HAL
open science

Innocent strategies as presheaves and interactive equivalences for CCS (expanded version)

Tom Hirschowitz, Damien Pous

To cite this version:

Tom Hirschowitz, Damien Pous. Innocent strategies as presheaves and interactive equivalences for CCS (expanded version). Scientific Annals of Computer Science, 2012, 22 (1), pp.147-199. 10.7561/SACS.2012.1.147 . hal-00555144v3

HAL Id: hal-00555144

https://hal.science/hal-00555144v3

Submitted on 12 Dec 2012

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.

Innocent strategies as presheaves and interactive equivalences for CCS[1]

Tom HIRSCHOWITZ[2] and Damien POUS[3]

Abstract

Seeking a general framework for reasoning about and comparing programming languages, we derive a new view of Milner's CCS [34]. We construct a category $\mathbb{E}$ of *plays*, and a subcategory $\mathbb{V}$ of *views*. We argue that presheaves on $\mathbb{V}$ adequately represent *innocent* strategies, in the sense of game semantics [20]. We equip innocent strategies with a simple notion of interaction.

We then prove decomposition results for innocent strategies, and, restricting to presheaves of finite ordinals, prove that innocent strategies are a final coalgebra for a polynomial functor [27] derived from the game. This leads to a translation of CCS with recursive equations.

Finally, we propose a notion of *interactive equivalence* for innocent strategies, which is close in spirit to Beffara's interpretation [1] of testing equivalences [7] in concurrency theory. In this framework, we consider analogues of *fair* testing and *must* testing. We show that must testing is strictly finer in our model than in CCS, since it avoids what we call 'spatial unfairness'. Still, it differs from fair testing, and we show that it coincides with a relaxed form of fair testing.

Note: This is an expanded version of our ICE '11 paper [19]. It notably simplifies a few aspects of the development, and corrects the mistaken statement that fair and must testing coincide in our semantic framework. Must testing only coincides with a relaxed variant of fair testing. This version also subsumes a previous preprint, providing more compact proofs.

1 Overview

Theories of programming languages Research in programming languages is mainly technological. Indeed, it heavily relies on techniques which

[1]Both authors have been partially funded by the French projects CHoCo (ANR-07-BLAN-0324), PiCoq (ANR-10-BLAN-0305-01), and CNRS PEPS CoGIP.

[2]CNRS, Université de Savoie, France, `tom.hirschowitz@univ-savoie.fr`

[3]CNRS, Laboratoire d'Informatique de Grenoble, France, `damien.pous@ens-lyon.fr`

are ubiquitous in the field, but almost never formally made systematic. Typically, the definition of a language then quotiented by variable renaming (α-conversion) appears in many theoretical papers about functional programming languages. Why isn't there yet any abstract framework performing these systematic steps for you? Because the quest for a real *theory* of programming languages is not achieved yet, in the sense of a corpus of results that actually help developing them or reasoning about them. However, many attempts at such a theory do exist.

A problem for most of them is that they do not account for the dynamics of execution, which limits their range of application. This is for example the case of Fiore et al.'s second-order theories [10, 15, 16]. A problem for most of the other theories of programming languages is that they neglect denotational semantics, i.e., they do not provide a notion of model for a given language. This is for example the case of Milner et al.'s *bigraphs* [22], or of most approaches to *structural operational semantics* [37], with the notable exception of the *bialgebraic semantics* of Turi and Plotkin [41]. A recent, related, and promising approach is *Kleene coalgebra*, as advocated by Bonsangue et al. [2]. Finally, *higher-order rewriting* [36], and its semantics in double categories [12] or in cartesian closed 2-categories [18], is not currently known to adequately account for process calculi.

Towards a new approach The most relevant approaches to us are bialgebraic semantics and Kleene coalgebra, since the programme underlying the present paper concerns a possible alternative. A first difference, which is a bit technical but may be of importance, is that both bialgebraic semantics and Kleene coalgebra are based on labelled transition systems (LTSs), while our approach is based on reduction semantics. This seems relevant, since reduction semantics is often considered more primitive than LTSs, and much work has been devoted to deriving the latter from the former [40, 29, 22, 39, 38].

More generally, our approach puts more emphasis on interaction between programs, and hence is less interesting in cases where there is no interaction. A sort of wild hope is that this might lead to unexpected models of programming languages, e.g., physical ones. This could also involve finding a good notion of morphism between languages, and possibly propose a notion of compilation. At any rate, the framework is not set up yet, so investigating the precise relationship with bialgebraic semantics and Kleene coalgebra is deferred to further work.

How will this new approach look like? Compared to such long-term goals, we only take a small step forward here, by considering a particular case, namely Milner's CCS [34], and providing a new view of it. This view borrows ideas from the following lines of research: game semantics [20], and in particular the notion of an *innocent strategy*, *graphical games* [8, 17], Krivine realisability [28], ludics [13], testing equivalences in concurrency [7, 1], the presheaf approach to concurrency [24, 25], and sheaves [32]. It is also, more remotely, related to graph rewriting [9] and computads [4].

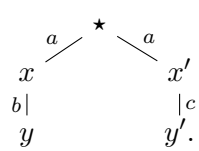
From strategies to presheaves Game semantics [20] has provided fully complete models of programming languages. It is based on the notion of a *strategy*, i.e., a set of *plays* in some game, satisfying a few conditions. In concurrency theory, taking as a semantics the set of accepted plays, or 'traces', is known as *trace semantics*. Trace semantics is generally considered too coarse, since it equates, for a most famous example, the right and the wrong coffee machines, $a.(b+c)$ and $ab+ac$ [34].

An observation essentially due to Joyal, Nielsen, and Winskel is that strategies, i.e., prefix-closed sets of plays, are actually particular *presheaves of booleans* on the category $\mathbb{C}$ with plays as objects, and prefix inclusions as morphisms. By presheaves of booleans on $\mathbb{C}$ we here mean functors $\mathbb{C}^{op} \to 2$, where 2 is the preorder category $0 \leqslant 1$. If a play p is *accepted*, i.e., mapped to 1, then its prefix inclusions $q \hookrightarrow p$ are mapped to the unique morphism with domain 1, i.e., id_1, which entails that q is also accepted.

Following Joyal, Nielsen, and Winskel, we observe that considering instead presheaves (of sets) on $\mathbb{C}$ yields a much finer semantics. So, a play p is now mapped to a set $S(p)$, to be thought of as the set of ways for p to be accepted by the strategy S. Considering the set of players as a team, $S(p)$ may also be thought of as the set of possible *states* of the team after playing p – which is empty if the team never accepts to play p.

This presheaf semantics is fine enough to account for bisimilarity [24, 25]. Indeed, presheaves are essentially forests with edges labelled by moves. For example, in the setting where plays are finite words on an alphabet, the wrong coffee machine may be represented by the presheaf S defined by the equations on the left and pictured as on the right:

$$\begin{array}{ll} S(\epsilon) = \{\star\}, & \\ S(a) = \{x, x'\}, & S(\epsilon \hookrightarrow a) = \{x \mapsto \star, x' \mapsto \star\}, \\ S(ab) = \{y\}, & S(a \hookrightarrow ab) = \{y \mapsto x\}, \\ S(ac) = \{y'\}, & S(a \hookrightarrow ac) = \{y' \mapsto x'\} : \end{array}$$



So, in summary: the standard notion of strategy may be generalised to account for branching equivalences, by passing from presheaves of booleans to presheaves of sets.

Multiple players Traditional game semantics mostly emphasises two-player games. There is an implicit appearance of three-player games in the definition of composition of strategies, and of four-player games in the proof of its associativity, but these games are never given a proper status. A central idea of graphical games, and to a lesser extent of ludics, is the emphasis on multiple-player games.

Here, there first is a base category $\mathbb{B}$ of *positions*, whose objects represent configurations of players. Since the game represents CCS, it should be natural that players are related to each other via the knowledge of *communication channels*. So, roughly, positions are bipartite graphs with vertex sets *players* and *channels*, and edges from channels to players indicating when the former is known to the latter. As a first approximation, morphisms of positions may be thought of as just embeddings of such graphs.

Second, there is a category $\mathbb{E}$ of *plays*, with a functor to $\mathbb{B}$ sending each play to its initial position. Plays are represented in a more flexible way than just sequences of moves, namely using a kind of string diagrams. This echoes the idea [33] that two moves may be independent, and that plays should not depend on the order in which two independent moves are performed. Furthermore, our plays are a rather general notion, allowing, e.g., to focus on a given player. Morphisms of plays account both for:

- prefix inclusion, i.e., inclusion of a play into a longer play, and
- position enlargement, e.g., inclusion of information about some players into information about more players.

Now, restricting to plays over a given initial position X, and then taking presheaves on this category $\mathbb{E}_X$, we have a category of strategies on X.

Innocence A fundamental idea of game semantics is the notion of *innocence*, which says that players have a restricted *view* of the play, and that their actions may only depend on that view.

We implement this here by defining a subcategory $\mathbb{V}_X \hookrightarrow \mathbb{E}_X$ of *views* on X, and deeming a presheaf F on $\mathbb{E}_X$ *innocent* when it is determined by

its restriction F' to $\mathbb{V}_X$, in the sense that it is isomorphic to the *right Kan extension* [31] of F' along $\mathbb{V}_X^{op} \hookrightarrow \mathbb{E}_X^{op}$.

We then define *innocent strategies* to be just presheaves on $\mathbb{V}_X$, and view them as (naive) strategies via the (essential) embedding $\widehat{\mathbb{V}_X} \hookrightarrow \widehat{\mathbb{E}_X}$ induced by right Kan extension.

Interaction For each position X, we thus have a category $\mathsf{S}_X = \widehat{\mathbb{V}_X}$ of innocent strategies. In game semantics, composition of strategies is achieved in two steps: *interaction* and *hiding*. Essentially, interaction amounts to considering the three-player game obtained by letting two two-player games interact at a common interface. Hiding then forgets what happens at that interface, to recover a proper two-player game.

We have not yet investigated hiding in our approach, but, thanks to the central status of multiple-player games, interaction is accounted for in a very streamlined way. For any position X with two subpositions $X_1 \hookrightarrow X$ and $X_2 \hookrightarrow X$ such that each player is in either X_1 or X_2, but none is in both, given innocent strategies $F_1 \in \mathsf{S}_{X_1}$ and $F_2 \in \mathsf{S}_{X_2}$, there is a unique innocent strategy, the *amalgamation* $[F_1, F_2]$ of F_1 and F_2, whose restrictions to X_1 and X_2 are F_1 and F_2.

Amalgamation in this sense models interaction in the sense of game semantics, and, using the correspondence with presheaves on $\mathbb{E}_X$ given by right Kan extension, it is the key to defining interactive equivalences.

CCS Next, we define a translation of CCS terms with recursive equations into innocent strategies. This rests on *spatial* and *temporal* decomposition results for innocent strategies. Spatial decomposition says that giving a strategy on a position X is the same as giving a strategy for each of its players. Temporal decomposition says that a strategy is determined up to isomorphism by its set of initial states, plus what remains of each of them after each basic move. Restricting to presheaves of finite ordinals, we also prove that innocent strategies form a final coalgebra for a polynomial functor (in the sense of Kock [27]) derived from the game, thus hinting at links with Kleene coalgebra. It is then easy to translate finite CCS into the language induced by our polynomial functor, and to finally extend the translation to CCS with recursive equations via infinite unfolding.

A natural question is then: which equivalence does this translation induce on CCS terms? As explained in the following paragraph, we provide some preliminary results about interactive equivalences, but essentially leave

the question open.

Interactive equivalences Returning to our model, we then define a notion of *interactive equivalence*, which is close in spirit to both testing equivalences in concurrency theory and Krivine realisability and ludics.

The game, as sketched above, allows interacting with players which are not part of the considered position. E.g., a player in the considered position X may perform an input which is not part of any synchronisation. A *test* for an innocent strategy F on X is then, roughly, an innocent strategy G on a position X' with the same channels as X. To decide whether F *passes* the test G, we consider a restricted variant of the game on the 'union' $X \cup X'$, forbidding any interaction with the outside. We call that variant the *closed-world* game.

Then F passes G iff the amalgamation $[F, G]$, right Kan extended to $\mathbb{E}_{X \cup X'}$ and then restricted to the closed-world game, belongs to some initially fixed class of strategies, $\perp\!\!\!\perp_{X \cup X'}$. Finally, two innocent strategies F and F' on X are equivalent when they pass the same tests.

Here are two examples for $\perp\!\!\!\perp$. Consider a *tick* move, fixed in advance. Then call *successful* all plays containing at least one tick, and accordingly call successful all states reached after a successful play. One may consider:

- $\perp\!\!\!\perp^m$, consisting of strategies whose maximal states (those that admit no strict extensions) are all successful; the tick move plays a rôle analogous to the daimon in ludics: it is the only move which is observable from the outside;
- $\perp\!\!\!\perp^f$, consisting of strategies in which all states on *finite* plays admit a successful extension.

From the classical concurrency theory point of view on behavioural equivalences, the first choice clearly mimicks *must* testing equivalence, while the second mimicks *fair* testing equivalence [35, 3].

