An Intensionally Fully-abstract Sheaf Model for pi

Clovis Eberhart, Tom Hirschowitz, Thomas Seiller

Fully-abstract concurrent games for π

Clovis Eberhart[1], Tom Hirschowitz[2⋆], and Thomas Seiller[3⋆]

[1] ENS Cachan [2] CNRS and Université de Savoie [3] INRIA

Abstract. We define a semantics for Milner's pi-calculus, with three main novelties. First, it provides a fully-abstract model for fair testing equivalence, whereas previous semantics covered variants of bisimilarity and the may and must testing equivalences. Second, it is based on reduction semantics, whereas previous semantics were based on labelled transition systems. Finally, it has a strong game semantical flavor in the sense of Hyland-Ong and Nickau. Indeed, our model may both be viewed as an innocent presheaf semantics and as a concurrent game semantics.

1 Introduction

The π-calculus [11, 37] was designed as a basic model to reason about concurrent programs, as the λ-calculus for functional programs. Its behavioural theory features several notions of equivalence, including variants of bisimilarity, contextually-defined congruences, and testing equivalences [8]. Its denotational semantics has been thoroughly investigated [12, 46, 14, 13, 42, 6, 22, 7, 38]. This paper introduces a new denotational semantics for π with three main novelties.

Fair testing equivalence First, our semantics provides a fully-abstract model for *fair* testing equivalence [39, 4], whereas previous work covers variants of bisimilarity, and the *may* and *must* testing equivalences.

Originally introduced for CCS-like calculi, fair testing equivalence reconciles the good properties of observation congruence [36] w.r.t. divergence, and the good properties of previous testing equivalences [8] w.r.t. choice. The idea is, as in most testing semantics, that two processes are equivalent when they pass the same *tests*. A process P passes the test T iff their parallel composition $P|T$ never loses the ability of playing some special 'tick' action, even after any reduction sequence. Fair testing is, to our knowledge, one of the finest testing equivalences.

Cacciagrano et al. [5], beyond providing an excellent survey on fairness, adapt the definition to π and study approximations of it. Their definition is not a congruence, for essentially the same reason as for standard bisimilarity [45]. We thus refine it by allowing tests to rename channels, which yields a congruence.

⋆ Partially funded by the French ANR projets blancs 'Formal Verification of Distributed Components' PiCoq ANR 2010 BLAN 0305 01 and 'Realizability for classical logic, concurrency, references and rewriting' Récré ANR-11-BS02-0010.

Reductions vs. labelled transitions A second novelty is that our model is based on the *reduction* semantics of π, while all others, to our knowledge, are based on its standard *labelled transition system* (LTS). The tension between the two has been the subject of substantial research [43]. Briefly, reduction semantics is simple and intuitive, but it operates on equivalence classes of terms (under so-called *structural* congruence). On the other hand, designing LTSs is a subtle task, rewarded by easier, more structural reasoning over reductions. LTSs are generally perceived as less primitive than reduction semantics, although they are often preferred for practical reasons.

Most LTSs for π distinguish two kinds of transitions for output, respectively called *free* and *bound*. This distinction is crucial in all models mentioned above, which rely on the standard LTS semantics, but not in our model. Actually, as explained below in the proof sketch for Theorem 4, our model suggests a new LTS for π which does not distinguish between free and bound output, in a way reminiscent of the carefully-crafted LTS of Rathke and Sobociński [43].

Innocent presheaves = concurrent strategies The third novelty of our semantics is its strong *game semantical* flavor, in the sense of Hyland-Ong [26] and Nickau [40], whereas most previous work was based on coalgebras, bialgebras, presheaves, event structures, or graph rewriting. Game semantics was designed to provide denotational semantics for functional programming languages, but also led to fully-abstract models for impure features like references or control operators. A few authors have defined game semantics for concurrent languages, as discussed below.

A particular feature of our game is its truly 'multi-player' aspect. An immediate benefit of this is that parallel composition, usually interpreted as a complex operation, is here just a move in the game, allowing a player to *fork* into two. On the other hand, considering a multi-player game opens the door to undesirable strategies, which are then ruled out by imposing an *innocence* condition, very close in spirit to game semantical innocence [26, 40]. Indeed, it amounts to requiring that players interact according only to their local *view* of the play.

A particular feature of our *strategies* helps dealing with π's *external choice* operator, which allows processes to accept the same action on some channel a in several ways. E.g., any process of the shape $a(x).P + a(x).Q$ may input on a in two ways, resp. leading to P and Q. In order to model this, we use the fact[1] that presheaves on the poset P of plays ordered by prefix are actually a form of concurrent strategies. Indeed, a strategy is traditionally a prefix-closed set of plays. This is equivalent to a functor $F\colon P^{op} \to \mathsf{2}$, where $\mathsf{2}$ is the poset $0 \to 1$, viewed as a category: the accepted plays $p \in P$ are those such that $F(p) = 1$. The action of F on morphisms ensures prefix closedness, because for any plays $p \leq q$, we must have $F(q) \leq F(p)$; hence, if q is accepted, then so is p. Now, $\mathsf{2}$ embeds fully and faithfully into sets by $0 \mapsto \emptyset$ and $1 \mapsto 1$ (the latter denoting any singleton set $\{\star\}$): this is as if strategies could accept plays in one way only, $\star$. Presheaves generalise this by mapping to arbitrary sets (or

[1] The second author learnt this from a talk by Sam Staton.

even just finite ones, as we do), and we think of $F(p)$ as the set of possible ways for F to accept p. Sticking to strategies as sets of plays would lead to models of coarser equivalences, as obtained by Ghica-Murawski [17] and Laird [29]. (Harmer and McCusker [21], on the other hand, consider a finer equivalence for a non-deterministic language rather than a concurrent one.)

So, our strategies may both be viewed as a concurrent variant of game semantical innocent strategies, and as an innocent variant of presheaves.

Playgrounds Finally, our category of strategies is not constructed by hand. It is derived using the previously defined theory of *playgrounds* [24]. This theory draws direct inspiration from *Kleene coalgebra* [2].

In Kleene coalgebra, the main idea is that both the syntax and the semantics of various kinds of automata should be *derived* from more basic data describing, roughly, the 'rule of the game'. Formally, starting from a well-behaved (*polynomial*) endofunctor on sets, one constructs both (1) an equational theory and (2) a sound and complete coalgebraic semantics. This framework has been applied in traditional automata theory, as well as in quantitative settings. Nevertheless, its applicability to programming language theory is yet to be established. E.g., the derived languages do not feature parallel composition.

Playgrounds may be seen as a first attempt to convey such ideas to the area of programming language theory.

In [24], it was shown how to construct, from any playground $\mathbb{D}$, (1) a syntax and a transition system $\mathcal{S}_{\mathbb{D}}$, together with (2) a denotational model in terms of innocent, concurrent strategies as described above. This mimicks the syntactic and semantic models derived in Kleene coalgebra, except that we replace the equational theory by a transition system, and the coalgebraic semantics by a game semantics. Then, a playground $\mathbb{D}^{CCS}$ was constructed and shown, by embedding CCS into $\mathcal{S}_{\mathbb{D}^{CCS}}$, to give rise to a fully-abstract model for fair testing equivalence.

In this paper, we construct a playground $\mathbb{D}$ for π and show, using similar techniques, that it yields a fully-abstract model for our variant of fair testing equivalence (Theorems 4 and 5).

Related work Building upon previous work [1, 35] on *asynchronous* games, recent work by Winskel and collaborators [44, 47] attempts to define a notion of concurrent game encompassing both innocent game semantics and presheaf models. Ongoing work shows that the model does contain innocent game semantics, but presheaf models are yet to be investigated.

Furthermore, our model is inspired by Girard's *ludics* [18], Melliès's game semantics in string diagrams [34], Harmer et al.'s categorical combinatorics of innocence [20], and combinatorial structures from algebraic topology [30].

Finally, Hildebrandt's approach to fair testing equivalence [23] uses related techniques, namely sheaves. We also use sheaves: our innocence condition may be viewed as a sheaf condition, as briefly reviewed in Sect. 2.6. In Hildebrandt's work, sheaves are used to correctly handle infinite behaviour, whereas here they are used to force reactions of players to depend only on their view.

Plan In Sect. 2, we sketch the construction of our playground for π, recalling the notion along the way. We emphasise one particular axiom for playgrounds, which was most challenging when passing from CCS to π. We then recall the notion of strategy. In Sect. 3, we recall and instantiate the transition system for strategies constructed in [24] and define our variant of fair testing equivalence. Finally, we state our main results.

