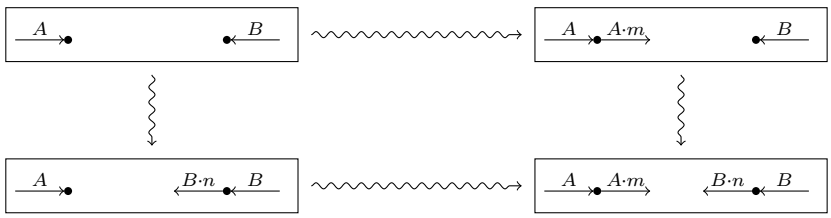


Now, a *move* is simply one of the three rewriting rules shown above, but happening in any context. For example, a possible move would be for x and y_2 in (3) to interact, which would lead to the position on the right. The two players have evolved according to the rule, leaving the rest of the position untouched.



Plays Once we have defined moves, the next step is to define plays. A possible first attempt would be to simply define them as sequences of moves. However, such a definition fails to take into account the topology of communication. Indeed, it forces an arbitrary order between moves that are possibly causally independent from one another, therefore creating two representations of the same object. The same problem occurs in classical game semantics: the definition of plays as sequences of moves is simple, but then morphisms must take into account the fact that some of these sequences are actually basically the same, up to reordering of independent moves.

Therefore, we want our plays to behave as sequences of moves, up to permutation of independent moves. For example, since the moves played by both players below are causally independent from one another, we want both sequences of moves below to be equivalent.



While this sounds very cumbersome, the formal definition of plays is surprisingly simple and corresponds exactly to what we want. We give a simple syntax for them in terms of proof trees in the case where there is a single player of type $(A \vdash B)$ in the initial position (Definition 8).

Strategies

Finally, a strategy over a position X is simply a presheaf over $\mathbb{E}(X)$, the category whose objects are plays starting from X, and whose morphisms are "temporal inclusion" of plays (a play P is included in P' if all the moves occurring in P also occur in P'). In classical game semantics, strategies are sets of accepted plays (with extra conditions), i.e., presheaves $\mathbb{E}(X) \to 2$. This however fails to take into account non-determinism, as exhibited by Milner's famous coffee machine example: a machine that plays a then decides whether to play b or c is different from one that decides whether to play a then b or a then c, but both have the same set of accepted plays. However, presheaves $\mathbb{E}(X) \to \mathsf{set}$ can discriminate these machines: the first one corresponds to a strategy that has a single way of accepting a, while the second one has two ways of accepting it (one of them can only play b, the other only c). In this setting, a strategy S rejects a play P when it cannot accept it, i.e., when $S(P) = \varnothing$.

Note that strategies defined in this way are *not* innocent in general. Indeed, a strategy $S{:}\mathbb{E}(X)^{op} \to \mathsf{set}$ simply associates to each play all the ways S accepts it. The presheaf structure only forces the strategy to accept all the plays "included" in any accepted play, but it does not force it to behave nicely. For example, if X is a position with a channel labelled A on which x_1 and x_2 can output values, and y can input values, then there is a strategy on X in which y accepts to communicate with x_1 but not with x_2, which is not an innocent behaviour.

2.3 A Concurrent Approach to HON Games

Formally, plays are particular pointed presheaves on a base category $\mathbb{L}$ that describes HON games. They are basically higher-dimensional graphs (which we call *string diagrams*), whose lower dimensions describe positions, while their higher dimensions describe the dynamics of the play. This representation of plays in a concurrent setting, which is akin to some abstract approach to graph rewriting [3] (in which moves would be considered rewriting steps) is simpler in many aspects than other such approaches [2]. However, manipulating such string diagrams takes a little bit of getting used to, so we will not give their formal definition, which can be found in the long version [4].