Consider the processes Ω and $\Omega|\overline{a}$, where Ω is a process doing infinitely many silent transitions. These processes are intuitively quite different: the latter can do an output on the channel a, while the former cannot. They are however equated by standard must testing equivalence: the infinite trace provided by Ω may prevent the output prefix from being performed. In fact, must testing equivalence heavily relies on the potential unfairness of the scheduler. In the literature, this peculiar behaviour actually motivates the introduction of fair testing equivalence.

In contrast, our notion of play is more flexible than standard traces, so that our counterpart to must testing equivalence actually distinguishes these two processes: the infinite play where the output prefix is not performed is not maximal, so that the corresponding unfair behaviour is not taken into account. In other words, thanks to our notion of play, the rather natural notion of must testing already avoids what we call 'spatial unfairness'. However, must testing does not coincide with fair testing in our setting, because there are other sources of unfairness, that are not handled properly. Technically, we prove that $\perp\!\!\!\perp^m$ coincides with the set of strategies whose states all admit a successful extension. However, the restriction to finite plays in the definition of $\perp\!\!\!\perp^f$ is required to rule out other sources of unfairness.

Summary In summary, our approach emphasises a flexible notion of multiple-player play, encompassing both views in the sense of game semantics, closed-world plays, and intermediate notions. Strategies are then described as presheaves on plays, while innocent strategies are presheaves on views. Innocent strategies admit a notion of interaction, or amalgamation, and are embedded into strategies via right Kan extension. This allows a notion of testing, or interactive equivalence by amalgamation with the test, right Kan extension, and finally restriction to closed-world.

Our main technical contributions are then a translation of CCS terms with recursive equations into innocent strategies, and the study of fair and must equivalences in our setting.

Perspectives Our next task is clearly to tighten the link with CCS. Namely, we should explore which equivalence on CCS is induced via our translation, for a given interactive equivalence. We will start with $\perp\!\!\!\perp^f$. Furthermore, the very notion of interactive equivalence might deserve closer consideration. Its current form is rather *ad hoc*, and one could hope to see it emerge more naturally from the game. For instance, the fixed class $\perp\!\!\!\perp$ of 'successful' strategies should probably be more constrained than is done here. Also, the paradigm of observing via the set of successful tests might admit sensible refinements, e.g., probabilistic ones.

Another possible research direction is to tighten the link with 'graphical' approaches to rewriting, such as graph rewriting or computads. E.g., our plays might be presented by a computad [14], or be the bicategory of rewrite sequences up to shift equivalence, generated by a graph grammar in the sense of Gadducci et al. [11]. Both goals might require some technical adjustments,

however. For computads, we would need the usual yoga of U-turns to flexibly model our positions; e.g., zigzags of U-turns are usually only equal up to a higher-dimensional cell, while they would map to equal positions in our setting. For graph rewriting, the problem is that our positions are not exactly graphs (e.g., the channels known to a player are linearly ordered).

Other perspectives include the treatment of more complicated calculi like π or λ. In particular, calculi with duplication of terms will pose a serious challenge. An even longer-term hope is to be able to abstract over our approach. Is it possible to systematise the process starting from a calculus as studied in programming language theory, and generating its strategies modulo interactive equivalence? If this is ever understood, the next question is: when does a translation between two such calculi preserve a given interactive equivalence? Finding general criteria for this might have useful implications in programming languages, especially compilation.

Notation Throughout the paper, we abusively identify n with $\{1 \dots n\}$, for readability. So, e.g., $i \in n$ means $i \in \{1, \dots, n\}$.

$$\begin{array}{ccc} FC & \xrightarrow{F(f)} & FC' \\ {\scriptstyle u}\downarrow & & \downarrow{\scriptstyle u'} \\ GD & \xrightarrow[G(g)]{} & GD' \end{array}$$

The various categories and functors constructed in the development are summed up with a short description in Table 1. There, given two functors $\mathbb{C} \xrightarrow{F} \mathbb{E} \xleftarrow{G} \mathbb{D}$, we denote (slightly abusively) by $\mathbb{C} \downarrow_{\mathbb{E}} \mathbb{D}$ the *comma* category: it has as objects triples (C, D, u) with $C \in \mathbb{C}$, $D \in \mathbb{D}$, and $u \colon F(C) \to G(D)$ in $\mathbb{E}$, and as morphisms $(C, D, u) \to (C', D', u')$ pairs (f, g) making the square above commute. Also, when F is the identity on $\mathbb{C}$ and $G \colon 1 \to \mathbb{C}$ is an object C of $\mathbb{C}$, this yields the usual *slice* category, which we abbreviate as $\mathbb{C}/C$. Finally, the category of presheaves on any category $\mathbb{C}$ is denoted by $\widehat{\mathbb{C}} = [\mathbb{C}^{op}, \mathsf{Set}]$.

We denote by $\mathrm{ob}(\mathbb{C})$ the set of objects of any small category $\mathbb{C}$. For any functor $F \colon \mathbb{C} \to \mathbb{D}$, we denote by $\mathbb{F}^{op} \colon \mathbb{C}^{op} \to \mathbb{D}^{op}$ the functor induced on opposite categories, defined exactly as F on both objects and morphisms. Also, recall that an *embedding* of categories is an injective-on-objects, faithful functor. This admits the following generalisation: a functor $F \colon \mathbb{C} \to \mathbb{D}$ is *essentially injective on objects* when $FC \cong FC'$ implies $C \cong C'$. Any faithful, essentially injective on objects functor is called an *essential embedding*.

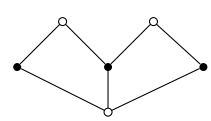
Category	Description of its objects
$\widehat{\mathbb{C}}$	'diagrams'
$\mathbb{B} \hookrightarrow \widehat{\mathbb{C}}$	positions
$\mathbb{E} \hookrightarrow (\mathbb{B} \downarrow_{\widehat{\mathbb{C}}} \widehat{\mathbb{C}})$	plays
$\mathbb{E}_X = (\mathbb{E} \downarrow_{\mathbb{B}} (\mathbb{B}/X))$	plays on a position X
$\mathbb{V}_X \hookrightarrow \mathbb{E}_X$	views on X
$\mathsf{S}_X = \widehat{\mathbb{V}_X}$	innocent strategies on X
$\mathbb{W} \hookrightarrow \mathbb{E}$	closed-world plays
$\mathbb{W}(X)$	closed-world plays on X

Table 1: Summary of categories and functors

2 Plays as string diagrams

We now describe our approach more precisely, starting with the category of multiple-player plays. For the sake of clarity, we first describe this category in an informal way, before giving the precise definition (Section 3).

2.1 Positions



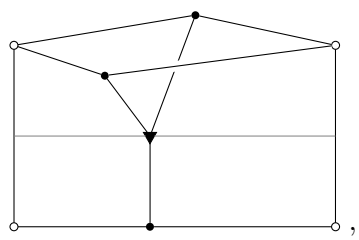
Since the game represents CCS, it should be natural that players are related to each other via the knowledge of *communication channels*. This is represented by a kind of[4] finite, bipartite graph: an example position is on the right. Bullets represent players, circles represent channels, and edges indicate when a player knows a channel. The channels known by a player are linearly ordered. Formally, as explained in Section 3, positions are presheaves over a certain category $\mathbb{C}_1$. Morphisms of positions are natural transformations, which are roughly morphisms of graphs, mapping players to players and channels to channels. In full generality, morphisms thus do not have to be injective, but include in particular embeddings of positions in the intuitive sense. Positions and morphisms between them form a category $\mathbb{B}$.

[4]Only 'a kind of', because, as mentioned above, the channels known to a player are linearly ordered.

2.2 Moves

Plays will be defined as glueings of *moves* between positions. Moves are derived from the very definition of CCS, as we now sketch. The diagrams we draw in this section will be given a very precise combinatorial definition in Section 3.

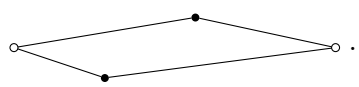
Let us start with the *forking* move, which corresponds to parallel composition in CCS: a process (the player) forks into two sub-processes. In the case of a player knowing two channels, the forking move is represented by the diagram



(1)

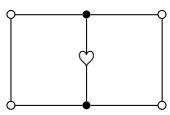
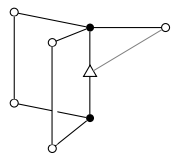
to be thought of as a move from the bottom position X

(with one player p) to the top position Y



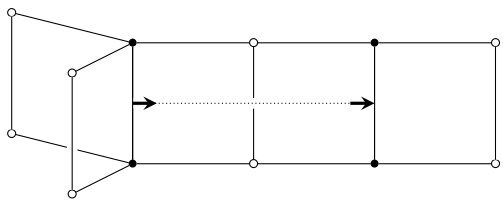
(with two players, which we call the 'avatars' of p). The left- and right-hand borders are just channels evolving in time, not noticing that the represented player forks into two. The surfaces spread between those vertical lines represent links (edges in the involved positions) evolving in time. For example, each link here divides into two when the player forks, thus representing the fact that both of the avatars retain knowledge of the corresponding channel. There is of course an instance π_n of forking for each n, according to the number of channels known to the player. As for channels known to a player, the players and channels touching the black triangle are ordered: there are different 'ports' for the initial player and its two avatars.

We then have a *tick* move $\heartsuit_n$, whose role is to define successful plays, and a move for the *channel creation* or *restriction* of CCS, here ν_n. In the case where the player knows two channels, they are graphically represented as

 and 

respectively. As expected, there is an instance of each of these two moves for each number n of channels known to the player.

We also need a move to model CCS-like synchronisation, between two players. For all n and m, representing the numbers of channels known to the players involved in the synchronisation, and for all $i \in n$, $j \in m$, there is a *synchronisation* $\tau_{n,i,m,j}$, represented, in the case where one player outputs on channel $3 \in 3$ and the other inputs on channel $1 \in 2$, by

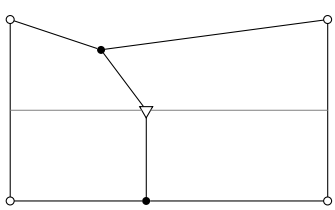
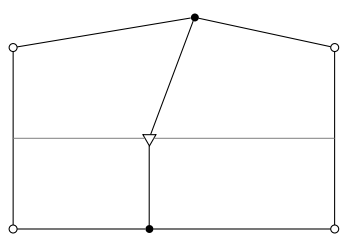


As we shall see in Section 3, the dotted wire in the picture is actually a point in the formal representation (i.e., an element of the corresponding presheaf).

The above four kinds of moves (forking, tick, channel creation, and synchronisation) come from the reduction semantics of CCS. We classify these as *closed-world* moves, since they correspond to the evolution of a group of players in isolation.

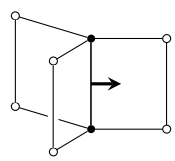
We however need a more fine-grained structure for moves: moves whose final position has more than one player (forking and synchronisation) must be decomposed into *basic* moves, to get an appropriate notion of view.

We introduce two sub-moves for forking: *left* and *right half-forking*. In the case where the player knows two channels, they are represented by the following diagrams, respectively:

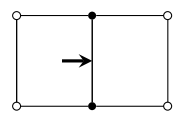
 and  (2)

These sub-moves represent what each of the 'avatars' of the forking player sees of the move. We call π^l_n and π^r_n the respective instances of the left-hand and right-hand basic moves for a player knowing n channels. Formally, there will be injections from the left and right half-forking moves to the corresponding forking moves.

We finally decompose synchronisation into an input move and an output move: $a.P$ and $\overline{a}.P$ in CCS become $o_{n,i}$ and $\iota_{n,i}$ here (where n is the number of known channels, $i \in \{1 \dots n\}$ is the index of the channel bearing the synchronisation). Here, output on the right-hand channel and input on the left-hand channel respectively look like



and



. (3)

Like with forking, there will be injections from the input and output moves to the corresponding synchronisation moves.

All in all, there are three classes of moves, which we summarise in Table 2.

- Tick, channel creation, half-forking, and input/output moves are *basic* moves: they evolve from a position with exactly one player to another position with exactly one player. These moves are used to define views later on.
- Forking, synchronisation, tick and channel creation moves are *closed-world* moves: they correspond to the case where a group of players evolves on its own, in isolation; they are central to the notion of interactive equivalence.
- We need a third class of moves, called *full*, which consists of forking, input, output, tick and channel creation. They involve a single player and all of its avatars. They appear, e.g., in the statement of Lemma 12, which is a partial correctness criterion for closed-world plays.

Formally, we define moves as cospans $X \hookrightarrow P \hookleftarrow Y$ in the category of diagrams (technically a presheaf category $\widehat{\mathbb{C}}$—see Section 3), where X is the initial position and Y the final one. Both legs of the cospan are actually monic morphisms in $\widehat{\mathbb{C}}$, as will be the case for all cospans considered here.