Perspectives We plan to adapt our semantics to other calculi like the Join and Ambients calculi, and ultimately get back to functional calculi. We hope to eventually generalise it, e.g., to some SOS format. More speculative directions include (1) defining a notion of morphism for playgrounds which would induce translations between strategies, and find sufficient conditions for such morphisms to preserve, resp. reflect behavioural equivalences; (2) applying playgrounds beyond programming language semantics; in particular, preliminary work shows that playgrounds easily account for cellular automata, which provides a testbed for morphisms of playgrounds [9].

Notation Throughout the paper, any finite ordinal n is seen as $\{1,\dots,n\}$ (rather than $\{0,\dots,n-1\}$). Set is the category of sets; set is the category of finite ordinals and arbitrary maps; ford is the category of finite ordinals and monotone maps. For any category $\mathbb{C}$, put $\widehat{\mathbb{C}} = [\mathbb{C}^{op}, \mathsf{Set}]$, $\overline{\mathbb{C}} = [\mathbb{C}^{op}, \mathsf{set}]$, $\widehat{\mathbb{C}} = [\mathbb{C}^{op}, \mathsf{ford}]$, and let $\widehat{\mathbb{C}}^f$ denote the category of *finite* presheaves, i.e., those presheaves F such that $\sum_{c\in \mathrm{ob}(\mathbb{C})} F(c)$ is finite. For all presheaves F of any such kind, $x \in F(d)$, and $f\colon c \to d$, let $x \cdot f$ denote $F(f)(x)$.

Our π-calculus processes will be infinite terms generated by the typing rules:

$$\frac{\dots \qquad \Gamma \cdot \alpha_i \vdash P_i \qquad \dots (\forall i \in n)}{\Gamma \vdash \sum_{i\in n} \alpha_i.P_i} \qquad \frac{\Gamma \vdash P \qquad \Gamma \vdash Q}{\Gamma \vdash P \mid Q} \qquad \frac{\Gamma + 1 \vdash P}{\Gamma \vdash \nu.P}\ .$$

Γ ranges over finite ordinals viewed as sets of variables $\{1,\dots,\Gamma\}$, and $\Gamma \cdot \alpha$ assumes that $\alpha ::= \bar{a}\langle b\rangle \mid a \mid \heartsuit$, for $a, b \in \Gamma$. In that case, $\Gamma\cdot\bar{a}\langle b\rangle = \Gamma\cdot\heartsuit = \Gamma$ and $\Gamma\cdot a = \Gamma+1$. The $\heartsuit$ form is a 'tick' action used to define fair testing equivalence. This is a de Bruijn-like presentation, which we equip with any standard reduction semantics [36], viewed as a reflexive graph Pi. For any $\Gamma \vdash P$ and map $h\colon \Gamma \to \Delta$ (of finite sets), we denote by $\Delta \vdash P[h]$ the result of renaming channels according to h in P.

Finally, and this will only be used in sketching our proof of Theorem 4, we consider a slightly more general notion of LTS than usual. We work in the category Gph of reflexive (directed, multi) graphs, and our category of LTSs over A is the slice category Gph/A. The usual notion of an LTS over an alphabet Σ is recovered by taking for A the free one-vertex reflexive graph with edges in Σ (and by restricting to faithful morphisms over A).

2 Diagrams and plays

In this section, we sketch the construction of our playground for π, and recall the notion of strategies. As sketched in the introduction, our playground will model a multi-player game, consisting of positions and plays between them. Positions are certain graph-like objects, where vertices represent players and channels. But what might be surprising is that moves are not just a binary relation between positions, because we not only want to say *when* there is a move from one position to another, but also *how* one moves from one to the other. This will be implemented by viewing moves from X to Y as *cospans* $X \to M \leftarrow Y$ in a certain category $\widehat{\mathbb{C}}^f$ of higher-dimensional graph-like objects, where X and Y respectively are the initial and final positions, and M describes how one goes from X to Y. By composing such moves (by pushout), we get a bicategory $\mathbb{D}_v$ of positions and plays. We then go on and equip this bicategory with more structure, namely that of a pseudo double category, where one direction models dynamics, and the other models space, e.g., the inclusion of a position into another. We then explain why the so-called 'fibration' axiom is non-obvious and describe our solution.

2.1 Diagrams

In preparation for the definition of our base category $\mathbb{C}$, recall that (directed, multi) graphs may be seen as presheaves over the category freely generated by the graph with two objects $\star$ and $[1]$, and two edges $s, t \colon \star \to [1]$. Any presheaf G represents the graph with vertices in $G(\star)$ and edges in $G[1]$, the source and target of any $e \in G[1]$ being respectively $e \cdot s$ and $e \cdot t$. A way to visualise how such presheaves represent graphs is to compute their *categories of elements* [33]. Recall that the category of elements $\int G$ for a presheaf G over $\mathbb{C}$ has as objects pairs (c, x) with $c \in \mathbb{C}$ and $x \in G(c)$, and as morphisms $(c, x) \to (d, y)$ all morphisms $f \colon c \to d$ in $\mathbb{C}$ such that $y \cdot f = x$. This category admits a canonical functor π_G to $\mathbb{C}$, and G is the colimit of the composite $\int G \xrightarrow{\pi_G} \mathbb{C} \xrightarrow{\mathsf{y}} \widehat{\mathbb{C}}$ with the Yoneda embedding. E.g., the category of elements for $\mathsf{y}[1]$ is the poset $(\star, s) \xrightarrow{s} ([1], id_{[1]}) \xleftarrow{t} (\star, t)$, which could be pictured as , where dots represent vertices, the triangle represents the edge, and links materialise the graph of $G(s)$ and $G(t)$, the convention being that t goes from the apex of the triangle. We thus recover some graphical intuition.

Our string diagrams will also be defined as (finite) presheaves over some base category $\mathbb{C}$. Let us give the formal definition of $\mathbb{C}$ for reference. We advise to skip it on a first reading: we then attempt to provide some graphical intuition.

Definition 1. *Let $G_{\mathbb{C}}$ be the graph with, for all n, m, with $a, b \in n$ and $c, d \in m$:*

- *vertices $\star$, $[n]$, π^l_n, π^r_n, π_n, ν_n, $\heartsuit_n$, $\iota_{n,a}$, $o_{n,a,b}$, and $\tau_{n,a,m,c,d}$;*
- *edges $s_1, ..., s_n : \star \to [n]$;*
- *for all $v \in \{\pi^l_n, \pi^r_n, \heartsuit_n, o_{n,a,b}\}$, edges $s, t : [n] \to v$;*
- *edges $[n] \xrightarrow{t} \nu_n \xleftarrow{s} [n+1]$ and $[n] \xrightarrow{t} \iota_{n,a} \xleftarrow{s} [n+1]$;*

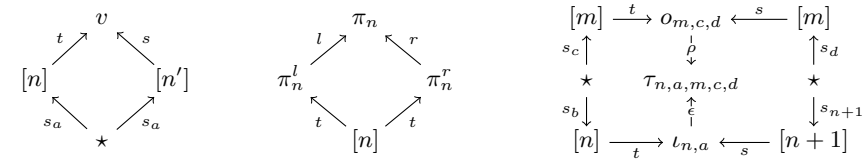
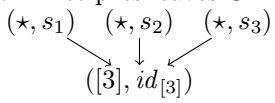


Fig. 1. Equations for $\mathbb{C}$

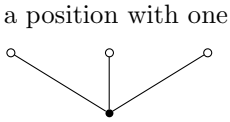
– *edges* $\pi_n^l \xrightarrow{l} \pi_n \xleftarrow{r} \pi_n^r$*;*
– *edges* $\iota_{n,a} \xrightarrow{\rho} \tau_{n,a,m,c,d} \xleftarrow{\epsilon} o_{m,c,d}$*.*

Let $\mathbb{C}$ *be the free category on* $G_{\mathbb{C}}$*, modulo the equations in Fig. 1, where, in the left-hand one,* n' *is* $n+1$ *when* $v = \nu_n$ *or* $\iota_{n,a}$*, and* n *otherwise.*

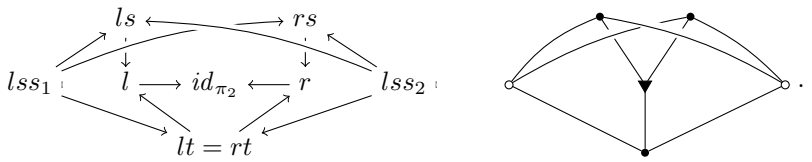
Our category of string diagrams will be the category of finite presheaves $\widehat{\mathbb{C}^f}$.



