Observably Deterministic Concurrent Strategies and Intensional Full Abstraction for Parallel-or
Simon Castellan, Pierre Clairambault, Glynn Winskel

To cite this version:

HAL Id: hal-01886926
https://hal.archives-ouvertes.fr/hal-01886926
Submitted on 3 Oct 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Observably Deterministic Concurrent Strategies and Intensional Full Abstraction for Parallel-or*

Simon Castellan¹, Pierre Clairambault², and Glynn Winskel³

¹ Université Lyon, CNRS, ENS de Lyon, UCB Lyon 1, LIP, Lyon, France
Simon.Castellan@ens-lyon.fr
² Université Lyon, CNRS, ENS de Lyon, UCB Lyon 1, LIP, Lyon, France
Pierre.Clairambault@ens-lyon.fr
³ Computer Laboratory, University of Cambridge, Cambridge, UK
Glynn.Winskel@cl.cam.ac.uk

Abstract

Although Plotkin’s parallel-or is inherently deterministic, it has a non-deterministic interpretation in games based on (prime) event structures – in which an event has a unique causal history – because they do not directly support disjunctive causality. General event structures can express disjunctive causality and have a more permissive notion of determinism, but do not support hiding. We show that (structures equivalent to) deterministic general event structures do support hiding, and construct a new category of games based on them with a deterministic interpretation of aPCFpor, an affine variant of PCF extended with parallel-or. We then exploit this deterministic interpretation to give a relaxed notion of determinism (observable determinism) on the plain event structures model. Putting this together with our previously introduced concurrent notions of well-bracketing and innocence, we obtain an intensionally fully abstract model of aPCFpor.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages

Keywords and phrases Game semantics, parallel-or, concurrent games, event structures, full abstraction


1 Introduction

Plotkin’s parallel-or is, in a sense, the most well-understood of programming language primitives. Recall that it is a primitive \( \text{por} : \mathbb{B} \to \mathbb{B} \to \mathbb{B} \) defined by the three equations:

\[
\text{por}\, t t \bot = t t, \quad \text{por} \, \bot t t = t t, \quad \text{por} \, \text{ff} \, \text{ff} = \text{ff}.
\]

As Plotkin famously proved [16], it is the primitive to be added to the paradigmatic purely functional higher-order programming language PCF in order to get a perfect match (full abstraction) with respect to Scott domains. Since Plotkin’s result, full abstraction has been the gold standard for semanticists. Indeed, whereas an adequate model is always sound to reason about program equivalence; a fully abstract model is also complete: two programs are observationally equivalent if and only if they have the same denotation. Though since Plotkin’s paper fully abstract models of various programming languages have been proposed (often through game semantics, with e.g. state [3, 1], control [5, 14], concurrency [11],

* This work was partially supported by French LABEX MILYON (ANR-10-LABX-0070), and by the ERC Advanced Grant ECSYM.
they are usually quite a bit more complicated than plain Scott domains. Parallel-or is well-understood, in that it has a simple input-output (extensional) behaviour, and its presence in PCF reduces observational equivalence to input-output behaviour.

But is it really so well-understood? While parallel-or is simple extensionally, how is it best understood intensionally? What is the operational behaviour of \( \text{por} M N \)? Any programmer will sense the subtleties in implementing such a primitive. It immediately spawns two threads, starting evaluating \( M \) and \( N \) in parallel. If both terminate with \( \text{ff} \), it terminates with \( \text{ff} \) as well after both have finished. But it suffices that one of the threads returns \( \text{tt} \), for parallel-or to return \( \text{tt} \) immediately, discarding the other. If both return \( \text{tt} \) then one of the threads will be discarded, but which one depends on the scheduler. From an input-output perspective it does not matter which one is selected as they race to produce the same result, but the race nonetheless happens from an operational viewpoint; and a sufficiently intensional semantics will show it. Such behaviour is useful in practice, for instance to speed up deciding the existence of a “good” branch in a search tree by spawning a thread for each branch to explore.

From a game semantics perspective, understanding this combination of racy concurrency and deterministic extensional behaviour has been an unexpected challenge. In earlier work, we dealt with deterministic parallelism using games and strategies based on event structures [17], showing an intensional full abstraction result for a parallel implementation of PCF [8]. We also showed how our tools supported shared memory concurrency [6]. Given that, it is no surprise that the racy behaviour mentioned above can be represented with event structures, yielding an adequate model of e.g. PCF plus parallel-or. But game semantics based on event structures is causal: each event comes with a unique causal history. Dependency is conjunctive: an event can only occur after all its dependencies. This feature is key for the notion of concurrent innocence leading to our definability result [8] and in fact for the very construction of our basic category of games [7]. This has the unfortunate consequence of forcing the strategy for parallel-or to be non-deterministic: the race in the operational behaviour of \( \text{por} \text{tt} \text{tt} \) yields a choice between two events competing to return \( \text{tt} \), each depending on one argument of \( \text{por} \). But interpreting \( \text{por} \) through non-determinism really is a workaround, as appears through the resulting failure of full abstraction.

Constructing a deterministic intensional model for parallel-or has proved demanding. Plays-based models [11] fail as they inline the non-deterministic choice of the scheduler, and cannot express even pure parallelism deterministically. We saw above that plain event structure games do not work either. Giving a deterministic model for parallel-or involves modifying the latter to allow disjunctive causality: a given event may have several distinct causal histories, and an event occurrence does not carry information on its specific causal history. Alternatives to event structures allowing this have existed for a long time: general event structures [18]. However, we will see that they do not, in general, support hiding – which is required to build a category of strategies. We will show that they do support it modulo some further conditions, making them adequate to model parallel-or deterministically.

Beyond the historical twist of giving a fully abstract games model for a language with parallel-or (as game semantics was originally driven by the full abstraction problem for PCF without parallel-or [13, 2]), a treatment of disjunctive causality is indispensable in a complete game semantics framework for concurrency. The following example illustrates how mundane and widespread it is: the causal history of a packet arriving from the network is simply its route. Clearly, introducing an event for each route is best avoided if possible – especially since all further events depending on it will be hereditarily duplicated as well. Deterministic models for disjunctive causality serve two purposes: they allow for (1) a more general, less intensional notion of determinism, and (2) a coarser equivalence relation between strategies.
that abstracts away benign races. These issues, that we address in this paper, have become recurrent obstacles in our research programme; and the historical importance of parallel-or in semantics made it the perfect candidate to test and showcase the solution.

**Contributions.** Concretely, we build an intensionally fully abstract games model of \(a\text{PCF}_{\text{por}}\), an affine version of \(\text{PCF}\) with parallel-or. Our tools are designed with the extension in the presence of symmetry [8] to full \(\text{PCF}_{\text{por}}\) in mind; but presenting them in an affine case allows us to focus on the issues pertaining to parallel-or and disjunctive causality, orthogonal to symmetry. Presenting everything at once in conference format is not reasonable.