Actually, to relate our plays to Tsukada and Ong's, we are only interested in a very particular type of plays: plays that start on the position composed of a single player $(A \vdash B)$ (and its two channels). In this case, things become much simpler because there can be no interactions between players. Indeed, such an interaction would require two players to be connected by a CUT rule, which is not the case at the beginning and cannot become true by rewriting positions with the rules that concern a single player. Therefore, players evolve on their own, without interacting with others, so we can safely discard all information about channels, which is there to allow interaction to happen. Therefore, there is an equivalent definition of plays on $(A \vdash B)$ in terms of proof trees for the

following sequent calculus:

$$\frac{\dots\ \Gamma, A\cdot m(i) \vdash\ \dots \qquad (\forall i \in I)}{\Gamma \vdash A}\ \textsc{right} \qquad \frac{\Gamma, A, \Delta \vdash A\cdot m}{\Gamma, A, \Delta \vdash}\ \textsc{left} \qquad \frac{}{\Gamma \vdash,}\ \textsc{daimon} \tag{4}$$

where the premises of the RIGHT rule are not ordered, the RIGHT rule can be applied with any family $m{:}I \to \sqrt{A}$, and the LEFT rule can be applied with any $m \in \sqrt{A}$. Notice how the sequent calculus (2) we have introduced at the beginning is basically a specialised version of this without the DAIMON rule and where m is always a bijection in the RIGHT rule.

A play on $(A \vdash B)$ is simply a proof tree whose conclusion is $(A \vdash B)$, which we call an $(A \vdash B)$-*tree*. A morphism of plays on $(A \vdash B)$ is simply the inclusion (both in depth and in width) of an $(A \vdash B)$-tree into another.

Definition 8. *$\mathbb{E}(A \vdash B)$ is the category of plays over $(A \vdash B)$.*

We will not give a complete description of plays in terms of presheaf, but we show how to recover it in the case of plays on $(A \vdash B)$. We extract one-to-one rules from the RIGHT rule and rename the left one to get:

$$\frac{A_1, \dots, A_n, A\cdot m \vdash}{A_1, \dots, A_n \vdash A}\ \Lambda_{(A_1,\dots,A_n\vdash A),m} \qquad \frac{A_1, \dots, A_n \vdash A_i \cdot m}{A_1, \dots, A_n \vdash}\ @_{(A_1,\dots,A_n\vdash),i,m} \tag{5}$$

for all arenas A, $A_1, \dots, A_n$, sequents $(A_1, \dots, A_n \vdash A)$ and $(A_1, \dots, A_n \vdash)$, and move $m \in \sqrt{A}$ in the first rule (and $m \in \sqrt{A_i}$ in the second). From this, we build:

Definition 9. *Let $\mathbb{L}$ be the category with all arenas, sequents, $\Lambda_{S,m}$ and $@_{S,i,m}$ as objects, and as morphisms:*

- *$A \to S$ for each occurence of an arena A in a sequent S,*
- *$S \to M$ for each occurence of a sequent S as premise or conclusion of the rule M (that is either a $\Lambda_{S,m}$ or a $@_{S,i,m}$),*

modulo identification of the arenas of same index in the premise and conclusion of each rule.

We can now give a more formal definition of $\mathbb{E}(A \vdash B)$:

Proposition 3. *$\mathbb{E}(A \vdash B)$ is the identity-on-object subcategory of the coslice $(A \vdash B)/\widehat{\mathbb{L}}$ whose morphisms are injective natural transformations.*

Remark 3. In general, plays over X are objects in the identity-on-objects subcategory of the coslice $X/\widehat{\mathbb{L}'}$ where $\mathbb{L}'$ has a few more objects that $\mathbb{L}$ (to take interaction between players into account) and whose morphisms are injective natural transformations, except maybe on arena objets. In the case of plays over $(A \vdash B)$, the two can easily be shown to be isomorphic, though it is not completely obvious either, because plays are not built as trees in the presheaf-based approach.