Basic	Full	Closed-world
Left half-forking Right half-forking	Forking	Forking
Input Output	Input Output	Synchronisation
Channel creation	Channel creation	Channel creation
Tick	Tick	Tick

Table 2: Summary of classes of moves

2.3 Plays

We now sketch how plays are defined as glueings of moves. We start with the following example, depicted in Figure 1. The initial position consists of two players p_1 and p_2 sharing knowledge of a channel a, each of them knowing another channel, resp. a_1 and a_2. The play consists of four moves: first p_1 forks into $p_{1,1}$ and $p_{1,2}$, then p_2 forks into $p_{2,1}$ and $p_{2,2}$, and then $p_{1,1}$ does a left half-fork into $p_{1,1,1}$; finally $p_{1,1,1}$ synchronises (as the sender) with $p_{2,1}$. Now, we reach the limits of the graphical representation, but the order in which p_1 and p_2 fork is irrelevant: if p_2 forks before p_1, we obtain the same play. This means that glueing the various parts of the picture in Figure 1 in different orders formally yields the same result (although there are subtle issues in representing this result graphically in a canonical way).

Let us now sketch a definition of plays. Recall that moves may be seen as cospans $X \hookrightarrow M \hookleftarrow Y$, and consider an *extended* notion of move, which may occur in a position not limited to players involved in the move. For example, the moves in Figure 1 are extended moves in this sense.

Definition 1 *A* play *is an embedding $X_0 \hookrightarrow U$ in the category $\widehat{\mathbb{C}}$ of diagrams, isomorphic to a possibly denumerable 'composition' of moves in the (bi)category $\mathsf{Cospan}(\widehat{\mathbb{C}})$ of cospans in $\widehat{\mathbb{C}}$, i.e., obtained as a colimit:*

$$\begin{tikzcd}
X_0 \arrow[dr, hook] & & X_1 \arrow[dl, hook] & \cdots & X_n \arrow[dr, hook] & & X_{n+1} \arrow[dl, hook] \arrow[dr, hook] & & X_{n+2} \arrow[dl, hook] & \cdots \\
& M_0 \arrow[drrr, hook] & & \cdots & & M_n \arrow[dr] & & M_{n+1} \arrow[dll, hook] & \cdots & \\
& & & & & & U, & & &
\end{tikzcd}$$

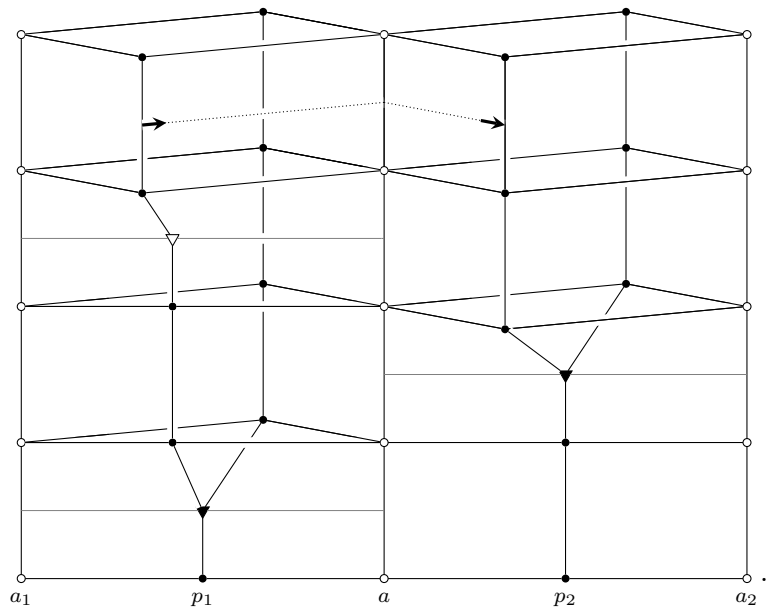


Figure 1: An example play

where each $X_i \hookrightarrow M_i \hookleftarrow X_{i+1}$ *is an extended move.*

We often denote plays just by U, leaving the embedding $X \hookrightarrow U$ implicit.

Remark 1 *For finite plays, one might want to keep track not only of the initial position, but also of the final position. This indeed makes sense. Finite plays then compose 'vertically', and form a double category. But infinite plays do not really have any final position, which explains our definition.*

Let a morphism $(X \hookrightarrow U) \to (Y \hookrightarrow V)$ of plays be a pair (h, k) making the diagram on the right commute in $\widehat{\mathbb{C}}$. This permits both inclusion 'in width' and 'in height'. E.g., the play consisting of the left-hand basic move in (2) embeds in exactly two ways into the play of Figure 1. (Only two because the image of the base position must lie in the base position of the codomain.) We have:

$$\begin{tikzcd} U \arrow[r, "k"] & V \\ X \arrow[u, hook] \arrow[r, "h"] & Y. \arrow[u, hook] \end{tikzcd}$$

Proposition 1 *Plays and morphisms between them form a category* $\mathbb{E}$.

There is a projection functor $\mathbb{E} \to \mathbb{B}$ mapping each play $X \hookrightarrow U$ to its base position X. This functor has a section, which is an embedding $\mathbb{B} \hookrightarrow \mathbb{E}$, mapping each position X to the 'identity' play $X \hookrightarrow X$ on X.

Remark 2 (Size) *The category $\mathbb{E}$ is only locally small. Since presheaves on a locally small category are less well-behaved than on a small category, we will actually consider a skeleton of $\mathbb{E}$. Because $\mathbb{E}$ consists only of denumerable presheaves, this skeleton is a small category. Thus, our presheaves in the next section may be understood as taken on a small category.*

Remark 3 *Plays are not very far from being just (infinite) abstract syntax trees (or forests) 'glued together along channels'. E.g., the play from Figure 1 is the glueing of, say $(\pi_2^l(\overline{a}.0))|0$ and $a|0$ along a.*

2.4 Relativisation

If we now want to restrict to plays over a given base position X, we may consider

Definition 2 *Let the category $\mathbb{E}_X$ have*

- *as objects pairs of a play $Y \hookrightarrow U$ and a morphism $Y \to X$,*
- *as morphisms $(Y \hookrightarrow U) \to (Y' \hookrightarrow U')$ all pairs (h, k) making the diagram*

$$\begin{tikzcd}
U \arrow[rr, "k"] & & U' \\
Y \arrow[u, hook] \arrow[rr, "h"] \arrow[dr] & & Y' \arrow[u, hook] \arrow[dl] \\
& X &
\end{tikzcd}$$

commute in $\widehat{\mathbb{C}}$.

We will usually abbreviate $U \hookleftarrow Y \to X$ as just U when no ambiguity arises. As for morphisms of positions, in full generality, h and k, as well as the morphisms $Y \to X$, do not have to be injective.

Example 1 *Let X be the position* *. The play in Figure 1, say $Y \hookrightarrow U$, equipped with the injection $Y \hookrightarrow X$ mapping the two players of Y to the two leftmost players of X, is an object of $\mathbb{E}_X$.*

One naively could imagine that the objects $\mathbb{E}_X$ could just consist of plays $X \hookrightarrow U$ on X. However, spatial decomposition, Theorem 1, relies on our slightly more complex definition. E.g., still in Figure 1, this allows us to distinguish between the identity view $[2] = [2] \overset{p_1}{\hookrightarrow} X$ on p_1 from the identity view $[2] = [2] \overset{p_2}{\hookrightarrow} X$ on p_2, which would otherwise not be possible.

3 Diagrams

In this section, we define the category on which the string diagrams of the previous section are presheaves. The techniques used here date back at least to Carboni and Johnstone [5, 6].

3.1 First steps

Let us first consider two small examples. It is well-known that directed graphs form a presheaf category: consider the category $\mathbb{C}$ freely generated by the graph with two vertices, say $\star$ and $[1]$, and two edges $d, c \colon \star \to [1]$ between them. One way to visualise this is to compute the *category of elements* of a few presheaves on $\mathbb{C}$. Recall that the category of elements of a presheaf F on $\mathbb{C}$ is the comma category $y \downarrow_{\widehat{\mathbb{C}}} F$, where y is the Yoneda embedding. Via Yoneda, it has as elements pairs (C, x) with $C \in \mathrm{ob}(\mathbb{C})$ and $x \in F(C)$, and morphisms $(C, x) \to (D, y)$ morphisms $f \colon C \to D$ in $\mathbb{C}$ such that $F(f)(y) = x$ (which we abbreviate as $y \cdot f = x$ when the context is clear).

Example 2 *Consider the graph*

$$0 \xrightarrow{\ e\ } 1 \xrightarrow{\ e'\ } 2$$

with three vertices $0, 1,$ and 2, and two edges e and e'.

This graph is represented by the presheaf F defined by the following equations, whose category of elements is actually freely generated by the graph on the right:

- $F(\star) = \{0, 1, 2\}$,
- $F([1]) = \{e, e'\}$,

- $e \cdot d = 0$,
- $e \cdot c = 1$,
- $e' \cdot d = 1$,
- $e' \cdot c = 2$,

$$\begin{tikzcd}
 & & 1 \arrow[dl, "c"'] \arrow[dr, "d"] & & \\
 & e & & e' & \\
0 \arrow[ur, "d"] & & & & 2. \arrow[ul, "c"']
\end{tikzcd}$$

This latter graph is not exactly the original one, but it does represent it. Indeed, for each vertex we know whether it is in $F(\star)$ or $F([1])$, hence whether it represents a 'vertex' or an 'edge'. The arrows all go from a 'vertex' v to an 'edge' e. They lie over d when v is the domain of e, and over c when v is the codomain of e.

Multigraphs, i.e., graphs whose edges have a list of sources instead of just one, may also be seen as a presheaves on the category freely generated by the graph with

- as vertices: one special vertex $\star$, plus for each natural number n a vertex, say, $[n]$; and
- for all $n \in \mathbb{N}$, $n+1$ edges $\star \to [n]$, called $d_1, \ldots, d_n$, and c.

It should be natural for presheaves on this category to look like multigraphs: the elements of a presheaf F over $\star$ are the vertices in the multigraph, the elements over $[n]$ are the n-ary multiedges, and the action of the d_i's give the ith source of a multiedge, while the action of c gives its target.

Example 3 *Similarly, computing a few categories of elements might help visualising. As above, consider F defined by*

- $F(\star) = \{0, 1, 2, 3, 4, 5\}$,
- $F([1]) = F([0]) = \varnothing$,
- $F([2]) = \{e'\}$,
- $F([3]) = \{e\}$,
- $F([n+4]) = \varnothing$,
- $e \cdot c = 0$,
- $e \cdot d_1 = 1$,
- $e \cdot d_2 = 2$,
- $e \cdot d_3 = 3$,
- $e' \cdot c = 1$,
- $e' \cdot d_1 = 4$,
- $e' \cdot d_2 = 5$,

whose category of elements is freely generated by the graph:

$$\begin{tikzcd}
 & & 0 \arrow[d, "c"] & \\
 & & e & \\
 & 1 \arrow[ur, "d_1"] \arrow[d, "c"'] & 2 \arrow[u, "d_2"'] & 3 \arrow[ul, "d_3"'] \\
 & e' & & \\
4 \arrow[ur, "d_1"] & & 5. \arrow[ul, "d_2"'] &
\end{tikzcd}$$

Now, this pattern may be extended to higher dimensions. Consider for example extending the previous base graph with a vertex $[m_1, \ldots, m_n; p]$ for all natural numbers $n, p, m_1, \ldots, m_n$, plus edges

$$\begin{aligned}
&s_1 \colon [m_1] \to [m_1, \ldots, m_n; p], \\
&\ldots, \\
&s_n \colon [m_n] \to [m_1, \ldots, m_n; p], \textit{and} \\
&t \colon [p] \to [m_1, \ldots, m_n; p].
\end{aligned}$$

Let now $\mathbb{C}$ be the free category on this extended graph. Presheaves on $\mathbb{C}$ are a kind of 2-multigraphs: they have vertices, multiedges, and multiedges between multiedges.

We could continue this in higher dimensions.

3.2 Constructing the base category

Our base category follows a very similar pattern. We start from a slightly different graph: let $\mathbb{G}_0$ have just one vertex $\star$; let $\mathbb{G}_1$, have one vertex $\star$, plus a vertex $[n]$ for each natural number n, plus n edges $d_1, \ldots, d_n \colon \star \to [n]$. Let $\mathbb{C}_0$ and $\mathbb{C}_1$ be the categories freely generated by $\mathbb{G}_0$ and $\mathbb{G}_1$, respectively. So, presheaves on $\mathbb{C}_1$ are a kind of hypergraphs with arity (since vertices incident to a hyperedge are numbered). This is enough to model positions.

Example 4 *The position drawn at the beginning of Section 2.1 may be represented as the presheaf*

- $\star \mapsto \{1, 2, 3\}$,
- $[2] \mapsto \{x, z\}$,
- $[3] \mapsto \{y\}$,
- $_ \mapsto \varnothing$,
- $x \cdot d_1 = 1$,
- $x \cdot d_2 = 2$,
- $z \cdot d_1 = 2$,
- $z \cdot d_2 = 3$,
- $y \cdot d_1 = 1$,
- $y \cdot d_2 = 2$,
- $y \cdot d_3 = 3$,

whose category of elements is:

$$\begin{tikzcd}
& 1 \arrow[dl, "d_1"'] \arrow[dr, "d_1"] & & 3 \arrow[dl, "d_3"'] \arrow[dr, "d_2"] & \\
x & & y & & z \\
& & 2. \arrow[ull, "d_2"] \arrow[u, "d_2"] \arrow[urr, "d_1"'] & &
\end{tikzcd}$$

Now, consider the graph $\mathbb{G}_2$, which is $\mathbb{G}_1$ augmented with:

- for all n, vertices $\heartsuit_n$, π^l_n, π^r_n, ν_n,
- for all n and $1 \leqslant i \leqslant n$, vertices $o_{n,i}$ and $\iota_{n,i}$,
- for all n, edges $s, t\colon [n] \to \heartsuit_n$, $s, t\colon [n] \to \pi^l_n$, $s, t\colon [n] \to \pi^r_n$, $s\colon [n] \to \nu_n$, $t\colon [n+1] \to \nu_n$,
- for all n and $1 \leqslant i \leqslant n$, edges $s, t\colon [n] \to o_{n,i}$, $s, t\colon [n] \to \iota_{n,i}$.