To explain this seemingly arbitrary definition, let us compute a few categories of elements. Let us start with an easy one, that of $[3] \in \mathbb{C}$ (we implicitly identify any $c \in \mathbb{C}$ with $\mathsf{y}c$). An easy computation shows that it is the poset pictured in the top part on the right. We will think of it as a position with one player $([3], id_{[3]})$ connected to three channels, and draw it as in the bottom part on the right, where the bullet represents the player, and circles represent channels. In particular, elements over [3] represent ternary players, while elements over $\star$ represent channels. The *positions* of our game are finite presheaves empty except perhaps on $\star$ and $[n]$'s. Other objects will represent moves. The graphical representation is slightly ambiguous, because the ordering of channels known to players is implicit. We will disambiguate in the text when necessary.



A more difficult category of elements is that of π_2. It is the poset generated by the graph on the left (omitting the base objects for conciseness):



We think of it as a binary player (lt) forking into two players (ls and rs), and draw it as on the right. The graphical convention is that a black triangle stands for the presence of id_{π_2}, l, and r. Below, we represent just l as a white triangle with only a left-hand branch, and symmetrically for r. Furthermore, in all our pictures, time flows 'upwards'.

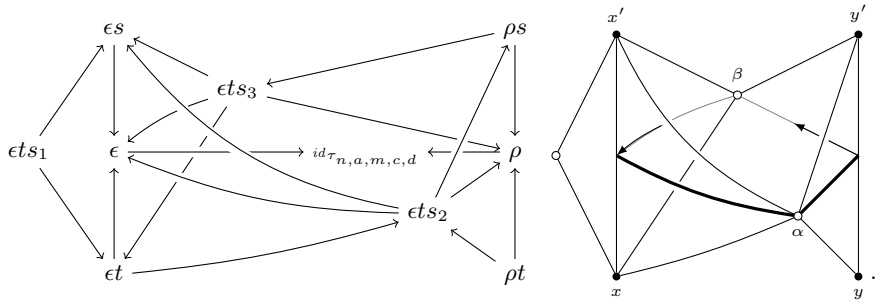
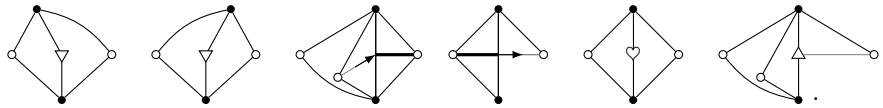


Fig. 2. Category of elements for $\tau_{1,1,3,2,3}$ and graphical representation

Another category of elements, characteristic of the π-calculus, is the one for synchronisation $\tau_{n,a,m,c,d}$. The case $(n, a, m, c, d) = (1, 1, 3, 2, 3)$ is the poset generated by the graph on the left of Fig. 2, which we will draw as on the right. The left-hand ternary player x outputs its 3rd channel, here β, on its 2nd channel, here α. The right-hand unary player y receives the sent channel on its 1st channel, here α. The carrier channel is marked with thick lines, while the transmitted channel is indicated with arrows. Both players have two occurrences, one before and one after the move, respectively marked as x/x' and y/y'. Both x and x' have arity 3 here, while y has arity 1 and y', having gained knowledge of channel β, has arity 2.

We leave the computation of other categories of elements as an exercise to the reader. The remaining diagrams for π^l_p, π^r_p, $o_{m,c,d}$, $\iota_{n,a}$, $\heartsuit_p$, and ν_p are depicted below, for $p = 2$ and $(m, c, d, n, a) = (3, 2, 3, 1, 1)$:



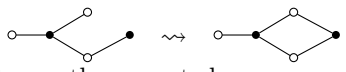
The first two are *views*, in the game semantical sense, of the fork move π_2 explained above. The next two, $o_{m,c,d}$ (for 'output') and $\iota_{n,a}$ (for 'input'), respectively represent what the sender and receiver can see of the above synchronisation move. The next diagram is a 'tick' move, used for defining fair testing equivalence. The last one is a channel creation move.

2.2 From diagrams to moves

In the previous section, we have defined our category of diagrams as $\widehat{\mathbb{C}^f}$, and provided some graphical intuition on its objects. The next goal is to construct a bicategory whose objects are positions (recall: presheaves empty except perhaps on $\star$ and $[n]$'s), and whose morphisms represent plays in our game. We start in this section by defining moves, and continue in the next one by explaining how to compose moves to form plays. Moves are defined in two stages: *seeds*, first,

give the local form for moves; moves are then defined by embedding seeds into bigger positions.

To start with, until now, our diagrams contain no information about the 'flow of time'. To add this information, for each diagram M representing a move, we define its initial and final positions, say X and Y, and view the whole move as a cospan $Y \xrightarrow{s} M \xleftarrow{t} X$. We have taken care, in drawing our diagrams before, of placing initial positions at the bottom, and final positions at the top. So, e.g., the initial position X and final position Y for the synchronisation move are pictured on the right and they map into (the representable presheaf over) $\tau_{1,1,3,2,3}$ in the obvious ways, yielding the cospan $Y \xrightarrow{s} M \xleftarrow{t} X$. We leave it to the reader to define, based on the above pictures, the expected cospans



$$\begin{array}{cccccccccc}
[n] \mid [n] & & [m]_{\,c,d}|_{a,n+1}\,[n+1] & & [n] & [n] & [m] & [n+1] & [n] & [n+1] \\
\downarrow & & \downarrow & & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
\pi_n & & \tau_{n,a,m,c,d} & & \pi^l_n & \pi^r_n & o_{m,c,d} & \iota_{n,a} & \heartsuit_n & \nu_n \\
\uparrow & & \uparrow & & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\
[n] & & [m]_{\,c}|_{a}\,[n] & & [n] & [n] & [m] & [n] & [n] & [n],
\end{array}$$

where initial positions are on the bottom row, and we denote by $[m]_{a_1,\ldots,a_p}|_{c_1,\ldots,c_p}$ $[n]$ the position consisting of an m-ary player x and an n-ary player y, quotiented by the equations $x \cdot s_{a_k} = y \cdot s_{c_k}$ for all $k \in p$. When both lists are empty, by convention, $m = n$ and the players share all channels in order.

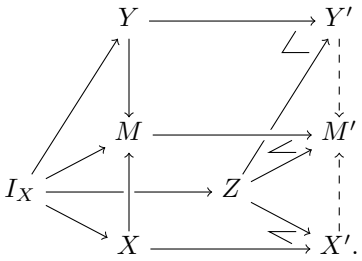
Definition 2. *These cospans are called* seeds. *Their lower legs are called* t-legs.

As announced, the moves of our game are obtained by embedding seeds into bigger positions. This means, e.g., allowing a fork move to occur in a position with more than one player. We proceed as follows.

Definition 3. *Let the* interface *of a seed* $Y \xrightarrow{s} M \xleftarrow{t} X$ *be* $I_X = X(\star) \cdot \star$, *i.e., the position consisting only of the channels of the initial position of the seed.*

Since channels present in the initial position remain in the final one, we have for each seed a commuting diagram as on the right. By gluing any position Z to the seed along its interface, we obtain a new cospan, say $Y' \to M' \leftarrow X'$. I.e., for any morphism $I_X \to Z$, we push $I_X \to X$, $I_X \to M$, and $I_X \to Y$ along $I_X \to Z$ and use the universal property of pushout, as in:

$$\begin{array}{ccccc}
 & & I_X & & \\
 & \swarrow & \downarrow & \searrow & \\
Y & \longrightarrow & M & \longleftarrow & X
\end{array}$$

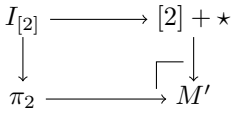


$$\begin{array}{ccccc}
Y & & \longrightarrow & & Y' \\
 & \nwarrow & \downarrow & \nearrow & \downarrow \\
 & & M & \longrightarrow & M' \\
 & \nearrow & \uparrow & \nearrow & \uparrow \\
I_X & \longrightarrow & Z & \searrow & \\
 & \searrow & \uparrow & & \uparrow \\
 & & X & \longrightarrow & X'.
\end{array}$$

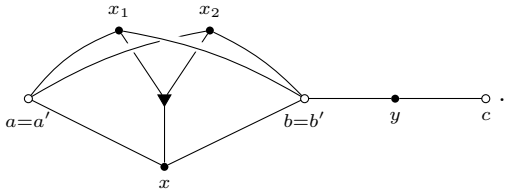
Definition 4. *Let* (global) moves *be all cospans obtained in this way.*

Recall that colimits in presheaf categories are pointwise. So, e.g., taking pushouts along injective maps graphically corresponds to gluing diagrams together. Let us do a few examples.