In order to define behaviours, we also need to define views:

Definition 10. *A* view *on* $(A \vdash B)$ *is a non-branching* $(A \vdash B)$*-tree of positive, even length. Let* $\mathbb{E}^{\mathbb{V}}(A \vdash B)$ *be the subcategory of* $\mathbb{E}(A \vdash B)$ *spanning views.*

Example 3. Here is an example of play on $(\mathbb{B} \vdash \mathbb{B})$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathbb{B}_l, \{\mathsf{t},\mathsf{f}\}_r, \varnothing_l \vdash \varnothing_r}\,\text{RIGHT}\ (I = \varnothing)}{\mathbb{B}_l, \{\mathsf{t},\mathsf{f}\}_r, \varnothing_l \vdash}\,\text{LEFT}\ (A = \{\mathsf{t},\mathsf{f}\}_r, m = \mathsf{f})}{\mathbb{B}_l, \{\mathsf{t},\mathsf{f}\}_r \vdash \{\mathsf{t},\mathsf{f}\}_l}\,\text{RIGHT}\ (I = \{\mathsf{t}\})}{\mathbb{B}_l, \{\mathsf{t},\mathsf{f}\}_r \vdash}\,\text{LEFT}\ (A = \mathbb{B}_l, m = q)}{\mathbb{B}_l \vdash \mathbb{B}_r,}\,\text{RIGHT}\ (I = \{q\})$$

where the subscripts are meant to indicate whether an arena is a residual of the left-hand copy of $\mathbb{B}$ or of the right-hand one. Since it is not branching, it is also a view. It corresponds to the play shown in Example 2.

Example 4. Consider the HON-play $s = q_r\, \overleftarrow{q_m\, \overleftarrow{q_l\, \mathsf{t}_l}}\, q_l\, \mathsf{f}_l$ on the arena $(\mathbb{B}_l \to \mathbb{B}_m) \to \mathbb{B}_r$ and its sub-preplay $s' = q_r\, q_m\, q_l$. The $(A \vdash B)$-tree that corresponds to s is:

$$\dfrac{\dfrac{\dfrac{I = \varnothing \quad \dfrac{}{\mathbb{B}_l \to \mathbb{B}_m, \{\mathsf{t},\mathsf{f}\}_r, \{\mathsf{t},\mathsf{f}\}_l \vdash \varnothing_l}}{m = \mathsf{t}_l \quad \mathbb{B}_l \to \mathbb{B}_m, \{\mathsf{t},\mathsf{f}\}_r, \{\mathsf{t},\mathsf{f}\}_l \vdash} \qquad \dfrac{\dfrac{}{\mathbb{B}_l \to \mathbb{B}_m, \{\mathsf{t},\mathsf{f}\}_r, \{\mathsf{t},\mathsf{f}\}_l \vdash \varnothing_l}\, I = \varnothing}{m = \mathsf{f}_l \quad \mathbb{B}_l \to \mathbb{B}_m, \{\mathsf{t},\mathsf{f}\}_r, \{\mathsf{t},\mathsf{f}\}_l \vdash}}{I(1) = q_l \qquad I(2) = q_l}}{\mathbb{B}_l \to \mathbb{B}_m, \{\mathsf{t},\mathsf{f}\}_r \vdash \mathbb{B}_l + \{\mathsf{t},\mathsf{f}\}_m}\, m = q_m}{\mathbb{B}_l \to \mathbb{B}_m, \{\mathsf{t},\mathsf{f}\}_r \vdash}}{\mathbb{B}_l \to \mathbb{B}_m \vdash \mathbb{B}_r.}\, I = \{q_r\}$$

There are two ways to embed the $(A \vdash B)$-tree that corresponds to s' into this tree. Indeed, it is equal to the branch going from the root to any of the $(\mathbb{B}_l \to \mathbb{B}_m, \{\mathsf{t},\mathsf{f}\}_r, \{\mathsf{t},\mathsf{f}\}_l \vdash)$ sequents (left or right, it does not matter) and closed with the DAIMON rule. There are therefore two morphisms of $(A \vdash B)$-trees, i.e., two possible ways to embed the branch into the tree (if the DAIMON rule may be embedded into the LEFT rule). This corresponds to the fact that there are two morphisms of preplays from s' to s: the one that exhibits the q_l move in s' as the beginning of $q_l\, \mathsf{t}_l$ and the one that exhibits it as the beginning of $q_l\, \mathsf{f}_l$.