Full abstraction for \(a\text{PCF}_{\text{por}}\) has two facets: on the one hand, we need to import the conditions of innocence and well-bracketing from [8], which rely on conjunctive causality. On the other hand, we need a disjunctive notion of determinism. Consequently our model will involve: (1) the standard category \(\text{CG}\) of concurrent games on event structures, and (2) the main novelty of our paper, a new category \(\text{Disj}\) supporting disjunctive determinism. Glueing the two together, we can import in \(\text{CG}\) the notion of determinism from \(\text{Disj}\), thereafter dubbed observational determinism, and prove intensional full abstraction.

**Related work.** Game semantics is a branch of denotational semantics. Originally driven by the full abstraction problem for \(\text{PCF}\), it has grown in the past 25 years into a powerful and versatile methodology to construct compositionally intensional representations of program execution. This lead on the one hand to many full abstraction results (particularly striking in the presence of state as the fully abstract models are then effectively presentable [3, 1]), and on the other hand to applications, ranging from program analysis (model-checking or equivalence checking) to compilation and hardware synthesis [10].

Within game semantics, the present paper is part of a line of work on so-called “truly concurrent” game semantics first pushed by Melliès and colleagues [4, 15], and which advocates the use of causal structures such as event structures and asynchronous transition systems in interactive semantics of programming languages. This line of work has seen a lot of activity in the past five years, prompted in part by a new category of games and strategies on event structures generalizing all prior work, introduced by Rideau and the last author [17]; but also the presheaf-based framework for concurrent games by Hirschowitz and colleagues [12].

Finally, our account of disjunctive causality relies heavily on determinism for hiding to work. The third author and Marc de Visme have developed event structures with disjunctive causality (edc) [9], supporting both disjunctive causality and hiding in a non-deterministic setting. Links between these two approaches are being explored; at the very least, extra axioms would have to be imposed on edcs in order to mimic the work here.

**Outline.** In Section 2 we introduce \(a\text{PCF}_{\text{por}}\) and give its (non-deterministic) event structure games model \(\text{CG}\). In Section 3 we introduce the deterministic model \(\text{Disj}\) for parallel-or. Finally in Section 4 we glue the two, generalize the conditions of [8] and prove full abstraction.

2 Causal game semantics for affine PCF with parallel-or

2.1 The language \(a\text{PCF}_{\text{por}}\)

To alleviate notation, we only take booleans \(\mathbb{B}\) as base type. We start with affine \(\text{PCF}\) (\(a\text{PCF}\)). Its types are either \(\mathbb{B}\), or \(A \rightarrow B\) for some types \(A\) and \(B\). Its terms are those of the
affine λ-calculus with a divergence \( \perp \) and boolean primitives (constants and conditionals):

\[
M, N ::= \lambda x. M \mid M N \mid x \mid \texttt{tt} \mid \texttt{ff} \mid \text{if } M \text{ then } N_1 \text{ else } N_2 \mid \perp
\]

We skip the affine typing rules, which are standard – the typing of if \( M \) then \( N_1 \) else \( N_2 \) is additive, i.e. \( N_1 \) and \( N_2 \) may share resources, and the \( N_i \) have boolean type. The (call-by-name) operational semantics, also standard, yield an evaluation relation \( M \Downarrow v \) between closed terms and \textit{values} (booleans or abstractions). We write \( M \Downarrow \) when there is some \( v \) such that \( M \Downarrow v \), and \( M \Uparrow \) otherwise. Two terms \( \Gamma \vdash M, N : A \) are \textbf{observationally equivalent} \( (M \approx_{\text{obs}} N) \) iff for all contexts \( C[-] \) such that \( C[M], C[N] \) are closed terms of type \( B \), \( C[M] \Downarrow \Leftrightarrow C[N] \Downarrow \).

Recall that an interpretation \( [-] \) in some model \( \mathcal{M} \) is \textbf{fully abstract} if for all \( M, N, M \approx_{\text{obs}} N \iff \llbracket M \rrbracket = \llbracket N \rrbracket \) – it is \textit{intensionally fully abstract} if \( \mathcal{M} \) quotiented by the semantic equivalent of \( \approx_{\text{obs}} \) is fully abstract.

In [16], Plotkin proved that Scott domains are fully abstract not for \textbf{PCF}, but for its extension with the so-called \textbf{parallel-or} operation:

\[
\begin{align*}
\frac{M \Downarrow \texttt{tt} \quad \text{por } M N \Downarrow \texttt{tt}}{\text{por } M N \Downarrow \texttt{tt}}, & \quad \frac{N \Downarrow \texttt{tt}}{\text{por } M N \Downarrow \texttt{tt}}, & \quad \frac{M \Downarrow \texttt{ff} \quad \text{por } M N \Downarrow \texttt{ff}}{\text{por } M N \Downarrow \texttt{tt}}.
\end{align*}
\]

The combinator \textbf{por} is \textbf{not sequential}: it is not the case that it evaluates one of its arguments first. In particular, \textbf{por} \texttt{tt} \Downarrow \texttt{tt} and \textbf{por} \texttt{tt} \Downarrow \texttt{tt}.

### 2.2 Strategies as event structures

As usual in game semantics, computation is a dialogue between the program and its environment. The \textit{moves}, or \textit{events}, are either due to the program (Player, +) or its environment (Opponent, −). They are either \textit{variable calls} (Questions, \( Q \)) or \textit{returns} (Answers, \( A \)). Unlike traditional game semantics, in our line of work such dialogues are \textit{partially ordered}.

Figure 1 presents two concurrent strategies, both playing on (the game for) \( B_1 \rightarrow B_2 \rightarrow B_3 \), where subscripts are for disambiguation. The diagrams are read from top to bottom. The first event in both strategies is an Opponent question on \( B_3 \), initiating the computation. Then Player (the program) starts evaluating in parallel its two arguments (\( q_1 \) and \( q_2 \)). These may return \texttt{tt} or \texttt{ff}, the wiggly line indicating that they cannot reply \textit{both}. Depending on these, Player may eventually answer \( q_3 \). In both diagrams, \( q_3 \) requires \textit{both} arguments to evaluate to \texttt{ff}; however they differ as to the events that trigger an answer \( q_3 \). These diagrams will be made more formal later, but we invite the readers to examine them and convince themselves that the first diagram represents a parallel implementation of the \textit{left or} (diverging if its first argument diverges), whereas the second diagram represents \textit{parallel or}.  

\begin{figure}
\centering
\includegraphics[width=\textwidth]{strategy_diagram}
\caption{Two strategies on \( B_1 \rightarrow B_2 \rightarrow B_3 \), for parallel-or and parallel-or.}
\end{figure}
We will often omit the indices in \( E \). Definition 1. An arena is a triple \((A, \pol_A, \lambda_A)\) where \( A \) is an event structure, \( \pol_A : A \to \{-, +\} \) and \( \lambda_A : A \to \{Q, A\} \) are labelings for polarity and Questions/Answers, such that:

- The order \( \leq_A \) is forest-shaped: for \( a_1, a_2 \leq a \in A \), either \( a_1 \leq a_2 \) or \( a_2 \leq a_1 \).
- The relation \( \vdash_A \) is alternating: if \( a_1 \vdash_A a_2 \), then \( \pol_A(a_1) \neq \pol(a_2) \).
- If \( \lambda_A(a_2) = A \), then there is \( a_1 \vdash a_2 \), and \( \lambda_A(a_1) = Q \).
- An arena \( A \) is race-free: if \( a_1 \vdash a_2 \), then \( \pol_A(a_1) = \pol_A(a_2) \).