Example 1. The copsan $[2] \mid [2] \xrightarrow{[ls,rs]} \pi_2 \xleftarrow{lt} [2]$ has as canonical interface the presheaf $I_{[2]} = 2 \cdot \star$, consisting of two channels, say a and b. Consider the position $[2] + \star$ consisting of a player y with two channels b' and c, plus an additional channel a'. Further consider the map $h \colon I_{[2]} \to [2] + \star$ defined by $a \mapsto a'$ and $b \mapsto b'$. The pushout



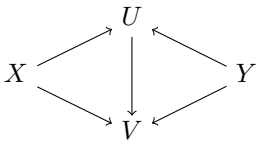
is



Example 2. The canonical interface, being the interface of the initial position, may not contain all channels of the move. In particular, for an input move which is not part of any synchronisation, the received channel cannot be part of the initial position.

2.3 From moves to plays

Having defined moves, we now define their composition to define our bicategory $\mathbb{D}_v$ of positions and plays. $\mathbb{D}_v$ will be a sub-bicategory of $\mathsf{Cospan}(\widehat{\mathbb{C}^f})$, the bicategory which has as objects all finite presheaves on $\mathbb{C}$, as morphisms $X \to Y$ all cospans $X \to U \leftarrow Y$, and as 2-cells $U \to V$ all commuting diagrams as on the right. Composition is given by pushout, and hence is not strictly associative.



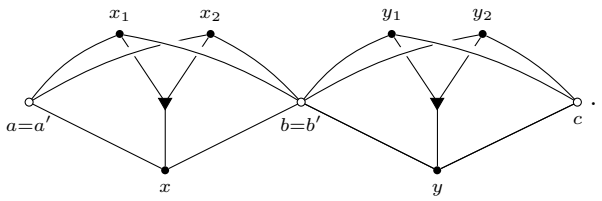
Remark 1. We choose to view the initial position as the *target* of the morphism in $\mathsf{Cospan}(\widehat{\mathbb{C}^f})$, in order to emphasise below that the fibration axiom is very close to a universal property of pullback [27].

Definition 5. Plays *are composites of moves in* $\mathsf{Cospan}(\widehat{\mathbb{C}^f})$. *Let* $\mathbb{D}_v$ *be the sub-bicategory consisting of positions and plays.*

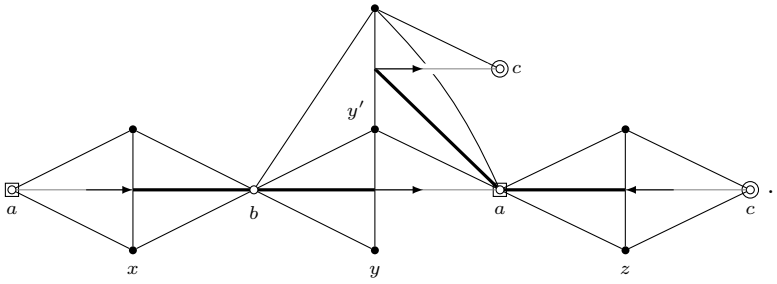
Remark 2. We do not yet specify what the 2-cells of $\mathbb{D}_v$ are. This will follow from the next section.

Intuitively, composition by pushout glues diagrams on top of each other, which features some concurrency.

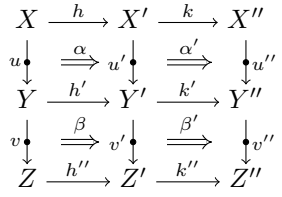
Example 3. Composing the move of Example 1 with a forking move by y yields



Example 4. Composition retains causal dependencies between moves. To see this, consider the following diagram. In the initial position, there are channels a, b, and c, and three players $x(a, b), y(b)$, and $z(a, c)$ (we indicate the channels known to each player in parentheses). In a first move, x sends a on b, which is received by y. In a second move, z sends c on a, which is received by (the avatar y' of) y. The second move is enabled by the first one, by which y gains knowledge of a. The corresponding diagram looks like the following, identifying the two framed nodes and the two circled ones:



2.4 A pseudo double category



We now continue the construction of our playground for π by adding a new dimension. Namely, we view $\mathbb{D}_v$ as the vertical part of a *(pseudo) double category* [19, 16]. This is a weakening of Ehresmann's double categories [10], where one direction has non-strictly associative composition. A pseudo double category $\mathbb{D}$ consists of a set $\mathrm{ob}(\mathbb{D})$ of *objects*, shared by a 'horizontal' category $\mathbb{D}_h$ and a 'vertical' bicategory $\mathbb{D}_v$. Following Paré [41], $\mathbb{D}_h$, being a mere category, has standard notation (normal arrows and $\circ$ for composition), while the bicategory $\mathbb{D}_v$ earns fancier notation ($\multimap$ arrows and $\bullet$ for composition). $\mathbb{D}$ is furthermore equipped with a set of *double cells* α, which have vertical, resp. horizontal, domain and codomain, denoted by $\mathrm{dom}_v(\alpha)$, $\mathrm{cod}_v(\alpha)$, $\mathrm{dom}_h(\alpha)$, and $\mathrm{cod}_h(\alpha)$. We picture this as, e.g., α above, where $u = \mathrm{dom}_h(\alpha)$, $u' = \mathrm{cod}_h(\alpha)$, $h = \mathrm{dom}_v(\alpha)$, and $h' = \mathrm{cod}_v(\alpha)$. Finally, there are operations for composing double cells: *horizontal* composition $\circ$ composes them along a common vertical morphism, *vertical* composition $\bullet$ composes along horizontal morphisms. Both vertical compositions (of morphisms and of double cells) may only be associative up to coherent isomorphism. The full axiomatisation is given by Garner [16],

and we here only mention the *interchange law*, which says that the two ways of parsing the above diagram coincide: $(\beta' \circ \beta) \bullet (\alpha' \circ \alpha) = (\beta' \bullet \alpha') \circ (\beta \bullet \alpha)$.

Returning to our playground for π, we put

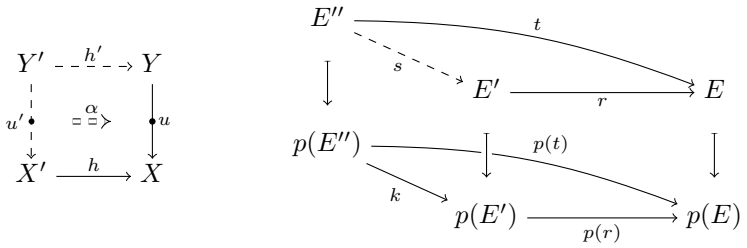
Definition 6. *Let* $\mathsf{H} \subseteq \widehat{\mathbb{C}}^f$ *be the identity-on-objects subcategory of natural transformations with injective components, except perhaps on channels.*

For $\mathbb{D}_h$, we take the full subcategory of H spanning positions[2]. Finally, as double cells, $\mathbb{D}$ has commuting diagrams as on the right, where the vertical cospans are plays and h, k, and l are in H.

$$\begin{array}{ccc} X & \xrightarrow[l]{} & X' \\ {\scriptstyle s}\downarrow & & \downarrow{\scriptstyle s'} \\ U & \xrightarrow{k} & V \\ {\scriptstyle t}\uparrow & & \uparrow{\scriptstyle t'} \\ Y & \xrightarrow[h]{} & Y' \end{array} \qquad (1)$$

Proposition 1. $\mathbb{D}$ *forms a pseudo double category.*

There is more data to provide and axioms to check to obtain that $\mathbb{D}$ forms a playground. The most serious challenge is to show that the vertical codomain functor $\mathrm{cod}_v \colon \mathbb{D}_H \to \mathbb{D}_h$ is a (Grothendieck) fibration [27]. Here, $\mathbb{D}_H$ denotes the category with vertical morphisms as objects, and double cells as morphisms. The functor cod_v maps any play to its initial position and any double cell to its lower border. Intuitively, the fibration axiom amounts to the existence, for all plays $Y \overset{u}{\bullet\!\!\to} X$ and horizontal morphisms $X' \xrightarrow{h} X$, of a universal ($\approx$ maximal) way of restricting u to X', as on the left below:



Formally, consider any functor $p \colon \mathbb{E} \to \mathbb{B}$. A morphism $r \colon E' \to E$ in $\mathbb{E}$ is *cartesian* when, as on the right above, for all $t \colon E'' \to E$ and $k \colon p(E'') \to p(E')$, if $p(r) \circ k = p(t)$ then there exists a unique $s \colon E'' \to E'$ such that $p(s) = k$ and $r \circ s = t$.