3 Bridging the Gap Between the Two Approaches

In this section, we relate Tsukada and Ong's model to ours. More precisely, we build a functor $F : \mathbb{P}(A, B) \to \mathbb{E}(A \vdash B)$ from HON-plays to plays, which induces a functor from our notion of strategy to theirs. We then show that F is a full embedding, that it restricts to an equivalence $F^{\mathbb{V}} : \mathbb{V}(A, B) \to \mathbb{E}^{\mathbb{V}}(A \vdash B)$ from HON-views to views, and how this induces by the theory of exact squares a tight correspondence between both models.

Note that we cannot expect F to be an equivalence of categories, as we have slightly more plays than classical game semantics. We can indeed model plays with two initial Opponent moves and no Proponent moves, which breaks alternation. What we have is a weaker version of alternation stating that the number of Proponent moves cannot be greater than that of Opponent moves.

3.1 Plays and Views: Relating the Two Approaches

In this section, we first build F, then show it is a full embedding, and that it restricts to an equivalence $F^{\mathbb{V}}$ on both categories of views.

Let us first build $F(s)$ by induction on s. If s is empty, $F(s)$ is simply the application of the RIGHT rule with the empty family. Otherwise, s can be written as a sum of threads $s = \sum_{i \in n} t_i$. Now, each thread t_i is of the form $m_i^1 m_i^2 s_i$ for moves m_i^1, m_i^2 and a play s_i, and we have:

Proposition 4. *For all arenas A and B, if $\mathbb{T}_{A,B}$ is the subcategory of $\mathbb{P}_{A,B}$ spanning threads, $\chi : (mm')/\mathbb{T}_{A,B} \to \mathbb{P}_{C,C \cdot m'}, (mm's) \mapsto s$ is an isomorphism, where $C = A + B \cdot m$.*

Each s_i can therefore be mapped to a $(C_i \vdash C_i \cdot m_i^2)$-tree recursively, where $C_i = A + B \cdot m_i^1$. This tree can straightforwardly be transformed into an $(A, B \cdot m_i^1 \vdash C_i)$-tree $[\![s_i]\!]$, which can in turn be composed with the LEFT and RIGHT rules to give $F(s)$ as below:

$$\dfrac{\dots \qquad \dfrac{\dfrac{[\![s_i]\!]}{A, B \cdot m_i^1 \vdash C_i}\, m = m_i^2}{A, B \cdot m_i^1 \vdash} \qquad \dots}{A \vdash B.}\, I(i) = m_i^1$$

Lemma 1. *F is a full embedding.*

Proof. Injectivity on objects and faithfulness are easy to prove. The proof of fullness is mostly straightforward, except that it requires a reading of the pointers of s inside $F(s)$ to show that the antecedent though F of a morphism of $(A \vdash B)$-trees is indeed a HON-morphism. We explain in details how to read pointers from $F(s)$ below.

First, recovering s's moves from $F(s)$ is easy: they are all the m's used in any LEFT or RIGHT rule, with the obvious multiplicity. Recovering pointers is a bit trickier and requires to know what an *occurrence* of an arena in a tree is, as well as what it means for a move to *create* such an occurrence, and what it means for a move to *play on* the occurrence of an arena.