We slightly abuse language here by calling all these t's and s's the same. We could label them with their codomain, but we refrain from doing so for the sake of readability.

Now, let $\mathbb{C}_2$ be the category generated by $\mathbb{G}_2$ and the relations $s \circ d_i = t \circ d_i$ for all n and $1 \leqslant i \leqslant n$ (for all sensible—common—codomains). The intuition here is that for any basic move by a player with n channels, these n channels remain the same after the move. This includes the case of ν_n, for which the absence of any equation involving the new channel makes it different from the others.

Example 5 *Again, computing a few categories of elements is in order. For example, the category of elements of (the representable presheaf on) $\iota^-_{3,3}$ is the poset freely generated by the graph*

$$\begin{tikzcd}
td_1 \arrow[rr] \arrow[dd, equal] & & t \arrow[d] & & td_3 \arrow[ll] \arrow[dd, equal] \\
& td_2 \arrow[ur] \arrow[dd, equal] & id_{\iota^-_{3,3}} & & \\
sd_1 \arrow[rr] & & s \arrow[u] & & sd_3 \arrow[ll] \\
& sd_2, \arrow[ur] & & &
\end{tikzcd}$$

to be compared with the corresponding pictures (3).

Example 6 *Similarly, the category of elements of ν_1 is the poset freely generated by the graph*

$$\begin{tikzcd}
td_1 \arrow[r] \arrow[dd, equal] & t \arrow[d] & td_2 \arrow[l] \arrow[dl] \\
 & id_{\nu_1} & \\
sd_1 \arrow[r] & s. \arrow[u] &
\end{tikzcd}$$

Note that only channel creation changes the number of channels known to the player, and accordingly the corresponding morphism t has domain $[n+1]$.

Presheaves on $\mathbb{C}_2$ are enough to model basic moves, but since we want more, we continue, as follows.

Let $\mathbb{G}_3$ be $\mathbb{G}_2$, augmented with:

- for all n, a vertex π_n, and
- edges $l\colon \pi_n^l \to \pi_n$ and $r\colon \pi_n^r \to \pi_n$.

Definition 3 *Let $\mathbb{C}_3$ be the category generated by $\mathbb{G}_3$, the previous relations, plus the relations $l \circ s = r \circ s$.*

The equation models the fact that a forking move should be played by just one player. We also call $s = l \circ s = r \circ s$ the common composite, which gives a uniform notation for the initial player of full moves.

Example 7 *The category of elements of π_2 is the poset freely generated by the graph*

$$\begin{tikzcd}
ltd_1 = rtd_1 \arrow[r] \arrow[rr, bend left] \arrow[dd, equal] & lt \arrow[d] & rt \arrow[d] & ltd_2 = rtd_2 \arrow[l] \arrow[ll, bend right] \arrow[dd, equal] \\
 & l \arrow[r] & id_{\pi_2} & r \arrow[l] \\
lsd_1 = rsd_1 \arrow[rr] & & ls = rs \arrow[ul] \arrow[ur] & lsd_2 = rsd_2. \arrow[ll]
\end{tikzcd}$$

The two views corresponding to left and right half-forking are subcategories, and the object id_{π_2} 'ties them together'.

Presheaves on $\mathbb{C}_3$ are enough to model full moves; to model closed-world moves, and in particular synchronisation, we continue as follows.

Let $\mathbb{G}_4$ be $\mathbb{G}_3$, augmented with, for all n, m, $1 \leqslant i \leqslant n$, and $1 \leqslant j \leqslant m$,

- a vertex $\tau_{n,i,m,j}$, and
- edges $\epsilon\colon o_{n,i} \to \tau_{n,i,m,j}$ and $\rho\colon \iota_{m,j} \to \tau_{n,i,m,j}$ (ϵ and ρ respectively stand for 'emission' and 'reception').

Definition 4 *Let $\mathbb{C}_4$ be the category generated by $\mathbb{G}_4$, the previous relations, plus, for each $o_{n,i} \xrightarrow{\epsilon} \tau_{n,i,m,j} \xleftarrow{\rho} \iota_{m,j}$, the relation $\epsilon \circ s \circ d_i = \rho \circ s \circ d_j$.*

This equation is the exact point where we enforce that a synchronisation involves an input and an output on the same channel, as announced in Example 5.

Example 8 *The category of elements of $\tau_{3,3,1,1}$ is the preorder freely generated by the graph*

$$\begin{tikzcd}
\epsilon t d_1 \arrow[rr] \arrow[dd, equal] & & \epsilon t & \epsilon t d_3 = \rho t d_1 \arrow[l] \arrow[rr] \arrow[d, equal] & & \rho t \arrow[d] & \rho t d_2 \arrow[l] \arrow[dd, equal] \\
& \epsilon t d_2 \arrow[ur] \arrow[dd, equal] & \epsilon \arrow[d] & \epsilon \arrow[rr] & id_{\tau_{3,3,2,1}} & \rho \arrow[l] & \\
\epsilon s d_1 \arrow[rr] & & \epsilon s \arrow[u] & \epsilon s d_3 = \rho s d_1 \arrow[l] \arrow[rr] \arrow[u, equal] & & \rho s \arrow[u] & \rho s d_2 \arrow[l] \\
& \epsilon s d_2. \arrow[ur] & & & & &
\end{tikzcd}$$

Again, the two views corresponding to $\iota^+_{3,3}$ and $\iota^-_{2,1}$ are subcategories, and the new object $\tau_{3,3,2,1}$ ties them together.

3.3 Positions and moves

We have now defined the base category $\mathbb{C} = \mathbb{C}_4$ on which the string diagrams of Section 2 are presheaves. More accurately we have defined a sequence $\mathbb{C}_0 \hookrightarrow \ldots \hookrightarrow \mathbb{C}_4$ of subcategories.

Positions Positions are finite presheaves on $\mathbb{C}_1$, or equivalently, finite presheaves on $\mathbb{C}_4$ empty except over $\mathbb{C}_1$.

Moves Basic moves should be essentially representable presheaves on objects in $\mathrm{ob}(\mathbb{C}_2)\backslash\mathrm{ob}(\mathbb{C}_1)$. Recall however that basic moves are defined as particular cospans in $\widehat{\mathbb{C}}$. This is also easy: in the generating graph $\mathbb{G}_2$, each such object c has exactly two morphisms s and t into it, from objects, say, $[n_s]$ and $[n_t]$, respectively. By Yoneda, these induce a cospan $[n_s] \xrightarrow{s} c \xleftarrow{t} [n_t]$ in $\widehat{\mathbb{C}}$, which is the desired cospan. (Observe, again, that only ν_n has $n_s \neq n_t$.)

Similarly, full moves either are basic moves, or are essentially representable presheaves on objects in $\mathrm{ob}(\mathbb{C}_3)\backslash\mathrm{ob}(\mathbb{C}_1)$, i.e., representables on some π_n. To define the expected cospan, first observe that by the equation $ls = rs$, we obtain an morphism $[n] \xrightarrow{s} \pi_n^l \xrightarrow{l} \pi_n$, equal to rs, in $\widehat{\mathbb{C}}$. This will form the first leg of the cospan. For the other, observe that for each n and $i \in n$, we obtain, by the equations $ltd_i = lsd_i = rsd_i = rtd_i$ and by Yoneda, that the outermost part of

$$\begin{tikzcd}
n \cdot \star \arrow[r, "{[d_i]_{i\in n}}"] \arrow[d, "{[d_i]_{i\in n}}"'] & {[n]} \arrow[d] \arrow[dr, "t"] & \\
{[n]} \arrow[r] \arrow[dr, "t"'] & n|n \arrow[r, dashed, "t"] & \pi_n^r \arrow[d, "r"] \\
& \pi_n^l \arrow[r, "l"'] & \pi_n
\end{tikzcd} \tag{4}$$

commutes in $\widehat{\mathbb{C}}$, where $n \cdot \star$ denotes an n-fold coproduct of $\star$. Letting $n|n$ be the induced pushout, and the dashed morphism t be obtained by its universal property, we obtain the desired cospan $[n] \xrightarrow{ls} \pi_n \xleftarrow{t} n|n$.

Finally, closed-world moves either are full moves, or are essentially representable presheaves on some $\tau_{n,i,m,j}$. To define the expected cospan, we proceed as in Figure 2: compute the pushout $n_i|_j m$, and infer the dashed morphisms s' and t' to obtain the desired cospan $n_i|_j m \xrightarrow{s'} \tau_{n,i,m,j} \xleftarrow{t'} n_i|_j m$.

Remark 4 (Isomorphisms) *Moves are particular cospans in $\widehat{\mathbb{C}}$. For certain moves, the involved objects are representable, but not for others, like forking or synchronisation, whose final position is not representable. In the latter cases, our definition thus relies on a choice, e.g., of pushout in* (4)*. Thus, let us be completely accurate: a move is a cospan which is isomorphic to one of the cospans chosen above, in $\widehat{\mathbb{C}}^{\cdot\leftarrow\cdot\rightarrow\cdot}$, i.e., the category of functors*

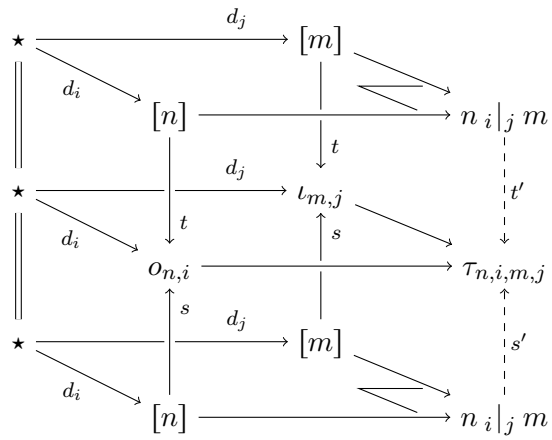


Figure 2: Construction of the synchronisation move

from the category $\cdot \leftarrow \cdot \rightarrow \cdot$ *(generated by the graph with three objects and an edge from one to each of the other two) to* $\widehat{\mathbb{C}}$*.*

3.4 Extended moves, plays, and relativisation

The most delicate part of our formalisation of Section 3 is perhaps the passage from moves to extended moves. Recall from the paragraph above Definition 1 that an extended move should be like a move occurring in a larger position.

Moves with interfaces To formalise this idea, we first equip moves with interfaces, as standard in graph rewriting [23]. Since moves are cospans, one might expect that interfaces are cospans too. This may be done, but there is a simpler, equivalent presentation. The route we follow here might have to be generalised in order to handle more complex calculi than CCS, but let us save the complications for later work.

Here, we define an *interface* for a cospan $X \rightarrow M \leftarrow Y$ to consist of a presheaf I and morphisms $X \leftarrow I \rightarrow Y$ such that

$$\begin{array}{ccc} I & \longrightarrow & Y \\ \downarrow & & \downarrow \\ X & \longrightarrow & M \end{array} \tag{5}$$

commutes, and I has dimension 0, i.e., is empty except over $\mathbb{C}_0$, i.e., consists only of channels.

Definition 5 *A cospan equipped with an interface is called a* cospan with interface*.*

Moves are particular cospans, and we now equip them with canonical interfaces: all moves except channel creation preserve the set of channels, the interface is then $n \cdot \star$, with the obvious inclusion. For example, the less obvious case is π_n: we choose

$$\begin{tikzcd} n \cdot \star \arrow[r] \arrow[d] & n|n \arrow[d] \\ {[n]} \arrow[r] & \pi_n, \end{tikzcd}$$

where the upper map is as in (4). For channel creation, we naturally choose

$$\begin{tikzcd} n \cdot \star \arrow[r, "{[d_i]_{i \in n}}"] \arrow[d] & {[n+1]} \arrow[d] \\ {[n]} \arrow[r] & \nu_n. \end{tikzcd}$$

Definition 6 *A* move with interface *is one of these cospans with interface. The basic, full, or closed-world character is retained from the underlying move.*

Extended moves We now plug moves with interfaces into contexts, in the following sense.