Definition 7. *A functor* $p \colon \mathbb{E} \to \mathbb{B}$ *is a* fibration *iff for all* $E \in \mathbb{E}$*, any* $h \colon B' \to p(E)$ *has a cartesian lifting, i.e., a cartesian antecedent by* p*.*

Unlike in the CCS case, what the lifting u' should be in our case is generally not obvious (see Ex. 6 and 7 below).

2.5 Factorisations and fibrations

Our approach to ensuring that cod_v is a fibration works for π as well as for CCS, and is much clearer conceptually than our first proposal [24].

[2] Injective transformations suffice for CCS, but not for π, because of channel mobility.

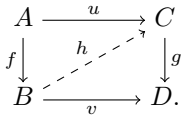
Definition 8. *A (strong)* factorisation system *[15, 28] on a category* $\mathbb{C}$ *consists of two subcategories* L *and* R *of* $\mathbb{C}$*, both containing all isomorphisms, such that any morphism in* $\mathbb{C}$ *factors essentially uniquely as* $r \circ l$ *with* $l \in \mathsf{L}$ *and* $r \in \mathsf{R}$*.*

'Essentially unique' here means unique up to unique commuting isomorphism.

Example 5. In Set, surjective and injective maps form a factorisation system.

Our aim is to construct such a factorisation system (L, R) on $\widehat{\mathbb{C}^{\mathcal{I}}}$, such that L contains all t-legs of plays, and R contains all morphisms in H. The idea is to compute the restriction of V along h, as in (1), by factoring $t' \circ h$ as $k \circ t$ with $k \in \mathsf{R}$ and $t \in \mathsf{L}$, and then taking the pullback of k and s'.

Actually, it is enough to demand that L contains t-legs of *seeds*. Indeed, as is well-known, L is always stable under pushout; and, by construction, t-legs of plays are composites of pushouts of t-legs of seeds.



We rely on Bousfield's construction [3, 28] of factorisation systems from a generating class of maps in L (the *generating cofibrations*). For any morphism $f\colon A \to B$ and $g\colon C \to D$, let $f \perp g$ iff for all commuting squares as on the right, there is a unique lifting h making both triangles commute. This extends in the obvious way to classes of morphisms, which we denote by $\mathsf{L} \perp \mathsf{R}$. For all classes L and R of morphisms, let $\mathsf{L}^{\perp} = \{g \mid \mathsf{L} \perp \{g\}\}$ and ${}^{\perp}\mathsf{R} = \{f \mid \{f\} \perp \mathsf{R}\}$.

Theorem 1 (Bousfield). *For any class* T *of morphisms in any locally presentable category* $\mathbb{E}$*, the pair* $({}^{\perp}(\mathsf{T}^{\perp}), \mathsf{T}^{\perp})$ *forms a factorisation system.*

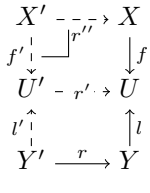
In the setting of the theorem, one may construct the double category $\mathbb{D}_{\mathsf{L},\mathsf{R}}$, with $\mathsf{L} = {}^{\perp}(\mathsf{T}^{\perp})$ and $\mathsf{R} = \mathsf{T}^{\perp}$, with the same objects as $\mathbb{E}$, and such that

- vertical morphisms $X \to Y$ are cospans $X \xrightarrow{f} U \xleftarrow{l} Y$ with $l \in \mathsf{L}$,
- horizontal morphisms are morphisms in R, and
- double cells are diagrams like (1) with $r' \in \mathsf{R}$.

The theorem yields:

Proposition 2. *The functor* $\mathrm{cod}_v\colon (\mathbb{D}_{\mathsf{L},\mathsf{R}})_H \to (\mathbb{D}_{\mathsf{L},\mathsf{R}})_h$ *is a fibration.*

Proof. Consider any vertical morphism $X \xrightarrow{f} U \xleftarrow{l} Y$ and $r\colon Y' \to Y$ in R. We construct the restriction of (f, l) along r by factoring $l \circ r$ as $r' \circ l'$, and then taking the pullback, as on the right. It is well-known that in any factorisation system, R is stable under pullback, hence $r'' \in \mathsf{R}$. The universal property of this restriction follows from the other well-known general fact that for all $l \in \mathsf{L}$ and $r \in \mathsf{R}$, we have $l \perp r$. Indeed, consider as in Fig. 3 any other vertical morphism $X'' \xrightarrow{f''} U'' \xleftarrow{l''} Y''$ and morphism (t, t', t'') to U, together with a morphism $s\colon Y'' \to Y'$ such that $r \circ s = t$. The lifting property $l'' \perp r'$ and the universal property of pullback give the unique s' and s'' making the diagram of Fig. 3 commute. Finally, s' and s'' are in R, by the general fact that R has the *left cancellation* property [28]: for all composable u and v, $vu \in \mathsf{R}$ and $v \in \mathsf{R}$ implies $u \in \mathsf{R}$. ◻



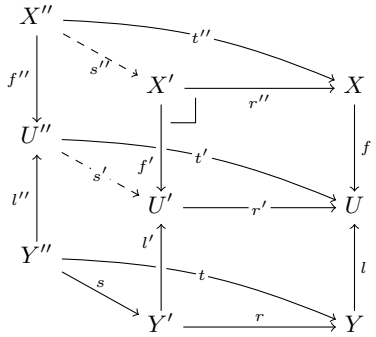


Fig. 3. Cartesianness of lifting

Corollary 1. *The functor* $\mathrm{cod}_v \colon \mathbb{D}_H \to \mathbb{D}_h$ *is a fibration.*

Proof. We apply the theorem for T the set of all t-legs of seeds to obtain a factorisation system (L, R) on $\widehat{\mathbb{C}}$. We temporarily work in $\widehat{\mathbb{C}}$, and then show that the needed factorisations remain in $\widehat{\mathbb{C}}^f$. Finally, we verify that H is contained in R, is stable under pullback, and has the left cancellation property. □

Let us briefly explain what factorisations do. First, for all morphisms $r \colon U \to V$ in $\widehat{\mathbb{C}}$, $r \in \mathsf{R}$ iff $t \perp r$ for all t-legs t of seeds. But for any seed $X \xrightarrow{s} M \xleftarrow{t} Y$, M is a representable presheaf. By Yoneda, $t \perp r$ intuitively means that for all $x \in V(M)$, if Y is already present in U, then the whole of M is. Otherwise said, morphisms in r may not 'grow' new moves from initial positions. Consequently, in (1), the first factor t of any $t' \circ h \colon Y \to V$ will perform all possible moves from Y; and the second factor will then map the result to V.

Example 6. Consider the synchronisation of Fig. 2. Let V be the synchronisation obtained by identifying α and β, so that the received name is already known to y. Consider the restriction of V to just y. By Ex. 2, an input move (in the absence of a corresponding output move) cannot receive an already known channel. Thus, the restriction of V to y is just the input seed.

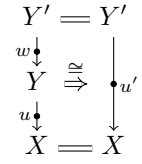
Example 7. Consider again the synchronisation of Fig. 2, say $X \xrightarrow{f} U \xleftarrow{l} Y$, and the position $[3] + [1]$ consisting of a ternary player x_0 and a unary player y_0, not sharing any channel. Consider the horizontal map $r \colon [3] + [1] \to Y$ defined by $x_0 \mapsto x$ and $y_0 \mapsto y$. The factorisation of $l \circ r$ as $r' \circ l'$ yields a play where x_0 does an $o_{3,2,3}$ move and y_0 does an $\iota_{1,1}$ move (the order is irrelevant). Thus, restrictions of moves may be plays of length strictly greater than one.

We at last obtain:

Theorem 2. $\mathbb{D}$ *forms a playground.*

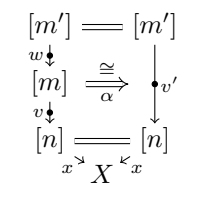
2.6 Innocent Strategies

Following our previous work [24], we now associate to each object, i.e., position X, its category of *strategies* S_X. Consider first the most naive notion of strategy over X. In a playground, the analogue of the poset of plays with prefix order is given by the category $\mathbb{P}(X)$ with plays $u \colon Y \nrightarrow X$ as objects and double cells as on the right as morphisms $u \to u'$. Let the category B_X of *behaviours* on X be $\overline{\mathbb{P}(X)}$. Behaviours do not yield a satisfactory notion of strategy:



Example 8. Consider the position X consisting of three players x, y, z sharing a channel a. Let $u_{x,y}$ denote the play where x sends a on a, which is received by y; let similarly $u_{x,z}$ denote the play where x sends a on a, which is received by z. Let, finally, i_z denote the play where z inputs on a. One may define a behaviour B mapping $u_{x,y}$ and i_z to a singleton, and $u_{x,z}$ to the empty set. Because $u_{x,y}$ is accepted, x accepts to send a on a. Because i_z is accepted, z accepts to receive on a. The problem is that B rejecting $u_{x,z}$ amounts to x or z choosing their partners for synchronising, e.g., x accepts to send a on a *only to players other than* z.