Definition 11. *Let T be an $(A \vdash B)$-tree. The* moves *occurring in T are defined inductively: there are no moves occurring in the* DAIMON *rule, there is a single move m occurring in the* LEFT *rule in* (4)*, and the moves that occur in the* RIGHT *rule are all the $m(i)$'s.*

If C is an arena, the occurrences *of C in T are all the occurrences of C in any sequent occurring in T, modulo identification, in the* RIGHT *rule in* (4)*, of all occurrences of arenas in Γ in any of the premises and the conclusion, and in the* LEFT *rule, of all the occurrences of arenas in Γ, A, Δ in the premise and the conclusion.*

In the RIGHT *rule in* (4)*, the move $m(i)$* plays *on A and* creates *$A \cdot m(i)$. In the* LEFT *rule, the move m* plays *on A and* creates *$A \cdot m$.*

Example 5. In the tree of Example 4, there is a single occurrence of the $\mathbb{B}_l \to \mathbb{B}_m$ arena: it exists in the bottommost sequent and the occurrences in all other sequents are equal to it; but there are two occurrences of $\{\mathsf{t},\mathsf{f}\}_l$, created by two different q_l moves.

In the tree of Example 3, the f move is played on (the only occurrence of) $\{\mathsf{t},\mathsf{f}\}_r$ and creates (the only occurrence of) $\varnothing_r$.

Lemmas 142 and 148 from [4] explain how to recover pointers from our structure of plays. They can be stated as follows: each move m is justified by the move that created the arena m was played on. Therefore, to recover pointers from the tree structure, one needs to know which arena each move was played on, and which move created which arena. The first point is easy for the RIGHT rule, as moves are always played on the distinguished arena, and it is also simple in the case of a LEFT rule, because the rule is actually annotated with the index of the arena the move was played on in the formal definition. The second point consists in climbing down the tree towards the root until the arena of interest "disappears": the move that created this arena is the move that "makes it appear" in the sequents (the only exception is the A in $(A \vdash B)$, which exists at the beginning, but is enabled by the initial move that creates the branch of interest, since roots of A are justified in $A \multimap B$ by those of B).

Example 6. Here is the proof tree from Example 3, where the different arenas have been coloured differently to keep track of them more easily, and annotated to the left with the corresponding move:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathbb{B}_l, \{\mathsf{t},\mathsf{f}\}_r, \varnothing_l \vdash \varnothing_r}\,\textsc{right}\ (I=\varnothing)}{\mathbb{B}_l, \{\mathsf{t},\mathsf{f}\}_r, \varnothing_l \vdash}\,\textsc{left}\ (A=\{\mathsf{t},\mathsf{f}\}_r, m=\mathsf{f}_r)}{\mathbb{B}_l, \{\mathsf{t},\mathsf{f}\}_r \vdash \{\mathsf{t},\mathsf{f}\}_l}\,\textsc{right}\ (I=\{\mathsf{t}_l\})}{\mathbb{B}_l, \{\mathsf{t},\mathsf{f}\}_r \vdash}\,\textsc{left}\ (A=\mathbb{B}_l, m=q_l)}{\mathbb{B}_l \vdash \mathbb{B}_r.}\,\textsc{right}\ (I=\{q_r\})$$

f_r, t_l, q_l, q_r

We can see that q_r is played on $\mathbb{B}_r$, which exists at the beginning, so it is initial; q_l is played on $\mathbb{B}_l$, which is the special case mentioned above, so it is justified by the move that created $\{\mathsf{t},\mathsf{f}\}_r$, i.e., q_r; t_l is played on $\{\mathsf{t},\mathsf{f}\}_l$, which is created by q_l; and f_r is played on $\{\mathsf{t},\mathsf{f}\}_r$, which is created by q_r, so we indeed recover the pointers from the play from Example 2.

We now prove the following lemma:

Lemma 2. *F restricts to a functor $F^{\mathbb{V}}\colon \mathbb{V}_{A,B} \to \mathbb{E}^{\mathbb{V}}(A \vdash B)$.*

Proof. Let us take a HON-view $s = (n, f, \varphi)$, and show that $F(s)$ is a view, i.e., a non-branching tree. The proof trees of our sequent calculus can only branch with the use of a RIGHT rule. In that case, we get by the method described above to recover pointers that two Opponent moves are justified by the same Proponent move. But we know by Proposition 2 that all Opponent moves in s are justified by the preceding move, so there can be no two Opponent moves justified by the same Proponent move.