Definition 7 *A* context *for a cospan with interface* (5) *is a position Z, equipped with a morphism $I \to Z$.*

From any cospan with interface μ as in (5) and context $C \colon I \to Z$, we construct the cospan $C[\mu]$ as in:

$$
\begin{tikzcd}
 & Y \arrow[rr] \arrow[dd, hook] & & Y' \arrow[dd, hook, dashed] \\
 & & & \\
 & M \arrow[rr] & & M' \\
I \arrow[uur] \arrow[ur] \arrow[rr, "C"] \arrow[dr, hook] & & Z \arrow[uur] \arrow[ur] \arrow[dr, hook] & \\
 & X \arrow[rr] \arrow[uu, hook] & & X'. \arrow[uu, hook, dashed]
\end{tikzcd}
$$

I.e., we push the available morphisms out of I along C, and infer the dashed morphisms, which form the desired cospan.

Definition 8 *An* extended move *is a cospan of the shape $C[\mu]$, for any move with interface μ and context C as above.*

Example 9 *Recall that* $[2]$ *is a position with one player knowing two channels. Recall from Figure 2 the pushout*

$$
\begin{tikzcd}
\star \arrow[r, "d_1"] \arrow[d, "d_2"'] & {[2]} \arrow[d, "p_2"] \\
{[2]} \arrow[r, "p_1"'] & 2\,{}_2|_1\,2,
\end{tikzcd}
$$

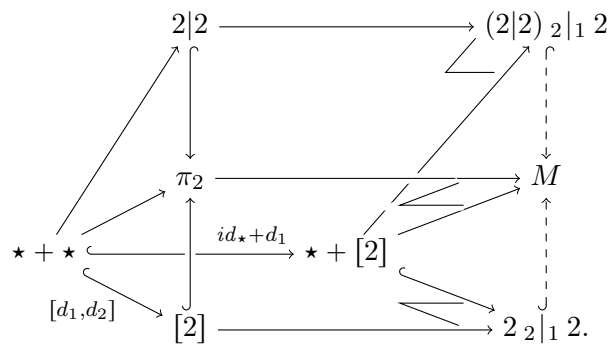
equivalently obtained as the pushout

$$
\begin{tikzcd}
\star + \star \arrow[r, "id_\star + d_1"] \arrow[d, "{[d_1,d_2]}"'] & \star + {[2]} \arrow[d, "{[a_1,p_2]}"] \\
{[2]} \arrow[r, "p_1"'] & 2\,{}_2|_1\,2.
\end{tikzcd}
$$

The base position of Figure 1 is thus $2\,{}_2|_1\,2$. *Recall also from* (4) *that* $2|2$ *denotes the position with two players both knowing two channels. Now, we have the forking move* $[2] \hookrightarrow \pi_2 \hookleftarrow 2|2$. *Equipping it with the interface*

$$[d_1, d_2]\colon\ \star + \star \to [2],$$

and putting it in the context $id_\star + d_1\colon\ \star + \star \to \star + [2]$, *(which happens to be the same as the interface), we obtain*



This formally constructs the first layer of Figure 1. Constructing the whole play would be a little too verbose to be included here, but essentially straightforward.

Plays and relativisation We may now read Definition 1 again, this time in the formal setting, to define plays. Similarly, the definition of morphisms now makes rigorous sense, as well as Proposition 1.

Proof of Proposition 1: $\mathbb{E}$ is the full subcategory of the arrow category of $\widehat{\mathbb{C}}$ whose objects are plays. $\square$

Similarly, Section 2.4 now makes rigorous sense.

4 Innocent strategies as sheaves

Now that the category of plays is defined, we move on to defining innocent strategies. There is a notion of a Grothendieck *site* [32], which consists of a category equipped with a (generalised) topology. On such sites, one may define a category of sheaves, which are very roughly the presheaves that are determined locally w.r.t. the generalised topology. We claim that there is a topology on each $\mathbb{E}_X$, for which sheaves adequately model innocent strategies. Fortunately, in our setting, sheaves admit a simple description, so that we can avoid the whole machinery. But sheaves were the way we arrived at the main ideas presented here, because they convey the right intuition: plays form a Grothendieck site, and the states of innocent strategies should be determined locally.

In this section, we first define innocent strategies, and state the spatial and temporal decomposition theorems. We then present our coalgebraic interpretation of innocent strategies, i.e., we define a polynomial endofunctor F, and show that presheaves of finite ordinals on views form a final

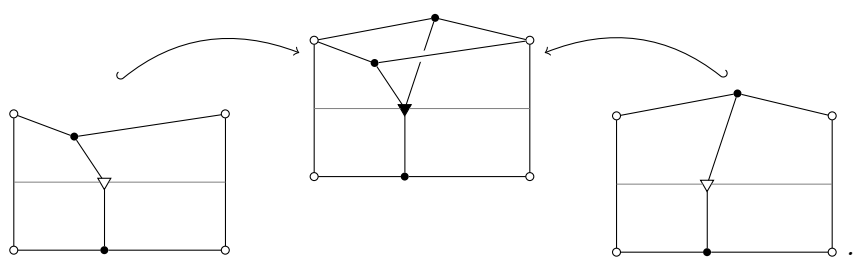
F-coalgebra. We then derive from this a formal language and its interpretation in terms of innocent strategies. We finally use this language to translate CCS with recursive equations into innocent strategies.

4.1 Innocent strategies

Definition 9 *A* view *is a finite, possibly empty 'composition'* $[n] \hookrightarrow V$ *of (extended) basic moves in* $\mathsf{Cospan}(\widehat{\mathbb{C}})$*, i.e., a play in which all the cospans are basic moves.*

When the composition is empty, we obtain $[n] \hookrightarrow [n]$, the *identity* view on $[n]$. We also note in passing that empty presheaves cannot be views, i.e., $X \hookrightarrow \varnothing$ is never a view.

Example 10 *Forking* (1) *has two non-trivial views, namely the (left legs of) basic moves* (2)*. Each of them embeds into forking:*



Example 11 *In Figure 1, the leftmost branch contains a view consisting of three basic moves: two* π_2^l *and an output.*

Definition 10 *For any position* X*, let* $\mathbb{V}_X$ *be the full subcategory of* $\mathbb{E}_X$ *consisting of views.*

More precisely, $\mathbb{V}_X$ consists of spans $U \hookleftarrow Y \to X$ where $Y \hookrightarrow U$ is a view.

Definition 11 *Let the category* S_X *of* innocent strategies *on* X *be the category* $\widehat{\mathbb{V}_X}$ *of presheaves on* $\mathbb{V}_X$*.*

A possible interpretation is that for a presheaf $F \in \widehat{\mathbb{V}_X}$ and view $V \in \mathbb{V}_X$, $F(V)$ is the set of possible *states* of the strategy F after playing V.

It might thus seem that we could content ourselves with defining only views, as opposed to plays. However, in order to define interactive equivalences in Section 5, we need to view innocent strategies as (particular) presheaves on the whole of $\mathbb{E}_X$.

The connection is as follows. Recall from MacLane [31] the notion of *right Kan extension*. Given functors F and G as on the right, a right Kan extension $\mathrm{Ran}_F(G)$ of G along F is a functor $H\colon \mathbb{D} \to \mathbb{E}$, equipped with a natural transformation $\varepsilon\colon HF \to G$, such that for all functors $K\colon \mathbb{D} \to \mathbb{E}$ and transformations $\alpha\colon KF \to G$, there is a unique $\alpha'\colon K \to H$ such that $\alpha = \varepsilon \bullet (\alpha' \circ id_F)$, where $\bullet$ is vertical composition of natural transformations. Now, precomposition with F induces a functor $\mathsf{Cat}(F,\mathbb{E})\colon \mathsf{Cat}(\mathbb{D},\mathbb{E}) \to \mathsf{Cat}(\mathbb{C},\mathbb{E})$, where $\mathsf{Cat}(\mathbb{D},\mathbb{E})$ is the category of functors $\mathbb{D} \to \mathbb{E}$ and natural transformations between them. When $\mathbb{E}$ is complete, right Kan extensions always exist (and an explicit formula for our setting is given below), and choosing one of them for each functor $\mathbb{C} \to \mathbb{E}$ induces a right adjoint to $\mathsf{Cat}(F,\mathbb{E})$. Furthermore, it is known that when F is full and faithful, then ε is a natural isomorphism, i.e., $HF \cong G$.

$$\begin{tikzcd}
\mathbb{C} \arrow[rr, "F"] \arrow[dr, bend right, "G"'] & & \mathbb{D} \arrow[dl, bend left, "H"] \arrow[dl, bend right, "K"'] \\
& \mathbb{E} \arrow[ul, Rightarrow, "\varepsilon"] \arrow[ul, Rightarrow, "\alpha"'] & \arrow[u, Rightarrow, "\alpha'"']
\end{tikzcd}$$

Proposition 2 *If F is full and faithful, then Ran_F is a full essential embedding.*

Proof: First, let us show that Ran_F is essentially injective on objects. Indeed, assume $H = \mathrm{Ran}_F(G)$, $\mathrm{Ran}_F(G') = H'$, and $i\colon H \to H'$ is an isomorphism with inverse k. We must construct an isomorphism $G \cong G'$. Let $j\colon G \to G'$ be $\varepsilon_{G'} \bullet (iF) \bullet \varepsilon_G^{-1}$. Similarly, let $l\colon G' \to G$ be $\varepsilon_G \bullet (kF) \bullet \varepsilon_{G'}^{-1}$. We have

$$\begin{aligned}
l \bullet j &= \varepsilon_G \bullet (kF) \bullet \varepsilon_{G'}^{-1} \bullet \varepsilon_{G'} \bullet (iF) \bullet \varepsilon_G^{-1} \\
&= \varepsilon_G \bullet (kF) \bullet (iF) \bullet \varepsilon_G^{-1} \\
&= \varepsilon_G \bullet ((k \bullet i) \circ F) \bullet \varepsilon_G^{-1} \\
&= \varepsilon_G \bullet \varepsilon_G^{-1} \\
&= id_G.
\end{aligned}$$

Similarly, $j \bullet l = id_{G'}$ and we have $G \cong G'$.

To see that Ran_F is full, observe that for any $i\colon H \to H'$, with $H = \mathrm{Ran}_F(G)$ and $H' = \mathrm{Ran}_F(G')$, $j = \varepsilon_{G'} \bullet (iF) \bullet \varepsilon_G^{-1}$ is an antecedent of i by Ran_F. Indeed, by definition, $\mathrm{Ran}_F(j)$ is the unique $i'\colon H \to H'$ such that $\varepsilon_{G'} \bullet (i'F) = j \bullet \varepsilon_G$. But the latter is equal to $\varepsilon_{G'} \bullet (iF)$, so $i' = i$.

Finally, to show that Ran_F is faithful, consider $G, G'\colon \mathbb{C} \to \mathbb{E}$ and two natural transformations $i, j\colon G \to G'$ such that $\mathrm{Ran}_F(i) = \mathrm{Ran}_F(j) = k$. Then, by construction of k, we have

$$i \bullet \varepsilon_G = \varepsilon_{G'} \bullet (kF) = j \bullet \varepsilon_G.$$

But, ε_G being an isomorphism, this implies $i = j$ as desired. $\square$

Returning to views and plays, the embedding $i_X : \mathbb{V}_X \hookrightarrow \mathbb{E}_X$ is full, so right Kan extension along $i_X^{op} : \mathbb{V}_X^{op} \to \mathbb{E}_X^{op}$ induces a full essential embedding $\mathrm{Ran}_{i_X^{op}} : \widehat{\mathbb{V}_X} \to \widehat{\mathbb{E}_X}$. The (co)restriction of this essential embedding to its essential image thus yields an essentially surjective, fully faithful functor, i.e., an equivalence of categories:

Proposition 3 *The category S_X is equivalent to the essential image of $\mathrm{Ran}_{i_X^{op}}$.*

The standard characterisation of right Kan extensions as ends [31] yields, for any $F \in \widehat{\mathbb{V}_X}$ and $U \in \mathbb{E}_X$:

$$\mathrm{Ran}_{i_X^{op}}(F)(U) = \int_{V \in \mathbb{V}_X} F(V)^{\mathbb{E}_X(V,U)},$$

i.e., giving an element of $\mathrm{Ran}_{i_X^{op}}(F)$ on a play U amounts to giving, for each view V and morphism $V \to U$, an element of $F(V)$, satisfying some compatibility conditions. In Example 12 below, we compute an example right Kan extension.

The interpretation of strategies in terms of states extends: for any presheaf $F \in \widehat{\mathbb{E}_X}$ and play $U \in \mathbb{E}_X$, $F(U)$ is the set of possible *states* of the strategy F after playing U. That F is in the image of $\mathrm{Ran}_{i_X^{op}}$ amounts to $F(U)$ being a compatible tuple of states of F after playing each view of U.