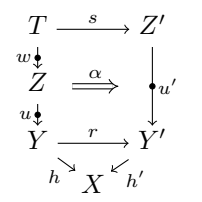
We want to rule out this kind of behaviour from our model, and our solution is innocence. Let *basic* seeds be all seeds of the shape $\iota_{n,a}$, $o_{n,a,b}$, ν_n, $\heartsuit_n$, π^l_n, or π^r_n, for $a, b \in n$. Intuitively, basic seeds follow exactly one player. Let *views* be composites of basic seeds in $\mathbb{D}_v$. We now replace our base $\mathbb{P}(X)$ with $\mathbb{V}_X$, whose objects are pairs (v, x) of a view $v \colon [m] \nrightarrow [n]$ and a horizontal morphism $x \colon [n] \to X$, i.e., by Yoneda, of a player of X and a view from it. Morphisms $v \to v'$ are cells α as on the right.



Definition 9. *Let the category* S_X *of* strategies *over* X *be* $\widehat{\mathbb{V}_X}$.

Remark 3. We restrict to presheaves of finite ordinals (as opposed to finite sets). There is an essentially surjective embedding $\widehat{\mathbb{V}_X} \hookrightarrow \overline{\mathbb{V}_X}$, so we do not really lose any strategy in the process, only some completeness properties. On the other hand, we gain the syntactic characterisation used in the next section.

To relate strategies and behaviours, consider the category $\mathbb{P}_X$ with as objects pairs (u, h) of a play $u \colon Z \nrightarrow Y$ and a horizontal morphism $h \colon Y \to X$, and as morphisms $(u, h) \to (u', h')$ all diagrams as on the right. This category contains both $\mathbb{V}_X$ and $\mathbb{P}(X)$ in obvious ways, and it furthermore allows to describe the views (v, x) of a general play u' by taking $h' = id_X$[3]. So our morphisms account both for prefix inclusion, and for 'spatial' inclusion, i.e., inclusion of a play into a play on a bigger position.



Right Kan extension and restriction along $\mathbb{V}_X^{op} \hookrightarrow \mathbb{P}_X^{op} \hookleftarrow \mathbb{P}(X)^{op}$ induce a functor $\mathsf{S}_X \to \mathsf{B}_X$. Intuitively, this functor maps any strategy S to the behaviour

[3] There is a small problem, however: morphisms should only describe how u maps to u', not w. We actually consider a quotient of morphisms to rectify this.

accepting a play u iff S accepts all views of u. This also allows to view strategies as sheaves [33] for a certain Grothendieck topology on $\mathbb{P}_X$, as explained in previous papers [25]. Intuitively, strategies provide (locally determined) behaviours for all subpositions of X. Such local 'behaviour' may become irrelevant when passing to the globally-defined behaviours. In particular, $\mathsf{S}_X \to \mathsf{B}_X$ is neither injective on objects, nor full, nor faithful.

Example 9. If two strategies differ, but are both empty on the views of some player, then both are mapped to the empty behaviour.

3 Bridging the gap with π

3.1 Syntax and transition system for strategies

One of the main results about playgrounds [24] entails that strategies over $\mathbb{D}$ are entirely described by the following typing rules

$$\frac{\ldots\ n_B \vdash S_B\ \ldots\ (\forall B\colon [n_B] \to [n])}{n \vdash_{\mathsf{D}} \langle (S_B)_{B \in \mathbb{B}_n} \rangle} \qquad \frac{\ldots\ n \vdash_{\mathsf{D}} D_i\ \ldots\ (\forall i \in m)}{n \vdash \oplus_{i \in m} D_i}\ (m \in \mathbb{N}),$$

where $\mathbb{B}_n$ is the set of basic seeds from $[n]$ as defined above. The rules feature two kinds of judgements, $\vdash$ for plain strategies, and $\vdash_{\mathsf{D}}$ for *definite* strategies, intuitively those with exactly one initial state.

Remark 4. The sum $\oplus$ is not commutative (although it is up to fair testing equivalence).

Theorem 3 ([24]). *Strategies over $[n]$ are in bijection with possibly infinite terms in context n.*

Furthermore, giving a strategy over any position X amounts to giving a strategy over $[n]$ for each n-ary player of X.

Theorem 3 yields the following coinductive interpretation of processes:

$$[\![\Gamma \vdash \textstyle\sum_i \alpha_i.P_i]\!] = \langle B \mapsto \oplus_{\{i \mid [\![\alpha_i]\!] = B\}} [\![\Gamma \cdot \alpha_i \vdash P_i]\!] \rangle$$

$$[\![\Gamma \vdash P \mid Q]\!] = \left\langle \begin{array}{c} \pi^l_\Gamma \mapsto [\![\Gamma \vdash P]\!] \\ \pi^r_\Gamma \mapsto [\![\Gamma \vdash Q]\!] \\ - \mapsto \emptyset \end{array} \right\rangle \qquad [\![\Gamma \vdash \nu.P]\!] = \left\langle \begin{array}{c} \nu_\Gamma \mapsto [\![\Gamma + 1 \vdash P]\!] \\ - \mapsto \emptyset \end{array} \right\rangle,$$

with $[\![\bar{a}\langle b \rangle]\!] = o_{\Gamma,a,b}$, $[\![a]\!] = \iota_{\Gamma,a}$, $[\![\heartsuit]\!] = \heartsuit_\Gamma$, $\emptyset$ is the empty $\oplus$ sum, and $- \mapsto \emptyset$ means that all unmentioned basic seeds are mapped to $\emptyset$.

Example 10. Omitting typing contexts, we have

$$[\![\Gamma \vdash a.P + a.Q + \bar{b}\langle c \rangle.R]\!] = \left\langle \begin{array}{c} \iota_{\Gamma,a} \mapsto [\![\Gamma + 1 \vdash P]\!] \oplus [\![\Gamma + 1 \vdash Q]\!] \\ o_{\Gamma,b,c} \mapsto [\![\Gamma \vdash R]\!] \\ - \mapsto \emptyset \end{array} \right\rangle.$$

We now define a transition system for definite strategies, which is useful for characterising fair testing equivalence, and for which we need to define two auxiliary operations. The first is an operation of *derivation* along a basic seed, defined from definite strategies to strategies by $\partial_B\langle(S_{B'})_{B'\in\mathbb{B}_n}\rangle = S_B$. The second is a partial *restriction* operation from strategies to definite strategies, defined if $i \in p$ by $(\oplus_{i'\in p} D_{i'})_{|i} = D_i$.

Example 11. Following up on Example 10 and omitting contexts, we have

$$(\partial_{[\![a]\!]}[\![a.P + a.Q + \bar{b}\langle c\rangle.R]\!])_{|2} = [\![Q]\!] \qquad (\partial_{[\![\bar{b}\langle c\rangle]\!]}[\![a.P + a.Q + \bar{b}\langle c\rangle.R]\!])_{|1} = [\![R]\!].$$

These operations may be extended to arbitrary strategies and moves, in a way which we will gloss over here. We may thus write $(\partial_M S)_{|i}$. This yields:

Definition 10. *Let $\mathcal{S}_{\mathbb{D}}$ denote the free reflexive graph with as vertices pairs of a position X and a definite strategy D over X, and as edges all well-defined triples $(X, D) \xrightarrow{M} (Y, (\partial_M D)_{|i})$, for all moves $M : Y \multimap X$.*

We view this graph as a transition system for strategies.

Example 12. We have examples mirroring transitions in π. Calling D the translation of the process of Example 10, we have, e.g., $([\Gamma], D) \xrightarrow{\iota_{\Gamma,a}} ([\Gamma + 1], [\![P]\!])$, and the same with Q. But we also have transitions for things which usually go into structural equivalence, e.g., $([\Gamma], [\![P \mid Q]\!]) \xrightarrow{\pi_\Gamma} ([\Gamma] \mid [\Gamma], ([\![P]\!], [\![Q]\!]))$. In the final state, by the second part of Theorem 3, we define a strategy on $[\Gamma] \mid [\Gamma]$ by providing two strategies on $[\Gamma]$. Similarly, we have a transition $([\Gamma], [\![\nu.P]\!]) \xrightarrow{\nu_\Gamma} ([\Gamma + 1], [\![P]\!])$.