Finally, we need one last lemma to show the tight correspondence between views in both settings:

Lemma 3. *$F^{\mathbb{V}}$ is an equivalence of categories.*

Proof. Since i, i_{HON}, and F are fully faithful, $F^{\mathbb{V}}$ is also fully faithful by left cancellation. Now, to show that $F^{\mathbb{V}}$ is essentially surjective on objects, we simply need to build an antecedent through $F^{\mathbb{V}}$ of any view v. The candidate HON-view is given by taking all moves of v from the root to the top (this is unambiguous since v is non-branching), and the pointers are given by the method described above. All that is left is to verify that the candidate HON-view is indeed a HON-view, which is done by verifying that it is a HON-play and that all Opponent moves are justified by the preceding move. The first point is easy, since the way we have chosen the antecedent may be generalised to any play, and the antecedent verifies all properties of HON-plays, except perhaps for alternation, which is trivial in our case. The second point is obvious by construction.

By putting everything together, we get:

Theorem 1. *The square on the left of* (1) *commutes, F is a ful embedding, and $F^{\mathbb{V}}$ is an equivalence of categories.*

3.2 Comparing Strategies

A useful tool to compare strategies and behaviours in both settings is Guitart's theory of *exact squares* [8], which we now recall.

Definition 12. *A* square *is a natural transformation as in:*

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ {\scriptstyle u}\downarrow & \overset{\phi}{\Longrightarrow} & \downarrow{\scriptstyle v} \\ C & \xrightarrow[g]{} & D. \end{array} \tag{6}$$

Any square yields by restriction a square as on the left below, and so by adjunction a further square as on the right:

$$\begin{array}{ccc} \widehat{A} & \xleftarrow{\Delta_f} & \widehat{B} \\ {\scriptstyle \Delta_u}\uparrow & \overset{\Delta_\phi}{\Longleftarrow} & \uparrow{\scriptstyle \Delta_v} \\ \widehat{C} & \xleftarrow[\Delta_g]{} & \widehat{D} \end{array} \qquad\qquad \begin{array}{ccc} \widehat{A} & \xrightarrow{\Pi_f} & \widehat{B} \\ {\scriptstyle \Delta_u}\uparrow & \overset{\tilde{\phi}}{\Longleftarrow} & \uparrow{\scriptstyle \Delta_v} \\ \widehat{C} & \xrightarrow[\Pi_g]{} & \widehat{D}. \end{array}$$

Definition 13. *A square ϕ is* exact *when $\tilde{\phi}$ is an isomorphism.*

The result we ultimately want to prove to show the tight relationship between both models is Corollary 1, which states that the right square in (1) commutes up to isomorphism. This corollary reduces to exactness of the left square (filled with the identity). In order to prove that it is is exact, let us recall two basic results from Guitart [8]:

Lemma 4. *For any functor $f : A \to B$, the square below left is exact; furthermore, the square below right is also exact if f is fully faithful:*

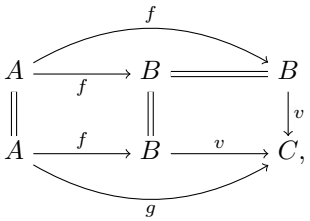
These results appear as 1.14 (1) and (4) in Guitart's paper. The next one appears as 1.8:

Lemma 5. *Exact squares are stable under horizontal composition.*

We put these together to obtain:

Lemma 6. *Any square* (6) *in which ϕ and u are identities and v is fully faithful is exact.*

Proof. We obtain the given square as the horizontal composite



which is exact by Lemma 5, because both squares are exact by Lemma 4.