An arena \( A \) is negative if all its minimal events have negative polarity.

In the second axiom, we say that the conflict \((e, e')\) is inherited from \((e, e')\). If a conflict \((e, e')\) is not inherited (meaning dependencies of \( e \) and \( e' \) are pairwise compatible, \( e \) is an irreflexive symmetric binary relation called conflict), such that:

- \( \forall e \in E \), \( [e] = \{e' \in E \mid e' \leq_E e\} \) is finite,
- \( \forall e \# e' \), \( \forall e' \leq E \), \( e \# e'' \).

We will often omit the indices in \( \leq_E \), \( \#_E \) if they are obvious from the context.

Event structures. Such diagrams are formalized as event structures. We use here event structures with binary conflict, whereas those of [17, 7] have a more general set of consistent sets. Binary conflict is sufficient for our purposes, and preserved by all operations we need.

Definition 2. A (prime) event structure (with binary conflict, es for short) is \((E, \leq_E, \#_E)\) where \( E \) is a set of events, \( \leq_E \) is a partial order on \( E \) called causality and \( \#_E \) is an irreflexive symmetric binary relation called conflict, such that:

- \( \forall e \in E \), \([e] = \{e' \in E \mid e' \leq_e e\}\) is finite,
- \( \forall e \# e' \), \( \forall e' \leq_e E \), \( e \# e'' \).

We will also use the notation \([e] = [e] \setminus \{e\}\). Between configurations, the covering relation \( x \downarrow \subseteq y \) means that \( y \) is obtained from \( x \) by adding exactly one event: \( y \) is an atomic extension of \( x \). We might also write \( x \downarrow \subseteq \) to mean that \( e \not\in x \) and \( x \cup \{e\} \in \mathcal{C}(E) \). Finally, when drawing event structures, we will not represent the full partial order \( \leq \) but the immediate causality generating it, defined as \( e \vdash e' \) whenever \( e < e' \) and for any \( e \leq e'' \leq e' \), either \( e = e'' \) or \( e'' = e' \).

The diagrams of Figure 1 represent event structures; only displaying immediate causality \( \vdash \) and minimal conflict \( \sim \).

Arenas. Besides causality and conflict, events in Figure 1 carry labels \((q, \ttt, \ff, \ldots)\): formally, those will come from the game, or the arena. Arenas are the semantic representatives of types. They are certain event structures with polarities and Questions/Answers labeling.

Definition 2. An arena is a triple \((A, \pol_A, \lambda_A)\) where \( A \) is an event structure, \( \pol_A : A \to \{-, +\} \) and \( \lambda_A : A \to \{Q, A\} \) are labelings for polarity and Questions/Answers, such that:

- The order \( \leq_A \) is forest-shaped: for \( a_1, a_2 \leq a \in A \), either \( a_1 \leq a_2 \) or \( a_2 \leq a_1 \).
- The relation \( \vdash_A \) is alternating: if \( a_1 \vdash_A a_2 \), then \( \pol_A(a_1) \neq \pol(a_2) \).
- If \( \lambda_A(a_2) = A \), then there is \( a_1 \vdash a_2 \), and \( \lambda_A(a_1) = Q \).
- A is race-free: if \( a_1 \sim a_2 \), then \( \pol_A(a_1) = \pol_A(a_2) \).

An arena \( A \) is negative if all its minimal events have negative polarity.

Conflict aside, our arenas resemble those of [13]: the justification relation traditionally denoted by \( \vdash_A \) is simply \( \vdash_A \). Basic arenas include the empty arena 1, and the arena \([B]\) displayed in Figure 2, often written \( B \) by abuse of notation, for booleans.
We mention here some constructions on arenas. The dual \( A^\perp \) of \( A \) is obtained by taking \( \text{pol}_{A^\perp} = -\text{pol}_A \), and leaving the rest unchanged. The simple parallel composition \( A \parallel B \) is obtained as having events the tagged disjoint union \( \{1\} \times A \cup \{2\} \times B \), and all components inherited. The product \( A \& B \) of negative \( A \) and \( B \) is obtained as \( A \parallel B \), with all events of \( A \) in conflict with events of \( B \). As types will be denoted by negative arenas, we need a negative arena to interpret \( \neg \). This is done by setting \( A \) to depend on (negative) minimal events of \( B \). If \( B \) has at most one minimal event, this is easy:

\begin{definition}
Let \( A, B \) be negative arenas. Assume that \( B \) is well-opened, i.e. \( \min(B) \) has at most one event. If \( B = 1 \), then \( A \rightarrow B = 1 \). Otherwise, \( \min(B) = \{b_0\} \). We define \( A \rightarrow B \) as \( A^\perp \parallel B \), with the additional causal dependency \( (2, b_0) \leq (1, a) \) for all \( a \in A \).
\end{definition}

Defining \( A \rightarrow B \) for well-opened \( B \) is sufficient to interpret \( \text{aPCF}_{\text{por}} \) types, but would be insufficient in the presence of a tensor type; with e.g. \( \mathbb{B} \rightarrow (\mathbb{B} \otimes \mathbb{B}) \). The complication here is due to a disjunctive causality at the level of types: the left \( \mathbb{B} \) is caused by either of the occurrences of \( \mathbb{B} \) on the right. As for parallel-or, the inability of event structures to express that can be worked around by introducing two conflicting copies of the left \( \mathbb{B} \), one for each causal justification – this is done in [6], and is reminiscent of the standard arena construction of [13]. This works well for some purposes, but the informed reader may see why this threatens definability for a concurrent language with tensor: indeed a counter-strategy can then behave differently depending on the cause of an occurrence of the left \( \mathbb{B} \).

We avoid the issue here, and only build \( \rightarrow \) for \( \text{aPCF}_{\text{por}} \) types. Interestingly the issue vanishes in Section 3: \( \text{Disj} \) supports disjunctive causality in both arenas and strategies.

**Prestrategies on arenas.** Strategies are certain event structures labeled by arenas. More formally, the labeling function is required to be a map of event structures:

\begin{definition}
A prestrategy on arena \( A \) is a function on events \( \sigma : S \rightarrow A \) s.t. \( \sigma \) is a map of event structures: it preserves configurations \( (\forall x \in \mathcal{C}(S), \sigma x \in \mathcal{C}(A)) \) and is locally injective \( (\forall s_1, s_2 \in x \in \mathcal{C}(S), \sigma s_1 = \sigma s_2 \implies s_1 = s_2) \).
\end{definition}

Event structures and their maps form a category \( \mathcal{E} \). Figure 1 displays such prestrategies \( \sigma : S \rightarrow A \) – the event structure drawn is \( S \), and events are annotated by their image in \( A \) through \( \sigma \). Strategies, introduced in the next section, will be subject to further conditions.