Example 12 *Here is an example of a presheaf $F \in \widehat{\mathbb{E}_X}$ which is not innocent, i.e., not in the image of $\mathrm{Ran}_{i_X^{op}}$. Consider the position X consisting of three players, say x, y, z, sharing a channel, say a. Let X_x be the subposition with only x and a, and similarly for X_y, X_z, $X_{x,y}$, and $X_{x,z}$. Let $I_x = (\iota_{1,1} \hookleftarrow X_x \hookrightarrow X)$ be the play where x inputs on a, and similarly let O_y and O_z be the plays where y and z output on a, respectively. Let now $S_{x,y} = (\tau_{1,1,1,1} \hookleftarrow X_{x,y} \hookrightarrow X)$ be the play where x and y synchronise on a (x inputs and y outputs), and similarly let $S_{x,z}$ be the play where x and z synchronise on a.*

Finally, we define a presheaf F on $\mathbb{E}/X$ such that $F(S_{x,y}) = 2$ is a two-element set, and $F(S_{x,z}) = \varnothing$. To define F on other plays, the idea is to map any strict subplay of $S_{x,y}$ and $S_{x,z}$ to a one-element set 1, and other plays to $\varnothing$. The cleanest technical way to do this seems to be as follows. The poset Ex_{11} defined by

$$\begin{tikzcd}
O_y \arrow[dr] & & I_x \arrow[dl] \arrow[dr] & & O_z \arrow[dl] \\
& S_{x,y} & & S_{x,z} &
\end{tikzcd}$$

fully embeds into $\mathbb{E}/X$*, via, say* i_{11}*. Let* F_0 *be the presheaf on* Ex_{11} *defined by:*

$$\begin{tikzcd}
1 & & 1 & & 1 \\
& 2 \arrow[ul] \arrow[ur] & & \varnothing. \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

We now let $F = \mathrm{Ran}_{i_{11}^{op}}(F_0)$*. Because* i_{11} *is fully faithful,* F *coincides with* F_0 *on the plays of* Ex_{11}*, as desired.*

Now, F *fails to be innocent on two counts. First, since* x *and* y *accept to input and output in only one way, it is non-innocent to accept that they synchronise in more than one way. Formally,* $S_{x,y}$ *has two non-trivial views,* I_x *and* O_y*, so since* F *maps identity views to a singleton,* $F(S_{x,y})$ *should be isomorphic to* $F(I_x) \times F(O_y) = 1 \times 1 = 1$*. The second reason why* F *is not innocent is that, since* x *and* z *accept to input and output,* F *should accept that they synchronise. Formally,* $F(S_{x,z})$ *should also be a singleton. This altogether models the fact that in CCS, processes do not get to know with which other processes they synchronise.*

The restriction of F *to* $\mathbb{V}_X$*, i.e.,* $F' = F \circ i_X^{op}$*, in turn has a right Kan extension* F''*, which is innocent. (In passing, the unit of the adjunction* $\mathsf{Cat}(i_X^{op}, \mathsf{Set}) \dashv \mathrm{Ran}_{i_X^{op}}$ *is a natural transformation* $F \to F''$*.) To conclude this example, let us compute* F''*. First,* F' *only retains from* F *its values on views. So, if* X_x *denotes the empty view on* X_x*,* $F'(X_x) = 1$*, and similarly* $F'(X_y) = F'(X_z) = 1$*. Furthermore,* $F'(I_x) = F'(O_y) = F'(O_z) = 1$*. Finally, for any view* V *not isomorphic to any of the previous ones,* $F'(V) = \varnothing$*. So, recall that* F'' *maps any play* $U \hookleftarrow Y \hookrightarrow X$ *to* $\int_{V \in \mathbb{V}_X} F'(V)^{\mathbb{E}_X(V,U)}$*. So, e.g., since the views of* $S_{x,y}$ *are subviews of* I_x *and* O_y*, we have* $F''(S_{x,y}) = F'(I_x) \times F'(O_y) = 1$*. Similarly,* $F''(S_{x,z}) = 1$*. But also, for any play* U *such that all views* $V \to U$ *are subviews of either of* I_x*,* O_y*, or* O_z*, we have* $F''(U) = 1$*. Finally, for any play* U *such that there exists a view* $V \to U$ *which is not a subview of any of* I_x*,* O_y*, or* O_z*, we have* $F''(U) = \varnothing$*.*

One way to understand Proposition 3 is to view $\widehat{\mathbb{V}_X}$ as the syntax for innocent strategies: presheaves on views are (almost) infinite terms in a cer-

tain syntax (see Section 4.4 below). On the other hand, seeing them as presheaves on plays will allow us to consider their global behaviour: see Section 5 when we restrict to the closed-world game. Thus, right Kan extension followed by restriction to closed-world will associate a semantics to innocent strategies.

Remark 5 *The relevant Grothendieck topology on $\mathbb{E}_X$ says, roughly, that a play is covered by its views. Any sheaf for this topology is determined by its restriction to $\mathbb{V}_X$, for its elements on any non-view play U are precisely amalgamations of its elements on views of U. Right Kan extension just computes these amalgamations in the particular case of a topology derived from a full subcategory, here views.*

So, we have defined for each X the category S_X of innocent strategies on X. This assignment is actually functorial $\mathbb{B}^{op} \to \mathsf{CAT}$, as follows (where CAT is the large category of locally small categories). Any morphism $f : Y \to X$ induces a functor $f_! : \mathbb{V}_Y \to \mathbb{V}_X$ mapping $(V \hookleftarrow Z \to Y)$ to $(V \hookleftarrow Z \to Y \to X)$. Precomposition with $(f_!)^{op}$ thus induces a functor $\mathsf{S}_f : \widehat{\mathbb{V}_X} \to \widehat{\mathbb{V}_Y}$.

Proposition 4 *This defines a functor $\mathsf{S} : \mathbb{B}^{op} \to \mathsf{CAT}$.*

Proof: A straightforward verification. $\square$

But there is more: for any position, giving a strategy for each player in it easily yields a strategy on the whole position. We call this *amalgamation* of innocent strategies (because the functor S is indeed a *stack* [43], and this is a particular case of amalgamation in that stack). Formally, consider any subpositions X_1 and X_2 of a given position X, inducing a partition of the players of X, i.e., such that $X_1 \cup X_2$ contains all players of X, and $X_1 \cap X_2$ contains none. Then $\mathbb{V}_X$ is isomorphic to the coproduct $\mathbb{V}_{X_1} + \mathbb{V}_{X_2}$. (Indeed, a view contains in particular an initial player in X, which forces it to belong either in $\mathbb{V}_{X_1}$ or in $\mathbb{V}_{X_2}$.)

Definition 12 *Given innocent strategies F_1 on X_1 and F_2 on X_2, let their* amalgamation *be their copairing*

$$[F_1, F_2] : \mathbb{V}_X^{op} \cong (\mathbb{V}_{X_1} + \mathbb{V}_{X_2})^{op} \cong \mathbb{V}_{X_1}^{op} + \mathbb{V}_{X_2}^{op} \to \mathsf{Set}.$$

By universal property of coproduct:

Proposition 5 *Amalgamation yields an isomorphism of categories*

$$\widehat{\mathbb{V}_X} \cong \widehat{\mathbb{V}_{X_1}} \times \widehat{\mathbb{V}_{X_2}}.$$

Example 13 *Consider again the position X from Example 12, and let $X_{y,z}$ be the subposition with only y and z. We have $\mathbb{V}_X \simeq (\mathbb{V}_{X_x} + \mathbb{V}_{X_{y,z}})$, which we may explain by hand as follows. A view on X has a base player, x, y, or z, and so belongs either in $\mathbb{V}_{X_x}$ or in $\mathbb{V}_{X_{y,z}}$. Furthermore, if V is a view on x and W is a view on y, then $\mathbb{V}_X(V, W) = \varnothing$ (and similarly for any pair of distinct players in X).*

Now, recall F', the restriction of F to $\mathbb{V}_X$. We may define $F_x \colon \mathbb{V}_{X_x}^{op} \to \mathsf{Set}$ to be the restriction of F' along the (opposite of the) embedding $\mathbb{V}_{X_x} \hookrightarrow \mathbb{V}_X$, and similarly $F_{y,z}$ to be the restriction of F' along $\mathbb{V}_{X_{y,z}} \hookrightarrow \mathbb{V}_X$. We have obviously $F' = [F_x, F_{y,z}]$.

Analogous reasoning leads to what we call spatial decomposition. For any X, let $\mathrm{Pl}(X) = \sum_n X([n])$, i.e., the set of pairs (n, x), where x is a player in X, knowing n channels.

Theorem 1 *We have $\widehat{\mathbb{V}_X} \cong \prod_{(n,x)\in \mathrm{Pl}(X)} \widehat{\mathbb{V}_{[n]}}$.*

Again, this is a particular case of amalgamation in the stack S, but we do not need to spell out the definition here.

4.2 Temporal decomposition

Let us now describe *temporal* decomposition. Recall that basic moves are left and right half-forking (2), input, output, tick, and channel creation.

Definition 13 *Let $\mathcal{M}$ be the graph with vertices all natural numbers n, and with edges $n \to n'$ all (isomorphism classes of) basic moves $M \colon [n] \to [n']$.*

Recall from Remark 4 that the notion of isomorphism considered here is that of an isomorphism of cospans in $\widehat{\mathbb{C}}$.

Definition 14 *Let $\mathcal{M}_n$ be the set of edges from n in $\mathcal{M}$.*

For stating the temporal decomposition theorem, we need a standard [21] categorical construction, the category of families on a given category $\mathbb{C}$. First, given a set X, consider the category $\mathrm{Fam}(X)$ with as objects X-indexed families of sets $Y = (Y_x)_{x\in X}$, and as morphisms $Y \to Z$ families

$(f_x\colon Y_x \to Z_x)_{x\in X}$ of maps. This category is equivalently described as the slice category Set/X. To see the correspondence, consider any family $(Y_x)_{x\in X}$, and map it to the projection function $\sum_{x\in X} Y_x \to X$ sending (x,y) to x. Conversely, given $f\colon Y \to X$, let, for any $x \in X$, Y_x be the fibre of f over x, i.e., $f^{-1}(x)$.

Generalising from sets X to small categories $\mathbb{C}$, $\mathrm{Fam}(\mathbb{C})$ has as objects families $p\colon Y \to \mathrm{ob}(\mathbb{C})$ indexed by the objects of $\mathbb{C}$. Morphisms $(Y,p) \to (Z,q)$ are pairs of $u\colon Y \to Z$ and $v\colon Y \to \mathrm{mor}(\mathbb{C})$, where $\mathrm{mor}(\mathbb{C})$ is the set of morphisms of $\mathbb{C}$, such that $\mathrm{dom} \circ v = p$, and $\mathrm{cod} \circ v = q \circ u$. Thus, any element $y \in Y$ over $C \in \mathbb{C}$ is mapped to some $u(y) \in Z$ over $C' \in \mathbb{C}$, and this mapping is labelled by a morphism $v(y)\colon C \to C'$ in $\mathbb{C}$. The obtained category is locally small.

Further generalising, for $\mathbb{C}$ a locally small category, we may define $\mathrm{Fam}(\mathbb{C})$ in exactly the same way (with Y still a set), and the obtained category remains locally small.

The temporal decomposition theorem is:

Theorem 2 *There is an equivalence of categories*

$$\mathsf{S}_n \simeq \mathrm{Fam}\left(\prod_{M\in\mathcal{M}_n} \mathsf{S}_{\mathrm{cod}(M)}\right).$$

The main intuition is that an innocent strategy is determined up to isomorphism by (i) its initial states, and (ii) what remains of them after each possible basic move. The family construction is what permits innocent strategies with several possible states over the identity play.

Proof sketch: For general reasons, we have:

$$\begin{array}{rcl} \mathrm{Fam}\left(\prod_{M\in\mathcal{M}_n} \mathsf{S}_{\mathrm{cod}(M)}\right) & = & \mathrm{Fam}\left(\prod_{M\in\mathcal{M}_n} [\mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}]\right) \\ & \cong & \mathrm{Fam}\left(\left[\sum_{M\in\mathcal{M}_n} \mathbb{V}_{\mathrm{cod}(M)}{}^{op}, \mathsf{Set}\right]\right) \\ & \simeq & \left[\sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}\right] \downarrow \Delta, \end{array}$$

where $\Delta\colon \mathsf{Set} \to [\sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}]$ maps any set X to the constant presheaf mapping any object to X and any morphism to the identity.

By definition, the last category is a lax pullback

$$\begin{tikzcd} \left[\sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}\right] \arrow[r, equal] & \left[\sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}\right] \\ \mathsf{Set} \arrow[u, "\Delta"] & \left[\sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}\right] \downarrow \Delta \arrow[l] \arrow[u] \arrow[ul, Rightarrow] \end{tikzcd}$$

in CAT.