3.2 Fair testing equivalence from the transition system

The point of the transition system $\mathcal{S}_{\mathbb{D}}$ is to characterise our semantic analogue of fair testing equivalence. For lack of space, we describe the characterisation, omitting the direct, game semantical definition. First, as announced in the introduction, we allow tests to rename some channels. Recall from Definition 3 the canonical interface I_X of a position X.

Definition 11. *For any state (X, D) of $\mathcal{S}_{\mathbb{D}}$, a* test *for (X, D) is a pair of a horizontal morphism $h : I_X \to Y$ and a strategy T on Y.*

The morphism $I_X \to Y$ may identify some channels and introduce new ones. Whether such a test is passed successfully will be determined by the 'closed-world' dynamics of the strategy (D, T) over the pushout $Z = X +_{I_X} Y$. Intuitively, the players of Z are partitioned into players from X and players from Y, so (D, T) is, by a slight abuse of language, a strategy for the whole.

Let now $\mathcal{S}^{\mathbb{W}}_{\mathbb{D}}$, the *closed-world* part of $\mathcal{S}_{\mathbb{D}}$, be the identity-on-vertices subgraph of $\mathcal{S}_{\mathbb{D}}$ consisting of edges whose underlying moves have the shape $\tau_{n,a,m,c,d}$, ν_n, $\heartsuit_n$, or π_n. There is an obvious morphism of reflexive graphs $\ell_{\mathbb{D}} : \mathcal{S}^{\mathbb{W}}_{\mathbb{D}} \to \Sigma$ to the one-vertex reflexive graph with one non-identity edge $\heartsuit$. We denote by $(X, D) \Rightarrow$

(X', D') the existence of a path in $\mathcal{S}_{\mathbb{D}}^{\mathbb{W}}$ mapped by $\ell_{\mathbb{D}}$ to a path of identities in Σ, and by $(X, D) \overset{\heartsuit}{\Rightarrow} (X', D')$ the existence of a path mapped to a path consisting of identities and exactly one $\heartsuit$ edge.

Definition 12. *Let $\bot^{\mathbb{D}}$ denote the set of all vertices x of $\mathcal{S}_{\mathbb{D}}$ such that, for all $x \Rightarrow x'$, there exists x'' such that $x' \overset{\heartsuit}{\Rightarrow} x''$. Let $(X, D)^{\bot}$ denote the set of all tests (h, T) such that $(D, T) \in \bot^{\mathbb{D}}$. Finally, let $(X, D) \sim^{\mathbb{D}} (X', D')$ iff $(X, D)^{\bot} = (X', D')^{\bot}$.*

3.3 Main results

In this section, we at last state our main results. First, let us define our variant of fair testing equivalence for π. Let a *test* for $\Gamma \vdash P$ consist of a pair of a map $h \colon \Gamma \to \Delta$ and a process $\Delta \vdash R$. Let $Pi^{\mathbb{W}}$ denote the identity-on-vertices sub-reflexive graph of Pi consisting of τ and $\heartsuit$ transitions. There is an obvious morphism $\ell^{Pi} \colon Pi^{\mathbb{W}} \to \Sigma$ and, mimicking previous notation, we put:

Definition 13. *Let $\bot^{Pi}$ denote the set of all vertices x of Pi such that, for all $x \Rightarrow x'$, there exists x'' such that $x' \overset{\heartsuit}{\Rightarrow} x''$. Let $(\Gamma \vdash P)^{\bot}$ denote the set of all tests (h, R) such that $(P[h] \mid R) \in \bot^{Pi}$. Finally, let $(\Gamma \vdash P) \sim^{Pi} (\Gamma \vdash Q)$ iff $(\Gamma \vdash P)^{\bot} = (\Gamma \vdash Q)^{\bot}$.*

Theorem 4. *For all P, Q, $(\Gamma \vdash P) \sim^{Pi} (\Gamma \vdash Q)$ iff $([\Gamma], [\![P]\!]) \sim^{\mathbb{D}} ([\Gamma], [\![Q]\!])$.*

Proof sketch. The main difficulty is that we have to compare LTSs over very different alphabets. A first point is that edges in $\mathcal{S}_{\mathbb{D}}$ are very intensional. E.g., an input transition describes not only the involved channels but also which *player* makes the move. A second point is that $\mathcal{S}_{\mathbb{D}}$ is not 'modular', in the sense that it is not obvious to infer the transitions of a vertex (X, D) from the transitions of players of X. E.g., we have transitions $[\![\nu a.a(x)]\!] \xrightarrow{\nu_0} [\![a(x)]\!] \xrightarrow{\iota_{1,1}} 0$.

We rectify the latter deficiency first, by designing a finer LTS $\mathcal{S}_{\mathbb{D}}^{\mathcal{L}}$ for $\mathbb{D}$. Its vertices are triples (I, h, S) of an interface I, a horizontal map $h \colon I \to X$, and a strategy S over X. I represents all channels known to the environment, and the idea is that all transitions in $\mathcal{S}_{\mathbb{D}}^{\mathcal{L}}$ may be completed into closed-world transitions by interacting at I. This corrects the second mentioned deficiency, but $\mathcal{S}_{\mathbb{D}}^{\mathcal{L}}$ remains too intensional. So, we coarsen the LTS $\mathcal{S}_{\mathbb{D}}^{\mathcal{L}}$ to a new LTS $\mathcal{S}_{\mathbb{D}}^{\mathbb{A}}$. The new labels are given by the free reflexive graph $\mathbb{A}$ with as vertices all maps $\Delta \to \Gamma$ of finite sets, and as with edges as defined by the rules in Fig. 4[4].

The idea, for vertices, is that Δ represents the channels of the interface, and Γ represents the channels that the considered process or strategy (say, an agent) knows locally. The first rule should be easy. The second rule says that an agent may create a private channel, *a priori* unknown to the environment. The next two rules, for input and output, have been simplified for clarity. The important point is their symmetry: both add one channel to the interface. The input rule,

[4] In Fig. 4, we put side conditions as premises for conciseness.

$$\frac{}{(\Delta \xrightarrow{h} \Gamma) \xleftarrow{\heartsuit} (\Delta \xrightarrow{h} \Gamma)} \qquad \frac{}{(\Delta \xrightarrow{h} \Gamma) \xleftarrow{\nu} (\Delta \xrightarrow{h} \Gamma \xhookrightarrow{\subseteq} \Gamma + 1)}$$

$$\frac{a \in \mathrm{Im}(h)}{(\Delta \xrightarrow{h} \Gamma) \xleftarrow{\iota(a)} (\Delta + 1 \xrightarrow{h+!} \Gamma + 1)} \qquad \frac{a \in \mathrm{Im}(h)}{(\Delta \xrightarrow{h} \Gamma) \xleftarrow{o(a,b)} (\Delta + 1 \xrightarrow{[h,b]} \Gamma)}$$

$$\frac{a, c \in \mathrm{Im}(h); a \neq c}{(\Delta \xrightarrow{h} \Gamma) \xleftarrow{o(a,b) \rightarrowtriangle \iota(c)} (\Delta \xrightarrow{h} \Gamma \xhookrightarrow{\subseteq} \Gamma + 1)} \qquad \frac{}{(\Delta \xrightarrow{h} \Gamma) \xleftarrow{\delta} (\Delta \xrightarrow{h} \Gamma)} .$$

Fig. 4. Edges for $\mathbb{A}$

however, locally considers the new channel as fresh, whereas the output rule records that it is the sent channel. By the side condition, the channel on which the synchronisation occurs should belong to Δ. The rules for input and output describe one way of decomposing a synchronisation. The last two rules describe another way, where an input on a and an output on c both occur for the same agent, which cannot verify locally that $a = c$. Again, we only present a particular case of our real rules (actually this is just the case $b \notin \mathrm{Im}(h)$). These rules are reminiscent of Rathke and Sobociński [43] (for input/output), and Crafa et al. [7] (for partial synchronisations).

We have already mentioned that $\mathcal{S}_{\mathbb{D}}$ may be viewed as an LTS $\mathcal{S}_{\mathbb{D}}^{\mathbb{A}}$ over $\mathbb{A}$. It is not too much work to also view $Pi^{\mathbb{A}}$ as an LTS over $\mathbb{A}$. Next, we define when two transitions in $\mathbb{A}$ are complementary, i.e., are the restrictions of a closed-world transition. This gives the right notion of complementarity for both $Pi^{\mathbb{A}}$ and $\mathcal{S}_{\mathbb{D}}^{\mathbb{A}}$, so that fair testing equivalence in Pi and $\mathcal{S}_{\mathbb{D}}$ may be checked in terms of transitions over $\mathbb{A}$. Thus, in order to check whether an agent P passes a test T, e.g., instead of considering transition sequences $P \mid T \Rightarrow Q$, one may consider complementary sequences $P \overset{w}{\Rightarrow}_{\mathbb{A}} P'$ and $T \overset{w'}{\Rightarrow}_{\mathbb{A}} T'$ such that $Q = P' \mid T'$.