### 2.3 An interpretation of \( \text{aPCF}_{\text{por}} \) as strategies

Following the methodology of denotational semantics, giving an interpretation of \( \text{aPCF}_{\text{por}} \) consists in constructing a category out of certain prestrategies, with enough structure to interpret \( \text{aPCF}_{\text{por}} \). For lack of space we can only sketch part of the construction, a more detailed account of the construction, originally from [17], can be found in [7].

A prestrategy from \( A \) to \( B \) is a prestrategy \( \sigma : S \rightarrow A^\perp \parallel B \). Given also \( \tau : T \rightarrow B^\perp \parallel C \), we wish to compose them. As usual in game semantics, this involves (1) parallel interaction where the strategies freely communicate, and (2) hiding. We focus on (1) first.

**Interaction.** The interaction of \( \sigma : S \rightarrow A^\perp \parallel B \) and \( \tau : T \rightarrow B^\perp \parallel C \) is an event structure \( T \circ S \), labeled by \( A \parallel B \parallel C \) via \( \tau \circ \sigma : T \circ S \rightarrow A \parallel B \parallel C \). The omitted definition [7] of \( T \circ S \) follows the lines of the third author’s event structure semantics for parallel composition
In CCS [19]. It is uniquely determined (up to iso) as a pullback [7] in \( \mathcal{E} \):

\[
\begin{array}{c}
S \parallel C \\
\bigtriangleup \downarrow \tau_{\sigma} \\
\sigma \parallel A \parallel T \\
\end{array}
\]

It is the “smallest” labeled event structure accommodating the constraints of \( \sigma \) and \( \tau \).

**Hiding.** Composition should yield a prestrategy from \( A \) to \( C \). Accordingly events of \( T \parallel S \) that map to \( A \) or \( C \) are available to the outside world, and called visible. On the other hand, those mapping to \( B \) are private synchronization events, dubbed invisible; that we wish to hide. The proposition below formalizes that event structures are expressive enough for that.

> **Proposition 5.** Event structures support hiding: for \( E \) an event structure and \( V \) any subset of events, then there exists an event structure \( E \downarrow V \) having \( V \) as events, and as configurations exactly those \( x \cap V \) for \( x \in \mathcal{C}(E) \).

The components of \( E \downarrow V \) (causality, conflict) are simply inherited from \( E \).

For \( \sigma \) and \( \tau \) as above, we first form \( T \parallel S \) as \( T \parallel S \downarrow V \) with \( V \) the visible events. The composition \( \tau \circ \sigma : T \parallel S \to A^\perp \parallel C \) is simply the restriction of \( \tau \circ \sigma \); and is a prestrategy.

**A category.** Composition is associative up to isomorphism of prestrategies, where \( \sigma_1 : S_1 \to A \) and \( \sigma_2 : S_2 \to A \) are isomorphic (written \( \sigma_1 \cong \sigma_2 \)) if there is an iso between \( S_1 \) and \( S_2 \) in \( \mathcal{E} \) making the obvious triangle commute. Finally, for any arena \( A \) there is a **copycat prestrategy** \( a_A : \mathcal{C}_A \to A^\perp \parallel A \), which has events and immediate conflict those of \( A^\perp \parallel A \), and with causality that of \( A^\perp \parallel A \) where additionally each positive event is set to depend on its negative counterpart on the other side. Details can be found in [17, 7]. Copycat is idempotent, but is not an identity in general. Rideau and Winskel [17] characterise the prestrategies for which it acts as an identity as the strategies:

> **Theorem 6.** For \( \sigma : S \to A^\perp \parallel B \), \( a_B \circ \sigma \circ a_A \cong \sigma \) iff \( \sigma \) is a **strategy**: it is **receptive** (for \( x \in \mathcal{C}(S) \), if \( x \xrightarrow{\sigma} \subseteq \) there is a unique \( x \xrightarrow{s} \subseteq \) s.t. \( \sigma s = \alpha \)) and **courteous** (if \( s_1 \xrightarrow{\sigma} s_2 \) and \( \text{pol}(s_1) = + \) or \( \text{pol}(s_2) = - \), then \( \sigma s_1 \xrightarrow{A^\perp \parallel B} \sigma s_2 \)).

We mention (see [7]) that \( \mathbb{CG} \) is a compact closed category, i.e. fit to interpret the linear \( \lambda \)-calculus. As \( \text{aPCF}_{	ext{for}} \) is affine, we work in a derived category of negative strategies.

**Negative arenas and strategies.** Negative arenas were defined earlier. We also have:

> **Definition 7.** A strategy \( \sigma : S \to A^\perp \parallel B \) is **negative** iff \( \forall s \in \text{min}(S) \), \( \text{pol}(s) = - \).

There is a subcategory \( \mathbb{CG}_- \) of \( \mathbb{CG} \) with negative arenas as objects, and negative strategies as morphisms. The negativity assumption on strategies ensures that \( 1 \) is terminal (the only negative strategy on \( A^\perp \parallel 1 \), for \( A \) negative, is the empty strategy \( e_A \)), which allows us to interpret weakening. The tensor product is defined as \( A \otimes B = A \parallel B \) on arenas, and as the obvious relabeling \( \sigma_1 \otimes \sigma_2 : S_1 \parallel S_2 \to (A_1 \parallel A_2)^\perp \parallel B_1 \parallel B_2 \) on strategies, for \( \sigma_1 : S_1 \to A_1^\perp \parallel B_1 \) and \( \sigma_2 : S_2 \to A_2^\perp \parallel B_2 \). Besides, \( \mathbb{CG}_- \) has **products**: for \( \sigma : S \to A^\perp \parallel B \) and \( \tau : A^\perp \parallel C \) their pairing \( (\sigma, \tau) : S \otimes T \to A^\perp \parallel (B \parallel C) \) is the obvious labeling, and we have projections \( \varpi_A : \mathcal{C}_A \to (A \parallel B)^\perp \parallel A \) and \( \varpi_B : \mathcal{C}_B \to (A \parallel B)^\perp \parallel B \). For well-opened \( C \), the negative strategies on \( A^\perp \parallel B^\perp \parallel C \) and \( A^\perp \parallel B \) are exactly the same. Because \( A \xrightarrow{\text{e}} B \) is only defined for well-opened \( B \), \( \mathbb{CG}_- \) is not monoidal closed; but well-opened arenas form an **exponential ideal**.
We have seen that \(|A|\) ways. We give their equivalent presentation \([20]\) in terms of visibility, well-bracketing and innocence as in \([8]\); and we will do so in Section 4.2.

Definition 9. \(|A|\) is a set of events, and \(A\) is a set of configurations, which are finite subsets of \(|A|\), such that \(\emptyset \in A\), and satisfying the two further axioms:

- **Disjunctive causality.** As we pointed out in the introduction, the non-determinism of parallel-or comes from the incapacity of our version of event structures, sometimes referred to as prime event structures for disambiguation, to express disjunctive causality. So the reader may wonder: why do we bother with prime event structures at all? After all, there are alternatives. General event structures \([18]\) allow events to be enabled in several distinct ways. We give their equivalent presentation \([20]\) in terms of configuration families:

- **Definition 9.** A configuration family (cf) is a pair \((|A|, A)\) (often just written \(A\)) where \(|A|\) is a set of events, and \(A\) is a set of configurations, which are finite subsets of \(|A|\), such that \(\emptyset \in A\), and satisfying the two further axioms:
Completeness. For \( x,y \in \mathcal{A} \), if \( x \uparrow y \) (they are compatible, i.e. there exists \( z \in \mathcal{A} \) such that \( x \subseteq z \) and \( y \subseteq z \)), then \( x \cup y \in \mathcal{A} \).