Now, any basic move $M\colon n \to n'$ induces a functor $(- \circ M)\colon \mathbb{V}_{[n']} \to \mathbb{V}_{[n]}$, mapping any view $V \in \mathbb{V}_{[n']}$ to $V \circ M$ (with composition in $\mathsf{Cospan}(\widehat{\mathbb{C}})$). We show that the square

$$\begin{tikzcd} \sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)} \arrow[r, equal] \arrow[d, "!"'] & \sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)} \arrow[d, "{[-\circ M]_{M\in\mathcal{M}_n}}"] \arrow[dl, phantom, "\lambda"] \\ 1 \arrow[r, "\ulcorner id_{[n]} \urcorner"'] & \mathbb{V}^{op}_{[n]} \end{tikzcd} \tag{6}$$

is a lax pushout in Cat, where $\lambda_{M,V}\colon id_{[n]} \to M \circ V$, seen in $\mathbb{V}_{[n]}$, is the obvious inclusion, which for general reasons is mapped by the hom-2-functor $\mathsf{CAT}(-, \mathsf{Set})$ to a lax pullback. But $\mathsf{CAT}(!, \mathsf{Set}) = \Delta$ and $\mathsf{CAT}(id, \mathsf{Set}) = id$, so we obtain a canonical isomorphism of lax pullbacks

$$\mathsf{S}_{[n]} = [\mathbb{V}^{op}_{[n]}, \mathsf{Set}] \cong \left[\sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set} \right] \downarrow \Delta.$$

More detail is in Appendix A. □

Remark 6 *The theorem almost makes innocent strategies into a* sketch *(on the category with positions as objects, finite compositions of extended moves as morphisms, and the $\mathcal{M}_X$'s as distinguished cones). Briefly, being a sketch would require a bijection of sets $\mathsf{S}_n \cong \prod_{M\in\mathcal{M}_n} \mathsf{S}_{\mathrm{cod}(M)}$. Here, the bijection becomes an equivalence of categories, and the family construction sneaks in.*

4.3 Innocent strategies as a terminal coalgebra

Temporal decomposition gives

$$\mathsf{S}_n \simeq \mathrm{Fam}\left(\prod_{M\in\mathcal{M}_n} \mathsf{S}_{\mathrm{cod}(M)} \right),$$

for all n. Considering a variant of this formula as a system of equations will lead to our interpretation of CCS. The first step is to replace Set with FinOrd, the category of finite ordinals and monotone functions. The proof applies *mutatis mutandis* and we obtain an equivalence, which, because both categories are skeletal, is an isomorphism:

$$\widehat{\mathbb{V}_{[n]}} \cong \mathrm{Fam}_f\left(\prod_{M\in\mathcal{M}_n} \widehat{\mathbb{V}_{\mathrm{cod}(M)}} \right), \tag{7}$$

where

- Fam_f is the same as Fam but with finite families, i.e., for any category $\mathbb{C}$, $\mathrm{ob}(\mathrm{Fam}_f(\mathbb{C})) = \sum_{I \in \mathsf{FinOrd}} (\mathrm{ob}(\mathbb{C}))^I = (\mathrm{ob}(\mathbb{C}))^*$ is the set of finite words over objects of $\mathbb{C}$, also known as the free monoid on $\mathrm{ob}(\mathbb{C})$;

- and for any category $\mathbb{C}$, $\widehat{\mathbb{C}}$ denotes the functor category $[\mathbb{C}^{op}, \mathsf{FinOrd}]$.

Remark 7 *Recall that in the proof of Theorem 2,* Fam *arises from the 'constant presheaf' functor* $\Delta \colon \mathsf{Set} \to \widehat{-}$*, with* $-$ *a complicated category. This functor itself is equal to restriction along* $- \to 1$*, via* $\widehat{1} \cong \mathsf{Set}$*. Replacing* Set *with* FinOrd *thus replaces* Δ *with the analogous functor* $\mathsf{FinOrd} \to \widehat{-}$*, via* $\widehat{1} \cong \mathsf{FinOrd}$*, and thus* Fam *with* Fam_f*.*

Furthermore, because FinOrd *embeds into* Set*, the special strategies of* $\widehat{\mathbb{V}_{[n]}}$ *embed into* $\mathsf{S}_{[n]}$*.*

Then, taking advantage of the fact that FinOrd is a small category, we consider its set FinOrd_0 of objects, i.e., finite ordinals, and the endofunctor F on $\mathsf{Set}/\mathsf{FinOrd}_0$ defined on any family of sets $X = (X_i)_{i \in \mathsf{FinOrd}_0}$ by:

$$(\mathsf{F}(X))_n = \sum_{I \in \mathsf{FinOrd}_0} \left(\prod_{M \in \mathcal{M}_n} X_{\mathrm{cod}(M)} \right)^I ,$$

where we abusively confuse $[n'] = \mathrm{cod}(M)$ and the natural number n' itself. The isomorphism (7) becomes

$$\mathrm{ob}(\widehat{\mathbb{V}_{[n]}}) \cong (\mathsf{F}(\mathrm{ob}(\widehat{\mathbb{V}_{-}})))_n .$$

We may decompose F as follows. Consider the endofunctor on $\mathsf{Set}/\mathsf{FinOrd}_0$ defined by $(\partial X)_n = \prod_{M \in \mathcal{M}_n} X_{\mathrm{cod}(M)}$, for any family X. We obviously have:

Lemma 1 F *is equal to the composite* $(\partial -)^*$*.*

This endofunctor is polynomial [27] and we now give a characterisation of its final coalgebra. The rest of this subsection is devoted to proving:

Theorem 3 *The family* $\mathrm{ob}(\widehat{\mathbb{V}_n})$ *formed for each* n *by (the objects of)* $\widehat{\mathbb{V}_n}$ *is a terminal coalgebra for* F*.*

Consider any F-coalgebra $a\colon X \to \mathsf{F}X$.

We define by induction on N a sequence of maps $f_N\colon X \to \widehat{\mathbb{V}_{[-]}}$, such that for any view V of length less than N (i.e., with less than N basic moves), and any $N' > N$, $f_{N'}(x)(V) = f_N(x)(V)$, and similarly the action of $f_N(x)$ on morphisms is the same as that of $f_{N'}(x)$.

To start the induction, take $f_0(x)$ to be the strategy mapping $id_{[n]}$ to $\pi(a(x))$, i.e., the length of $a(x) \in \sum_{I \in \mathsf{FinOrd}_0}((\partial X)_n)^I$, and all other views to 0.

Furthermore, given f_N, define f_{N+1} to be

$$X \xrightarrow{a} \mathsf{F}X \xrightarrow{\mathsf{F}(f_N)} \mathsf{F}(\widehat{\mathbb{V}_{[-]}}) \xrightarrow{\cong} \widehat{\mathbb{V}_{[-]}},$$

where the equivalence is by temporal decomposition.

Unfolding the definitions yields:

Lemma 2 *Consider any $x \in X_n$, and $a(x) = (z_1, \ldots, z_k)$. For any move $M\colon n \to n'$ and view $V\colon n' \to n''$ of length at most N, and for any $i \in k$, $f_{N+1}(x)(V \circ M) = \sum_{i \in k} f_N(z_i(M))(V)$.*

For any $x \in X_n$, we have a sequence $f_0(x) \hookrightarrow f_1(x) \hookrightarrow \ldots f_N(x) \hookrightarrow f_{N+1}(x) \hookrightarrow \ldots$ which is pointwise stationary. This sequence thus has a colimit in $\widehat{\mathbb{V}_{[n]}}$, the presheaf mapping any view V of length N to $f_N(V)$ (or equivalently $f_{N'}(V)$ for any $N' \geqslant N$), which allows us to define:

Definition 15 *Let $f\colon X \to \widehat{\mathbb{V}_{[-]}}$ map any $x \in X_n$ to $\bigcup_N f_N(x)$.*

By construction, we have

Lemma 3 *The following diagram commutes:*

$$\begin{tikzcd} X \arrow[r, "a"] \arrow[d, "f"'] & \mathsf{F}X \arrow[d, "\mathsf{F}(f)"] \\ \widehat{\mathbb{V}_{[-]}} & \mathsf{F}(\widehat{\mathbb{V}_{[-]}}). \arrow[l, "\cong"'] \end{tikzcd}$$

Lemma 4 *The map f is a morphism of F-coalgebras.*

Proof: Let, for any innocent strategy $S \in \widehat{\mathbb{V}_{[n]}}$ and $i \in S(id_{[n]})$, $S_{|i}$ be the strategy mapping any view V to the fibre over i of $S(V) \to S(id_{[n]})$. Using the notations of Lemma 2, we must show that for any $i \in k$, we have $(f(x))_{|i}(V \circ M) = f(z_i(M))(V)$. But Lemma 2 entails that $f(x)(V \circ M) \to f(x)(id_{[n]})$ is actually the coproduct over $i' \in k$ of all $f(z_{i'}(M))(V) \xrightarrow{!} 1 \xrightarrow{i'} \pi(a(x))$, so its fibre over i is indeed $f(z_i(M))(V)$. $\square$

Lemma 5 *The map f is the unique map $X \to \widehat{\mathbb{V}_{[-]}}$ of F-coalgebras.*

Proof: Consider any such map g of coalgebras. It must be such that $g(x)(id_{[n]}) = \pi(a(x))$, and furthermore, using the same notation as before, for any $i \in k$ $(g(x))_{|i}(V \circ M) = g(z_i(M))(V)$, which imposes by induction that $f = g$. $\square$

The last two lemmas directly entail Theorem 3.

4.4 Languages

A consequence of Theorem 3 is that the family $\widehat{\mathbb{V}_n}$ supports the operations of the grammar

$$\frac{\ldots\ n \vdash F_i\ \ldots\ (\forall i \in I)}{n \vdash \sum_{i \in I} F_i}\ (I \in \mathsf{FinOrd}_0)$$

$$\frac{\ldots\ n' \vdash F_M\ \ldots\ (\forall M\colon [n] \to [n'] \in \mathcal{M})}{n \vdash \langle M \mapsto F_M \rangle}\ .$$

Here, $n \vdash F$ denotes a presheaf of finite ordinals on $\mathbb{V}_n$. The interpretation is as follows: given presheaves $F_1, \ldots, F_I$, for $I \in \mathsf{FinOrd}_0$, the first rule constructs the finite coproduct $\sum_{i \in I} F_i$ of presheaves (finite coproducts exist in $\widehat{\mathbb{V}_n}$ because they do in FinOrd). In particular, when I is the empty ordinal, we sum over an empty set, so the rule degenerates to

$$\frac{}{n \vdash \varnothing}\ .$$

In terms of presheaves, this is just the constantly empty presheaf.

For the second rule, if for all basic $M\colon [n] \to [n']$, we are given $F_M \in \widehat{\mathbb{V}_{[n']}}$, then $\langle M \mapsto F_M \rangle$ denotes the image under (7) of

$$(1, 1 \mapsto M \mapsto F_M).$$

Here, we provide an element of the right-hand side of (7), consisting of the finite ordinal $I = 1 = \{1\}$, and the function mapping M to $F_M \in \widehat{\mathbb{V}_{[n']}}$ (up to currying). That was for parsing; the intuition is that we construct a presheaf with one initial state, 1, which maps any view starting with M, say $V \circ M$, to $F_M(V)$. Thus the F_M's specify what remains of our presheaf after each possible basic move. In particular, when all the F_M's are empty, we obtain a presheaf which has an initial state, but which does nothing beyond it. We abbreviate it as $0 = \langle _ \mapsto \varnothing \rangle$.

$$\textsc{CCSApp}\quad \frac{}{\Xi;\Gamma \vdash x(a_1,\ldots,a_n)}\ ((x\colon n)\in\Xi \text{ and } a_1,\ldots,a_n\in\Gamma)$$

$$\frac{\Xi;\Gamma,a\vdash P}{\Xi;\Gamma\vdash \nu a.P}\ (a\notin\Gamma) \qquad \frac{\Xi;\Gamma\vdash P \qquad \Xi;\Gamma\vdash Q}{\Xi;\Gamma\vdash P|Q}$$

$$\frac{\ldots \qquad \Xi;\Gamma\vdash P_i \qquad \ldots \qquad (\forall i\in I)}{\Xi;\Gamma\vdash \sum_{i\in I}\alpha_i.P_i}\ (I\in\mathsf{FinOrd}_0 \text{ and } \forall i\in I, \lfloor\alpha_i\rfloor\in\Gamma)$$

$$\textsc{Global}\quad \frac{\Xi;\Delta_1\vdash P_1 \qquad \ldots \qquad \Xi;\Delta_n\vdash P_n \qquad \Xi;\Gamma\vdash P}{\Gamma\vdash \texttt{rec}\ x_1(\Delta_1)\ \texttt{:=}\ P_1,\ldots,x_n(\Delta_n)\ \texttt{:=}\ P_n\ \texttt{in}\ P}$$

Figure 3: CCS syntax

4.5 Translating CCS

It is rather easy to translate CCS into this language. First, define CCS syntax by the natural deduction rules in Figure 3, where Names and Vars are two fixed, disjoint, and infinite sets of *names* and *variables*; Ξ ranges over finite sequences of pairs $(x\colon n)$ of a variable x and its *arity* $n\in\mathsf{FinOrd}_0$, such that the variables are pairwise distinct; Γ ranges over finite sequences of pairwise distinct names; there are two judgements: $\Gamma\vdash P$ for *global* processes, $\Xi;\Gamma\vdash P$ for *open* processes. Rule Global is the only rule for forming global processes, and there $\Xi=(x_1\colon|\Delta_1|,\ldots,x_n\colon|\Delta_n|)$. Finally, α denotes a or $\overline{a}$, for $a\in\mathsf{Names}$, and $\lfloor a\rfloor=\lfloor\overline{a}\rfloor=a$.