Thanks to this, one reduces to proving that the translation $[\![-]\!] \colon Pi \to \mathcal{S}_{\mathbb{D}}$ is surjective up to weak bisimilarity (except for empty strategies), which ensures that there are enough tests in Pi. For this, the only subtlety is that in Pi, ν is a standalone construct, which may not be part of a guarded sum, while in $\mathcal{S}_{\mathbb{D}}$ it is treated exactly as inputs, outputs, and ticks. This is dealt with by encoding any guarded sum $\nu.P + \ldots$ as, informally, $\nu c.(\bar{c}.\nu.P + \ldots)$. □

In the course of our proof, we have shown that almost all strategies are weakly bisimilar, hence fair testing equivalent, to some $[\![P]\!]$. Actually, the only strategy which is not is $\emptyset$, which is in fact fair testing equivalent to $[\![\heartsuit]\!]$! This entails

Theorem 5. *For all strategies S over $[\Gamma]$, there exists a process $\Gamma \vdash P$ such that $[\![\Gamma \vdash P]\!] \sim^{\mathbb{D}} ([\Gamma], S)$.*

Theorems 4 and 5 together are the desired full abstraction result.

References

1. S. Abramsky and P.-A. Melliès. Concurrent games and full completeness. In LICS 1999 [32], pages 431–442.
2. M. M. Bonsangue, J. J. M. M. Rutten, and A. Silva. A Kleene theorem for polynomial coalgebras. In *FoSSaCS*, volume 5504 of *LNCS*, pages 122–136. Springer, 2009.
3. A. K. Bousfield. Constructions of factorization systems in categories. *Journal of Pure and Applied Algebra*, 9(2-3):287–329, 1977.
4. E. Brinksma, A. Rensink, and W. Vogler. Fair testing. In *CONCUR*, volume 962 of *LNCS*, pages 313–327. Springer, 1995.
5. D. Cacciagrano, F. Corradini, and C. Palamidessi. Explicit fairness in testing semantics. *Logical Methods in Computer Science*, 5(2), 2009.
6. G. L. Cattani, I. Stark, and G. Winskel. Presheaf models for the pi-calculus. In *Category Theory and Computer Science*, volume 1290 of *LNCS*, pages 106–126. Springer, 1997.
7. S. Crafa, D. Varacca, and N. Yoshida. Event structure semantics of parallel extrusion in the pi-calculus. In *FoSSaCS*, volume 7213 of *LNCS*, pages 225–239. Springer, 2012.
8. R. De Nicola and M. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.
9. M. Delorme, J. Mazoyer, N. Ollinger, and G. Theyssier. Bulking I: An abstract theory of bulking. *Theoretical Computer Science*, 412(30):3866–3880, 2011.
10. C. Ehresmann. *Catégories et structures*. Dunod, 1965.
11. U. Engberg and M. Nielsen. A calculus of communicating systems with label passing. Technical Report PB-208, Aarhus University, 1986.
12. M. P. Fiore, E. Moggi, and D. Sangiorgi. A fully-abstract model for the pi-calculus (extended abstract). In LICS 1996 [31], pages 43–54.
13. M. P. Fiore and S. Staton. A congruence rule format for name-passing process calculi from mathematical structural operational semantics. In *LICS*, pages 49–58. IEEE Computer Society, 2006.
14. M. P. Fiore and D. Turi. Semantics of name and value passing. In *LICS*, pages 93–104. IEEE Computer Society, 2001.
15. P. Freyd and G. Kelly. Categories of continuous functors, I. *Journal of Pure and Applied Algebra*, 2:169–191, 1972.
16. R. Garner. *Polycategories*. PhD thesis, University of Cambridge, 2006.
17. D. R. Ghica and A. S. Murawski. Angelic semantics of fine-grained concurrency. In *FoSSaCS*, volume 2987 of *LNCS*, pages 211–225. Springer, 2004.
18. J.-Y. Girard. Locus solum: From the rules of logic to the logic of rules. *Mathematical Structures in Computer Science*, 11(3):301–506, 2001.
19. M. Grandis and R. Paré. Limits in double categories. *Cahiers de Topologie et Géométrie Différentielle Catégoriques*, 40(3):162–220, 1999.
20. R. Harmer, M. Hyland, and P.-A. Melliès. Categorical combinatorics for innocent strategies. In *LICS*, pages 379–388. IEEE Computer Society, 2007.
21. R. Harmer and G. McCusker. A fully abstract game semantics for finite nondeterminism. In LICS 1999 [32], pages 422–430.
22. M. Hennessy. A fully abstract denotational semantics for the pi-calculus. *Theoretical Computer Science*, 278(1-2):53–89, 2002.
23. T. T. Hildebrandt. Towards categorical models for fairness: fully abstract presheaf semantics of SCCS with finite delay. *Theoretical Computer Science*, 294(1/2):151–181, 2003.

24. T. Hirschowitz. Full abstraction for fair testing in CCS. In *CALCO*, volume 8089 of *LNCS*, pages 175–190. Springer, 2013. Long version submitted.
25. T. Hirschowitz and D. Pous. Innocent strategies as presheaves and interactive equivalences for CCS. *Scientific Annals of Computer Science*, 22(1):147–199, 2012. Selected papers from ICE '11.
26. J. M. E. Hyland and C.-H. L. Ong. On full abstraction for PCF: I, II, and III. *Inf. Comput.*, 163(2):285–408, 2000.
27. B. Jacobs. *Categorical Logic and Type Theory*. Number 141 in Studies in Logic and the Foundations of Mathematics. North Holland, Amsterdam, 1999.
28. A. Joyal. Factorisation systems. `http://ncatlab.org/joyalscatlab`.
29. J. Laird. Game semantics for higher-order concurrency. In *FSTTCS*, volume 4337 of *LNCS*, pages 417–428. Springer, 2006.
30. T. Leinster. *Higher Operads, Higher Categories*, volume 298 of *London Mathematical Society Lecture Notes*. Cambridge University Press, Cambridge, 2004.
31. *11th Symposium on Logic in Computer Science*. IEEE Computer Society, 1996.
32. *14th Symposium on Logic in Computer Science*. IEEE Computer Society, 1999.
33. S. MacLane and I. Moerdijk. *Sheaves in Geometry and Logic: A First Introduction to Topos Theory*. Universitext. Springer, 1992.
34. P.-A. Melliès. Game semantics in string diagrams. In *LICS*, pages 481–490. IEEE, 2012.
35. P.-A. Melliès and S. Mimram. Asynchronous games: Innocence without alternation. In *CONCUR*, volume 4703 of *LNCS*, pages 395–411. Springer, 2007.
36. R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.
37. R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, I/II. *Information and Computation*, 100(1):1–77, 1992.
38. U. Montanari and M. Pistore. Concurrent semantics for the pi-calculus. *Electronic Notes in Theoretical Computer Science*, 1:411–429, 1995.
39. V. Natarajan and R. Cleaveland. Divergence and fair testing. In *ICALP*, volume 944 of *LNCS*, pages 648–659. Springer, 1995.
40. H. Nickau. Hereditarily sequential functionals. In *LFCS*, volume 813 of *LNCS*, pages 253–264. Springer, 1994.
41. R. Paré. Yoneda theory for double categories. *Theory and Applications of Categories*, 25(17):436–489, 2011.
42. A. Popescu. A fully abstract coalgebraic semantics for the pi-calculus under weak bisimilarity. Technical Report UIUCDCS-R-2009-3045, University of Illinois, 2009.
43. J. Rathke and P. Sobocinski. Deconstructing behavioural theories of mobility. In *IFIP TCS*, volume 273 of *IFIP*, pages 507–520. Springer, 2008.
44. S. Rideau and G. Winskel. Concurrent strategies. In *LICS '11*. IEEE Computer Society, 2011.
45. D. Sangiorgi. A theory of bisimulation for the pi-calculus. *Acta Informatica*, 33(1):69–97, 1996.
46. I. Stark. A fully abstract domain model for the pi-calculus. In LICS 1996 [31], pages 36–42.
47. G. Winskel. Strategies as profunctors. In *FoSSaCS*, volume 7794 of *LNCS*, pages 418–433. Springer, 2013.