Coincidence-freeness. If \( x \in \mathcal{A} \), for all distinct \( e_1,e_2 \in x \) there exists \( y \subseteq x \) such that \( y \in \mathcal{A} \) and \( e_1 \in y \Leftrightarrow e_2 \notin y \).

For every (prime) event structure \( A \), \( \mathcal{E}(A) \) is a configuration family. However configuration families are more general. In a configuration family, the same event can occur for different reasons. For instance, on the set of events \( \{ \ominus_1, \oplus_2, \ominus \} \), Figure 4 represents a configuration family where the event \( \oplus \) can occur either because of \( \ominus_1 \) or \( \ominus_2 \), represented symbolically as in the diagram on the below – importantly, an occurrence of \( \ominus \) carries no data on its effective cause.

\[
\ominus_1 \lor \ominus_2 \lor \ominus
\]

With the same symbolic representation, the causally disjunctive nature of parallel-or is made explicit in Figure 5. It seems clear that this is a more accurate description of parallel-or. In particular, it is deterministic, in a sense to be made formal in the next section.

Are configuration families sufficiently expressive as a basis to construct a category of strategies, as we did with prime event structures in the previous section? Recall that for that we used two properties of event structures: that the corresponding category \( \mathcal{E} \) has pullbacks (for interactions), and then that event structures support hiding. We will show very soon that the first criterion holds. However, we run into an issue for the second:

\textbf{Proposition 10.} Configuration families do not support hiding. There is a cf \( A \) and \( V \subseteq |A| \) such that the set \( \{ x \cap V \mid x \in \mathcal{A} \} \) is not a configuration family.

\textbf{Proof.} Set \( |A| = \{a,b,c,d\} \), and configurations those specified by: \( d \) is caused either by \( a \), or by \( b \) and \( c \) together). With \( V = \{b,c,d\} \), the obtained hidden set fails completeness: it contains \( \{b\}, \{d\}, \{b,c,d\} \) but not \( \{b,d\} \).

\[
\begin{array}{c}
a \\ \lor \\
\searrow \\
\downarrow \\
\nearrow \\
c \\
\searrow \\
\downarrow \\
\nearrow \\
d
\end{array}
\]

It is part for this reason that prime event structures are used when building categories of games and strategies – as indeed, hiding is crucial. Removing either coincidence-freeness or completeness loses the correspondence with general event structures [20], and leads to various pathologies further down the road – for instance we lose the key lemma:

\textbf{Lemma 11.} Let \( A \) a set of configurations on \( |A| \) satisfying completeness. Then, it is coincidence-free iff for any \( x,y \in \mathcal{A} \) s.t. \( x \subseteq y \), there is a covering chain in \( \mathcal{A} \): \( x \subset \ldots \subset y \).
3 Disjunctive deterministic games

In this section we introduce the main contribution of the paper, a category $\textup{Disj}$ of disjunctive deterministic strategies supporting the interpretation of parallel-or. This relies on a notion of deterministic configuration families, and the observation that those do support hiding.

3.1 Deterministic configuration families

First we adjoin cfs with polarities, and define determinism (for Player).

Definition 12. A cf with polarities (cfp) (resp. partial polarities (cfpp)) is a cf $\mathcal{A}$ with $\text{pol}_\mathcal{A}: |\mathcal{A}| \to \{-, +\}$ (resp. $\text{pol}_\mathcal{A}: |\mathcal{A}| \to \{-, 0, +\}$).

A cfpp $\mathcal{A}$ is deterministic (dcfp/dcftp) iff for all $x \subseteq \mathcal{A}$ y and $x \subseteq z$, we have $y \cup z \in \mathcal{A}$ – we write $x \leq^{\mathcal{A}} y$ (resp. $x \subseteq^{\mathcal{A}} y$, $x \subseteq^{\mathcal{A}} y$) to mean that $x \subseteq y$ and $\text{pol}(y \setminus x) \subseteq \{0, +\}$ (resp. $\text{pol}_\mathcal{A}(y \setminus x) \subseteq \{-\}$; $\text{pol}_\mathcal{A}(y \setminus x) \subseteq \{+\}$).

This means that only Opponent makes irreversible choices regarding the evolution of the play. We show that dcfps support hiding of neutral events. In the sequel, given a dcfpp $\mathcal{A}$, $\mathcal{V}$ always denotes the set of non-neutral events in $|\mathcal{A}|$. We write $\mathcal{A}_0$ for the candidate cf supporting the configurations those of the form $x_i = x \cap V$ for $x \in \mathcal{A}$. The following key lemma is proved by successive applications of determinism and completeness.

Lemma 13. For $x \in \mathcal{A}_0$, $y \in \mathcal{A}$ such that $x \subseteq y_i$, for any $x' \in \mathcal{A}$ a witness for $x$ (i.e. $x'_i = x$), there is $y \leq y' \in \mathcal{A}$ such that $x' \subseteq y'$.

From Lemma 13 follows that for $x, y \in \mathcal{A}$, if $x_i \uparrow y_i$ in $\mathcal{A}_0$, then $x \uparrow y$ in $\mathcal{A}$. Using that observation together with Lemma 13 and determinism, it is straightforward to prove:

Proposition 14. For $\mathcal{A}$ a dcfpp, $\mathcal{A}_0$ is a dcf.

3.2 Deterministic disjunctive strategies and composition

A pregame is simply a cfp – for $\mathcal{A}, \mathcal{B}$ cfps, $\mathcal{A}^\perp$ has the same events and configurations as $\mathcal{A}$ but inverted polarity. We will also write $\mathcal{B}^0$ for the cfpp with the same configurations as $\mathcal{B}$ but polarity set as globally 0. The cf $\mathcal{A} \parallel \mathcal{B}$ has the tagged disjoint union, and configurations $x_A \parallel x_B$ for $x_A \in \mathcal{A}$, $x_B \in \mathcal{B}$; $\mathcal{A} \& \mathcal{B}$ has the same events as $\mathcal{A} \parallel \mathcal{B}$, but only those configurations with one side empty.

Unlike in Definition 4, disjunctive prestrategies are simply substructures of the (pre)games.

Definition 15. A prestrategy on $\mathcal{A}$ is a dcfp $\sigma$ on $|\mathcal{A}|$, such that $\sigma \subseteq \mathcal{A}$ – written $\sigma : \mathcal{A}$.

Example 16. On events (with polarities) $\{q^-, t^+, f^+\}$, $\mathcal{E}(\mathcal{B})$ is a cfp – that by abuse of notation we still write $\mathcal{B}$. Then, Figure 5 denotes a prestrategy on $\mathcal{B}^\perp \parallel \mathcal{B}^\perp \parallel \mathcal{B}$.