Lemma 7. *Any square* (6) *in which ϕ is an identity, u is an equivalence, and v is fully faithful is exact.*

Proof. Similar to the previous lemma.

Corollary 1. *The right square of* (1) *commutes up to isomorphism.*

4 Conclusion

Even though Tsukada and Ong's approach differs significantly from ours in terms of presentation, both approaches define similar notions of play (even though our notion of play is slightly looser than the one from traditional game semantics) and views, and the resulting notions of behaviours and innocent strategies are related in a very strong way. This shows that the differences between the two approaches are mainly choices of presentation.

However, Tsukada and Ong's approach goes further than ours: they define a cartesian closed category of arenas and strategies, which allows them to compose strategies. We leave the problem of doing something similar in our setting for future work. Two steps are required to compose strategies, called parallel composition and hiding: the first executes two strategies in parallel, and the second one hides the middle arena. While parallel composition is easy to manipulate in our setting because our game is intrinsically multi-party, hiding could be more difficult to handle.

References

1. S. Abramsky, R. Jagadeesan, and P. Malacaria. Full abstraction for PCF. *Information and Computation*, 163(2):409–470, 2000.
2. P. Baldan, A. Corradini, T. Heindel, B. König, and P. Sobociński. Processes and unfoldings: concurrent computations in adhesive categories. *Mathematical Structures in Computer Science*, 24(4), 2014.
3. C. Eberhart and T. Hirschowitz. Presheaves for Processes and Unfoldings. `http://www.lama.univ-savoie.fr/~eberhart/ProcUnfold.pdf`, 2015. Online extended abstract for a talk at CALCO Early Ideas.
4. C. Eberhart and T. Hirschowitz. Justified sequences in string diagrams: a comparison between two approaches to concurrent game semantics. Preprint, 2016.
5. C. Eberhart and T. Hirschowitz. A yoneda-theoretic reconstruction of concurrent game semantics. Preprint accessible at `http://lama.univ-savoie.fr/~hirschowitz/papers/yoneda-fossacs.pdf`, 2016. Submitted.
6. C. Eberhart, T. Hirschowitz, and T. Seiller. An intensionally fully-abstract sheaf model for π. In *CALCO*, Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015. To appear.
7. D. R. Ghica and A. S. Murawski. Angelic semantics of fine-grained concurrency. In *FoSSaCS*, volume 2987 of *LNCS*, pages 211–225. Springer, 2004.
8. R. Guitart. Relations et carrés exacts. *Annales des Sciences Mathématiques du Québec*, 4(2):103–125, 1980.
9. T. Hirschowitz. Full abstraction for fair testing in CCS. In *CALCO*, volume 8089 of *LNCS*, pages 175–190. Springer, 2013.
10. T. Hirschowitz. Full abstraction for fair testing in CCS (expanded version). *Logical Methods in Computer Science*, 10(4), 2014.
11. 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.
12. A. Joyal, M. Nielsen, and G. Winskel. Bisimulation and open maps. In *LICS*, pages 418–427. IEEE Computer Society, 1993.
13. J. Laird. Game semantics for higher-order concurrency. In *FSTTCS*, volume 4337 of *LNCS*, pages 417–428. Springer, 2006.
14. P.-A. Melliès. Game semantics in string diagrams. In *LICS*, pages 481–490. IEEE, 2012.
15. H. Nickau. Hereditarily sequential functionals. In *LFCS*, volume 813 of *LNCS*, pages 253–264. Springer, 1994.
16. M. Nielsen, G. Plotkin, and G. Winskel. Event structures and domains, part 1. *Theoretical Computer Science*, 13:65–108, 1981.
17. S. Rideau and G. Winskel. Concurrent strategies. In *LICS*, pages 409–418. IEEE Computer Society, 2011.
18. T. Tsukada and C.-H. L. Ong. Nondeterminism in game semantics via sheaves. In *LICS*. IEEE Computer Society, 2015.