Though disjunctive prestrategies are not primarily defined as maps, it will be helpful in composing them that they can be. Given two cfs $\mathcal{A}$ and $\mathcal{B}$, a map from $\mathcal{A}$ to $\mathcal{B}$ is a function on events $f : |\mathcal{A}| \to |\mathcal{B}|$ which preserves configurations and is locally injective. Configuration families and their maps forms a category $\mathcal{Fam}$. Note that if $\mathcal{A}$ is an event structure, $\mathcal{E}(\mathcal{A})$ is a configuration family on $|\mathcal{A}|$. The definition of maps of cfs is compatible with that of maps of event structures, making $\mathcal{E}(\cdot) : \mathcal{E} \to \mathcal{Fam}$ a full and faithful functor. Finally, a prestrategy $\sigma : \mathcal{A}$ on $\mathcal{A}$ can be regarded as an identity-on-events $\mathcal{Fam}$-morphism $\sigma : \mathcal{A}$.

Like $\mathcal{E}$, $\mathcal{Fam}$ has pullbacks. The interaction of $\sigma$ on $\mathcal{A}^\perp \parallel \mathcal{B}$ and $\tau$ on $\mathcal{B}_0 \parallel \mathcal{C}$ is the pullback of identity-on-events maps $\sigma \parallel \mathcal{C} \to \mathcal{A} \parallel \mathcal{B} \parallel \mathcal{C}$ and $\mathcal{A} \parallel \tau \to \mathcal{A} \parallel \mathcal{B} \parallel \mathcal{C}$. 

Proposition 17. The pullback above is (up to iso) the cf $\tau \odot \sigma$ with events $|A| \parallel |B| \parallel |C|$ and configurations those $x \in (\sigma \parallel C) \cap (A \parallel \tau)$ that are secured: for distinct $p_1, p_2 \in x$ there is $x \supseteq y \in (\sigma \parallel C) \cap (A \parallel \tau)$ s.t. $p_1 \in y \iff p_2 \notin y$.

Equivalently, $x \in (\sigma \parallel C) \cap (A \parallel \tau)$ is secured iff it has a covering chain, i.e. a sequence: $0 = x_0 \vartriangleleft x_1 \vartriangleleft \ldots \vartriangleleft x_n = x$ where for all $0 \leq i \leq n$, $x_i \in (\sigma \parallel C) \cap (A \parallel \tau)$.

We regard $\tau \odot \sigma$ as a cfpp by setting as polarities those of $A^\parallel \parallel B^0 \parallel C$. To use Proposition 14 and finish defining composition, $\tau \odot \sigma$ needs to be deterministic: a sufficient condition for that is that $\sigma$ and $\tau$ are receptive. A prestrategy $\sigma$ on $A$ is receptive iff for all $x \in \sigma$, if $x \cup \{ a^- \} \in A$ then $x \cup \{ a^- \} \in \sigma$.

Proposition 18. If $\sigma$ and $\tau$ are receptive prestrategies, then $\tau \odot \sigma$ is a deterministic cfpp. Then, $\tau \odot \sigma = (\tau \odot \sigma)_I$ is a receptive prestrategy on $A^\parallel \parallel C$.

Hence, composition is well-defined on receptive prestrategies – it is also associative. To get a category, we now prove the copycat strategy to be an identity.

3.3 Copycat and the compact closed category $\text{Disj}$

We cannot replicate in pregames the definition of copycat sketched in Section 2.3. However, as observed in [22, 7], if $A$ is an arena, the configurations of $C_A$ are those configurations of $A^\parallel \parallel A$, necessarily of the form $x_1 \parallel x_r$, such that every positive event of $x_r$ (w.r.t. $A$) is already in $x_1$, and every positive event in $x_1$ (w.r.t. $A^\parallel$) is already in $x_r$. In other words, $x_1 \supseteq_A x_1 \cap x_r \subseteq_A x_r$, written $x_r \subseteq_A x_1$ in [22, 7] and referred to as the “Scott order”.

Accordingly, on a pregame $A$, given $x, y \in A$ we write $x \subseteq_A y$ iff $x \cap y \in A$ and the relation above holds. The candidate prestrategy copycat $\alpha_A$ comprises all $x \parallel y \in A^\parallel \parallel A$ s.t. $y \subseteq_A x$. Indeed $\alpha_A$ is a configuration family and is receptive, but prestrategies must be deterministic. It turns out that as in [21], $\alpha_A$ is only deterministic when $A$ is race-free:

Proposition 19. Let $A$ be a pregame. Then, $\alpha_A$ is a prestrategy iff $A$ is race-free: for all $x, y, z \in A$ such that $x \subseteq_A y$ and $x \subseteq_A z$, we have $y \parallel_A z$.

We aim to reproduce Theorem 6 in this new setting, and characterise the prestrategies left invariant under composition with copycat. However, there is a new subtlety here: for arbitrary $A$, $\alpha_A$ might not even be idempotent!

Example 20. Consider the cfpp on events $A = \{ \bigoplus_1, \bigoplus_2, \bigoplus_3 \}$ given on the below:

\[
\begin{array}{c}
\bigoplus_1 \text{ or } \bigoplus_2 \\
\bigoplus_3
\end{array}
\]

This is a race-free pregame, so by Proposition 19, $\alpha_A \odot \alpha_A$ is a prestrategy on $A^\parallel \parallel A$. However, it is distinct from $\alpha_A$.

Indeed, the following configuration of $A^\parallel \parallel A^0 \parallel A$ belongs to $\alpha_A \odot \alpha_A$:

\[
\begin{array}{c|c|c}
\bigoplus_1 & \bigoplus_2 & \bigoplus_1 \\
\bigoplus_3 & \bigoplus_3 & \bigoplus_3
\end{array}
\]

After hiding, there is a change in the causal history of $\bigoplus_3$, not authorized by $\alpha_A$ but authorised by $A$: it is caused by $\bigoplus_2$ on the left, but $\bigoplus_1$ on the right. This issue comes from the fact that in $A$, the same event can be caused either by a positive or a negative move.
Such behaviour will be banned in games:

- **Proposition 21.** If $\mathcal{A}$ is a race-free pregame, the following are equivalent: (1) $\sigma_{\mathcal{A}}$ is idempotent, (2) $\equiv_{\mathcal{A}}$ is a partial order, (3) $\mathcal{A}$ is co-race-free: for all $x, y, z \in \mathcal{A}$ with $x \geq y, x \geq z$, we have $y \cap z \in \mathcal{A}$.

A game will be a race-free, co-race-free pregame. Copycat on any game is an idempotent prestrategy – strategies are those prestrategies that compose well with copycat. We prove:

- **Theorem 22.** Let $\sigma$ be a receptive prestrategy on $A^\perp \parallel B$. Then, $\sigma \circ \sigma \circ \sigma_{\mathcal{A}} = \sigma$ iff $\sigma$ is courteous: for any $x \land x \cup \{a_1^+\} - x \cup \{a_2^+, a_2^-\}$ in $\sigma$, if $x \cup \{a_2^+\} \in A^\perp \parallel B$, then $x \cup \{a_2^+\} \in \sigma$ as well. In this case, $\sigma$ is a strategy, written: $\sigma : A^\perp \parallel B$.

It follows that there is a category $\text{Disj}$ with games as objects, and strategies $\sigma : A^\perp \parallel B$ as morphisms from $A$ to $B$. It is fairly easy to show that just as $\CG$, this category is compact closed. But unlike $\CG$, $\text{Disj}$ supports a deterministic interpretation of parallel-or: indeed the set of configurations of $B^\perp \parallel B^\perp \parallel B$ denoted by the diagram of Figure 5 is a strategy.

### 3.4 An SMCC and deterministic interpretation of $a\text{PCF}_{\text{por}}$

As for $\CG$, $\text{Disj}$ lacks structure to interpret $a\text{PCF}_{\text{por}}$: the family of bottom strategies $e_A : A^\perp \parallel 1$ (where again 1 is the empty game) fails naturality. As before, we hence restrict $\text{Disj}$ to a subcategory of negative games and strategies.

- **Definition 23.** A cfp $A$ is negative if any non-empty $x \in A$ includes a negative event.

Copycat on negative $A$ is negative and negative strategies are stable under composition; so there is a subcategory $\text{Disj}_-$ of $\text{Disj}$ with negative games as objects and negative strategies as morphisms, inheriting a symmetric monoidal structure from $\text{Disj}$. Moreover 1 is terminal, and $\text{Disj}_-$ has products given by $A \& B$.

To prove the monoidal closure, as in $\CG$, we have to deal with the fact that for $A$ and $B$ negative, $A^\perp \parallel B$ is not necessarily negative, so we define:

- **Definition 24.** Let $A$ and $B$ be two negative games. The game $A \dashv B$ has the same events (and polarity) as $A^\perp \parallel B$, but non-empty configurations those $x_A \parallel x_B \in A^\perp \parallel B$ such that $x_B$ is non-empty (and hence includes a negative event).

Recall from Definition 3 that the arrow arena $A \dashv B$ was only defined for $B$ well-opened (i.e. with at most one minimal event). This was to avoid constructing arenas for types like $B \rightarrow (B \& B)$ (invalid for $a\text{PCF}_{\text{por}}$, but valid in an extension with a tensor type), where the left hand side $B$ can be opened because of either of the right hand side occurrences of $B$. In $\text{Disj}$, when constructing $A \rightarrow B$ there is no well-openness condition on $B$: exploiting disjunctive causality we can express that $A$ is opened only after some event of $B$ – it does not matter which one. The negative strategies in $A^\perp \parallel B^\perp \parallel C$ and $A^\perp \parallel B \rightarrow C$ are the same, from which (using the compact closed structure of $\text{Disj}$) it follows that $\text{Disj}_-$ is monoidal closed, instead of only having an exponential ideal. Like $\CG_-$, it supports an adequate interpretation of $a\text{PCF}_{\text{por}}$. Unlike $\CG_-$, its strategies are deterministic.

### 4 Glueing $\CG_-$ and $\text{Disj}_-$, and full abstraction

By moving from $\CG_-$ to $\text{Disj}_-$, we gain determinism. However, $\text{Disj}_-$ is not intensionally fully abstract – there are some strategies that distinguish syntactically undistinguishable terms.
of \( \text{aPCF}_{\text{per}} \). For instance, one has a strategy on \((B \to B \to B) \to B\) that calls its argument, feeds it \( \mathsf{t} \) as first argument, and \( \mathsf{t} \) as second argument only if the first argument has been evaluated; then copies the final result to toplevel. Doing so, it distinguishes the observationally equivalent \( \lambda x y. \) if \( x \) then \( (y \text{ then } \mathsf{t} \text{ else } \bot) \) else \( \bot \) and \( \lambda x y. \) if \( y \) then \( (x \text{ then } \mathsf{t} \text{ else } \bot) \) else \( \bot \).

Getting rid of such strategies is the responsibility of the concept of \textit{innocence} in Hyland-Ong games [13]. But the \textit{P-views} of innocent strategies carry precisely the \textit{causal} information that we have lost when moving from \( \text{CG}_- \) to \( \text{Disj}_- \)! This causal information was crucial in [8] to give concurrent versions of well-bracketing and innocence, so as to capture the behaviour of parallel \( \text{PCF} \) strategies. So, the deterministic account of \( \text{aPCF}_{\text{per}} \) is not enough; the causal (where innocence and well-bracketing are defined) and deterministic models should be related, to establish in what sense the causal model is already “observably” deterministic.

### 4.1 A glued games model

For any arena \( A \) (Definition 2), \( \mathcal{C}(A) \) is a game – it is race-free and co-race-free. Moreover, arena constructions and game constructions match: for any \( A, B \), \( \mathcal{C}(A \parallel B) = \mathcal{C}(A) \parallel \mathcal{C}(B) \); for negative \( A, B \) we have \( \mathcal{C}(A \& B) = \mathcal{C}(A) \& \mathcal{C}(B) \), and if \( B \) is well-opened, \( \mathcal{C}(A \to B) = \mathcal{C}(A) \to \mathcal{C}(B) \). Therefore, we will just take the objects of our glued game model to be arenas. The strategies, though, will be strategies in \textit{both categories}.

**Definition 25.** Let \( A \) be an arena. A strategy \( \sigma : S \to A \) is \textbf{observationally deterministic (\textit{odet})} if (1) the display of \( \sigma \), \( O(\sigma) = \{ \sigma x \mid x \in \mathcal{C}(S) \} \) is a disjunctive deterministic strategy on \( \mathcal{C}(A) \) (in \textit{Disj}). It follows then that \( \sigma \) is also a \textbf{Fam-morphism} \( \sigma : \mathcal{C}(S) \to O(\sigma) \). We also ask that it has (2) the \textbf{configuration extension property}: for any \( x \in \mathcal{C}(S) \), if \( \sigma x \subseteq y \in O(\sigma) \), then there is \( x \subseteq x' \in \mathcal{C}(S) \) such that \( \sigma x' = y \).

The configuration extension property ensures that two causal realizations of the same displayed configuration have bisimilar futures. The display operation yields a coarser equivalence on strategies: odet strategies \( \sigma, \tau : A \), are \textbf{display-equivalent}, written \( \sigma \approx \tau \), if \( O(\sigma) = O(\tau) \). This coarser equivalence is key to ensure soundness of our interpretation, as the interpretation in \( \text{CG}_- \) of \( \text{por} \mathsf{tt} \) and \( \mathsf{tt} \) are \textit{not} isomorphic, but only display-equivalent.

**Theorem 26.** There is a compact closed category \( \text{Odet} \) with arenas as objects and odet strategies up to \( \approx \) as morphisms; with a symmetric monoidal subcategory \( \text{Odet}_- \) of negative arenas and negative strategies with products and 1 terminal. It has well-opened arenas as an exponential ideal. Finally, \( \text{Odet}_- \) supports an adequate interpretation of \( \text{aPCF}_{\text{per}} \).

**Proof.** For odet \( \sigma \) and \( \tau \), \( O(\tau \circ \sigma) = O(\tau) \circ O(\sigma) \): \( \subseteq \) is direct, \( \supseteq \) exploits the extension property for \( \sigma \) and \( \tau \). So, \( O(\tau \circ \sigma) \) is a deterministic disjunctive strategy, and it has the extension property – hence \( \tau \circ \sigma \) is odet. In general, \( O \) links the structure of \( \text{CG} \) to that of \( \text{Disj} \), forming \( \text{Odet} \). We sketch adequacy.

**Soundness.** For all \( M \downarrow v \), \( [M] \models [v] \) (by induction on the evaluation derivations).

**Adequacy.** We prove by induction on the size of \( M \) that if \( M \uparrow \), then \([M]\) is bottom. The size is an adequate measure because as \( \text{aPCF}_{\text{per}} \) is affine, substitution is non-copying and each induction step is on a strictly smaller term.

As a result, interpretations in \( \text{CG}_- \) and \( \text{Disj} \) are also adequate. We have accommodated the causal representation of programs permitted by \( \text{CG} \) and the determinism of \( \text{Disj} \). Now, it remains to import the causal conditions on strategies of [8], and prove full abstraction.
4.2 Conditions

We now recall innocence and well-bracketing, introduced in [8]. Formulated for deterministic strategies (in the sense of CG_−), those would not suffice to prove full abstraction for a genuinely non-deterministic language. For that, further conditions are needed to ensure the locality of conflicts – those will appear in the first author’s forthcoming PhD thesis. However, we will show in Lemma 31 that for aPCF_par, distinguishable strategies can be distinguished via deterministic (in the sense of CG_−) contexts, so these simpler conditions will suffice.

All our conditions rely crucially on the notion of grounded causal chain.

Definition 27. A grounded causal chain (gcc) in S is a sequence \( p = \rho_0 \rightarrow \ldots \rightarrow \rho_n \) where \( \rho_0 \) is minimal. The set of gccs of \( S \) is written gcc(S).

By courtesy, gccs of strategies on arenas are always alternating. They give a notion of thread of a concurrent strategy. Two gccs \( \rho, \rho' \in \text{gcc}(S) \) are forking when \( \rho \cup \rho' \) is consistent in \( S \), and there is \( k \in \mathbb{N} \) s.t. \( \rho_i = \rho'_j \) for \( i < k \) and \( \{\rho_i\}_{i \geq k} \) and \( \{\rho'_j\}_{j \geq k} \) are non-empty and disjoint. They are negatively forking when \( \text{pol}(\rho_{k+1}) = \text{pol}(\rho'_{k+1}) = \vdash \), positively forking otherwise. Two forking gccs \( \rho, \rho' \) are joined at \( s \in S \) when \( \rho_\omega \rightarrow s \) and \( \rho'_\omega \rightarrow s \). (\( \rho_\omega \) refers to the last event of \( \rho \)) as shown in this picture:

\[
\rho_0 \rightarrow \ldots \rightarrow \rho_k \xrightarrow{\rho_{k+1}} \ldots \rightarrow \rho_\omega \rightarrow s
\]

Innocence. Innocence enforces independence between threads forked by Opponent.

Definition 28. Let \( \sigma : S \rightarrow A \) be a strategy on an arena \( A \). It is innocent when it is:

Visible: The image \( \sigma \rho \) of a gcc (regarding \( \rho \in \text{gcc}(S) \) as a set) is a configuration of \( A \).

Preinnocent: Two negatively forking gccs are never joined.

Preinnocence is a locality condition, forcing Player to deal independently with threads forked by Opponent – in the sequential case, it coincides with Hyland-Ong innocence. In the concurrent case though, it is not by itself stable under composition; that is where visibility comes in. Together they are preserved under composition and under all the operations on strategies involved in the interpretation of aPCF_par in CG_− (and hence Odet_−).

Well-bracketing. Traditionally, well-bracketing in game semantics rules out strategies that behave like a control operator, manipulating the call stack. It exploits the question/answer labelling, reminiscent of the function calls/returns. In arenas from aPCF_par types, answers to the same question are always conflicting, so every consistent set has at most one answer to any question. A consistent set \( X \) is complete when it has exactly one answer to any question. A question is pending in a set \( X \) if it has no answer in \( X \) and maximally so in \( X \).

Definition 29. A strategy \( \sigma : S \rightarrow A \) is well-bracketed when (1) if \( a \in S \) answers \( q \in S \), then \( q \) is pending in \( [a] \) and (2) for \( \rho, \rho' \in \text{gcc}(S) \) forking at \( \rho_k = \rho'_k \), and joined, the segments \( \rho_{>k} \) and \( \rho'_{>k} \) must be complete.

The affinity condition of [8] comes for free here, thanks to the conflict in \( \mathbb{B} \). Well-bracketing is proved stable under composition and other operations in [8]. A aPCF_par-strategy is a negative, innocent, well-bracketed and odet strategy.

Theorem 30. There is a symmetric monoidal category PorStrat of negative arenas and aPCF_par-strategies, with products, a terminal object and an exponential ideal comprising \( \mathbb{B} \). Moreover, the interpretation of aPCF_par in Odet_− factors through PorStrat \( \subseteq \) Odet_−.

From now on, all strategies will be assumed to be aPCF_par-strategies.
4.3 Intensional full abstraction

Two strategies $\sigma, \tau : A$ are observationally equivalent ($\sigma \equiv_{\text{obs}} \tau$) when for all strategies $\alpha : A \perp \parallel B$, $\alpha \circ \sigma \approx \alpha \circ \tau$. In fact, there is no need to quantify over all strategies. A path-strategy is a strategy $\sigma : S \to A$ such that there exists a configuration $x \in \mathcal{C}(S)$ that contains all the positive moves of $S$. Any distinguishing strategy yields by restriction a distinguishing path-strategy – this would fail without affinity, as the restriction would fail uniformity [8], and enforcing uniformity would make it non-deterministic.

▶ **Lemma 31.** Two distinguishable strategies can be distinguished by a path-strategy.

▶ **Lemma 32.** Every path-strategy can be defined by a term of aPCF up to $\equiv_{\text{obs}}$.

Proof. As in [8] since path-strategies are deterministic – the only difference is the necessity to distribute the context among the subterms to ensure affine typing. ◄

A corollary is that in an affine setting, por adds no distinguishing power: two aPCF$_{\text{per}}$ terms are observationally equivalent if and only if they cannot be distinguished by a context from affine PCF without por. Intensional full abstraction follows by standard techniques:

▶ **Theorem 33.** The interpretation of aPCF$_{\text{per}}$ into PorStrat is intensionally fully abstract: it preserves and reflects observational equivalence (see Section 2.1).

5 Conclusion

This leaves many avenues for further investigation. On the semantic front, we plan among other applications to exploit Odet to model non-interfering concurrent languages. On the foundational front we need to understand better how the present approach relates to the treatment of disjunctive causality in edcs [9].

Acknowledgments. We are grateful to Tamás Kispéter for interesting discussions about strategies as general event structures.

References