First, we define the following (approximation of a) translation on open processes, mapping each open process $\Xi;\Gamma\vdash P$ to $[\![P]\!]\in\widehat{\mathbb{V}_n}$, for $n=|\Gamma|$. This translation ignores the recursive definitions, and we will refine it below to take them into account. We proceed by induction on P, leaving contexts $\Xi;\Gamma$ implicit:

$$\begin{aligned} x(a_1,\ldots,a_k) &\mapsto \varnothing \\ P|Q &\mapsto \langle\ \pi^l_n \mapsto [\![P]\!],\ \pi^r_n \mapsto [\![Q]\!],\ _\mapsto\varnothing\ \rangle \end{aligned} \qquad \begin{aligned} \nu a.P &\mapsto \langle \nu_n\mapsto [\![P]\!], _\mapsto\varnothing\rangle \\ \textstyle\sum_{i\in I}\alpha_i.P_i &\mapsto \langle\ (o_{n,j}\mapsto \textstyle\sum_{k\in I_j}[\![P_k]\!],\ \iota_{n,j}\mapsto \textstyle\sum_{k\in I_j}[\![P_k]\!]\ )_{j\in n},\ _\mapsto\varnothing\ \rangle. \end{aligned}$$

Let us explain intuitions and notation. In the first case, we assume implicitly that $(x\colon k) \in \Xi$; the intuition is just that we approximate variables with empty strategies. Next, $P|Q$ is translated to the strategy with one initial state, which only accepts left and right half-forking first, and then lets its avatars play $[\![P]\!]$ and $[\![Q]\!]$, respectively. Similarly, $\nu a.P$ is translated to the strategy with one initial state, accepting only the channel creation move, and then playing $[\![P]\!]$. In the last case, the guarded sum $\sum_{i\in I} \alpha_i.P_i$ is translated to the strategy with one initial state, which

- accepts input on any channel a when $\alpha_i = a$ for some $i \in I$, and output on any channel a when $\alpha_i = \overline{a}$ for some $i \in I$;
- after an input on a, plays the sum of all $[\![P_i]\!]$'s such that $\alpha_i = a$; and after an output on a, plays the sum of all $[\![P_i]\!]$'s such that $\alpha_i = \overline{a}$.

Formally, in the definition, we let, for all $j \in n$, $I_{\overline{j}} = \{i \in I \mid \alpha_i = \overline{a_j}\}$ and $I_j = \{i \in I \mid \alpha_i = a_j\}$. In particular, if $I = \varnothing$, we obtain 0.

Thus, almost all translations of open processes have exactly one initial state, i.e., map the identity view on $[n]$ to the singleton 1. The only exceptions are variable applications, which are mapped to the empty presheaf.

The translation extends to global processes as follows. Fixing a global process $Q = (\texttt{rec}\ x_1(\Delta_1)\ \texttt{:=}\ P_1, \ldots, x_k(\Delta_k)\ \texttt{:=}\ P_k\ \texttt{in}\ P)$ typed in Γ with n names, define the sequence $(P^i)_{i\in\mathsf{FinOrd}_0}$ of open processes (all typed in $\Xi;\Gamma$) as follows. First, $P^0 = P$. Then, let $P^{i+1} = \mathsf{d}P^i$, where d is the *derivation* endomap on open processes typed in any extension $\Xi;(\Gamma,\Delta)$ of $\Xi;\Gamma$, which unfolds one layer of recursive definitions. This map is defined by induction on its argument as follows:

$$\begin{aligned} \mathsf{d}(x_l(a_1,\ldots,a_{k_l})) &= P_l[b_j \mapsto a_j]_{1\leqslant j\leqslant k_l} & \mathsf{d}(\nu a.P) &= \nu a.\mathsf{d}P \\ \mathsf{d}(P|Q) &= \mathsf{d}P|\mathsf{d}Q & \mathsf{d}(\textstyle\sum_{i\in I}\alpha_i.P_i) &= \textstyle\sum_{i\in I}\alpha_i.(\mathsf{d}P_i), \end{aligned}$$

where for all $l \in \{1,\ldots,k\}$, $\Delta_l = (b_1,\ldots,b_{k_l})$, and $P[\sigma]$ denotes simultaneous, capture-avoiding substitution of names in P by σ.

By construction, the translations of these open processes form a sequence $[\![P^0]\!] \hookrightarrow [\![P^1]\!] \ldots$ of inclusions in $\widehat{\mathbb{V}_n}$, such that for any natural number i and view $V \in \mathbb{V}_n$ of length i, $[\![P^j]\!](V)$ is fixed after $j = (k+1)i$, at worst, i.e., for all $j \geqslant (k+1)i$, $[\![P^j]\!](V) = [\![P^{(k+1)i}]\!](V)$. Thus, this sequence has a colimit in $\widehat{\mathbb{V}_n}$, the presheaf sending any view V of length i to $[\![P^{(k+1)i}]\!](V)$. We put:

Definition 16 *Let the translation of Q be $[\![Q]\!] = \mathrm{colim}_{i\in\mathsf{FinOrd}}[\![P^i]\!]$.*

Which equivalence is induced by this mapping on CCS, especially when taking into account the interactive equivalences developed in the next section? This is the main question we will try to address in future work.

5 Interactive equivalences

5.1 Fair testing vs. must testing: the standard case

An important part of concurrency theory consists in studying *behavioural equivalences*. Since each such equivalence is supposed to define when two processes behave the same, it might seem paradoxical to consider several of them. Van Glabbeek [42] argues that each behavioural equivalence corresponds to a physical scenario for observing processes.

A distinction we wish to make here is between *fair* scenarios, and *potentially unfair* ones. An example of a fair scenario is when parallel composition of processes is thought of as modelling different physical agents, e.g., in a game with several players. Otherwise said, players are really independent. On the other hand, an example of a potentially unfair scenario is when parallelism is implemented via a scheduler.

This has consequences on so-called *testing* equivalences [7]. Let $\heartsuit$ be a fixed action.

Definition 17 *A process P is* must orthogonal *to a context C, notation $P \perp^m C$, when all maximal traces of $C[P]$ play $\heartsuit$ at some point.*

Here, maximal means either infinite or finite without extensions. Let $P^{\perp^m}$ be the set of all contexts must orthogonal to P.

Definition 18 *P and Q are* must equivalent*, notation $P \sim_m Q$, when $P^{\perp^m} = Q^{\perp^m}$.*

In transition systems, or automata, we have $\Omega \sim_m \Omega|\overline{a}$ (where Ω is the looping process, producing infinitely many silent transitions). This might be surprising, because the context $C = a.\heartsuit\,|\,\Box$ intuitively should distinguish these processes, by being orthogonal to $\Omega|\overline{a}$ but not to Ω alone. However, it is not orthogonal to $\Omega|\overline{a}$, because $C[\Omega|\overline{a}]$ has an infinite looping trace giving priority to Ω. This looping trace is unfair, because the synchronisation on a is never performed. Thus, one may view the equivalence $\Omega \sim_m \Omega|\overline{a}$ as exploiting potential unfairness of a hypothetical scheduler.

Usually, concurrency theorists consider this too coarse, and resort to *fair* testing equivalence.

Definition 19 *A process P is* fair orthogonal *to a context C, notation $P \perp^f C$, when all finite traces of $C[P]$ extend to traces that play $\heartsuit$ at some point.*

Again, $P^{\perp^f}$ denotes the set of all contexts fair orthogonal to P.

Definition 20 *P and Q are* fair equivalent*, notation $P \sim_f Q$, when $P^{\perp^f} = Q^{\perp^f}$.*

This solves the issue, i.e., $\Omega \not\sim_f \Omega|\overline{a}$.

In summary, the mainstream setting for testing equivalences relies on traces; and the notion of maximality for traces is intrinsically unfair. This is usually rectified by resorting to fair testing equivalence over must testing equivalence. Our setting is more flexible, in the sense that maximal plays are better behaved than maximal traces. In terms of the previous section, this allows viewing the looping trace $\Omega|\overline{a}|a.\heartsuit \xrightarrow{\tau} \Omega|\overline{a}|a.\heartsuit \xrightarrow{\tau} \ldots$ as non-maximal. In the next sections, we define an abstract notion of interactive equivalence (still in the particular case of CCS but in our setting) and we instantiate it to define and study the counterparts of must and fair testing equivalences.

5.2 Interactive equivalences

Definition 21 *A play is* closed-world *when it is a composite of closed-world extended moves.*

Equivalently, a play is closed-world when all of its basic moves are part of a closed-world move.

Let $\mathbb{W} \hookrightarrow \mathbb{E}$ be the full subcategory of closed-world plays, $\mathbb{W}(X)$ being the *fibre* over X for the projection functor $\mathbb{W} \to \mathbb{B}$, i.e., the subcategory of $\mathbb{W}$ consisting of closed-world plays with base X, and morphisms (id_X, k) between them[5].

Let the category of *closed-world behaviours* on X be the category $\mathsf{G}_X = \widehat{\mathbb{W}(X)}$ of presheaves on $\mathbb{W}(X)$. We may now put:

Definition 22 *An* observable criterion *consists for all positions X, of a replete subcategory $\perp\!\!\!\perp_X \hookrightarrow \mathsf{G}_X$.*

[5]This is not exactly equivalent to what could be noted $\mathbb{W}_X$, since in the latter there are objects $U \hookleftarrow Y \hookrightarrow X$ with a strict inclusion $Y \hookrightarrow X$. However, both should be equivalent for what we do in this paper, i.e., fair and must equivalences.

Recall that $\perp\!\!\!\perp_X$ being replete means that for all $F \in \perp\!\!\!\perp_X$ and isomorphism $f\colon F \to F'$ in G_X, F' and f are in $\perp\!\!\!\perp_X$.

An observable criterion specifies the class of 'successful', closed-world behaviours. The two criteria considered below are two ways of formalising the idea that a successful behaviour is one in which all accepted closed-world plays are 'successful', in the sense that some player plays the tick move at some point.

We now define interactive equivalences. Recall that $[F, G]$ denotes the amalgamation of F and G, and that right Kan extension along i_Z^{op} induces a functor $\mathrm{Ran}_{i_Z^{op}}\colon \widehat{\mathbb{V}_Z} \to \widehat{\mathbb{E}_Z}$. Furthermore, precomposition with the canonical inclusion $j_Z\colon \mathbb{W}(Z) \hookrightarrow \mathbb{E}_Z$ induces a functor $j_Z^*\colon \widehat{\mathbb{E}_Z} \to \widehat{\mathbb{W}(Z)}$. Composing the two, we obtain a functor $\mathsf{Gl}\colon \mathsf{S}_Z \to \mathsf{G}_Z$:

$$\mathsf{S}_Z = \widehat{\mathbb{V}_Z} \xrightarrow{\mathrm{Ran}_{i_Z^{op}}} \widehat{\mathbb{E}_Z} \xrightarrow{j_Z^*} \widehat{\mathbb{W}(Z)} = \mathsf{G}_Z.$$

Definition 23 *For any innocent strategy F on X and any pushout square P of positions as on the right, with I consisting only of channels, let $F^{\perp\!\!\!\perp_P}$ be the class of all innocent strategies G on Y such that $\mathsf{Gl}([F, G]) \in \perp\!\!\!\perp_Z$.*

$$\begin{array}{ccc} I & \longrightarrow & Y \\ \downarrow & & \downarrow \\ X & \longrightarrow & Z \end{array} \qquad (8)$$

Here, G is thought of as a *test* for F. Also, P denotes the whole pushout square and $F^{\perp\!\!\!\perp_P}$ denotes all the valid tests for the considered pushout square P. From the CCS point of view, I corresponds to the set of names shared by the process under observation (F) and the test (G).

Definition 24 *Any two innocent strategies $F, F' \in \mathsf{S}_X$ are* $\perp\!\!\!\perp$-equivalent*, notation $F \sim_{\perp\!\!\!\perp} F'$, iff for all pushouts P as in 8, $F^{\perp\!\!\!\perp_P} = F'^{\perp\!\!\!\perp_P}$.*

5.3 Fair vs. must

Let us now define fair and must testing equivalences. Let a closed-world play be *successful* when it contains a $\heartsuit_n$. Furthermore, for any closed-world behaviour $G \in \mathsf{G}_X$ and closed-world play $U \in \mathbb{W}(X)$, an *extension* of a state $\sigma \in G(U)$ to U' is a $\sigma' \in G(U')$ with $i\colon U \to U'$ and $G(i)(\sigma') = \sigma$. The extension σ' is *successful* when U' is. The intuition is that the behaviour G, before reaching U' with state σ', passed through U with state σ.

Definition 25 *The* fair *criterion* $\perp\!\!\!\perp^f$ *contains all closed-world behaviours* G *such that any state* $\sigma \in G(U)$ *for finite* U *admits a successful extension.*

Now call an extension of $\sigma \in G(U)$ *strict* when $U \to U'$ is not surjective, or, equivalently, when U' contains more moves than U. For any closed-world behaviour $G \in \mathsf{G}_X$, a state $\sigma \in G(U)$ is *G-maximal* when it has no strict extension.

Definition 26 *Let the* must *criterion* $\perp\!\!\!\perp^m$ *consist of all closed-world behaviours* G *such that for all closed-world* U *and* G*-maximal* $\sigma \in G(U)$*,* U *is successful.*

As explained in the introduction and Section 5.1, unlike in the standard setting, this definition of must testing equivalence distinguishes between the processes Ω and $\Omega|\overline{a}$. Indeed, take the CCS context $C = a.\heartsuit \mid \square$, which we can implement by choosing as a test the strategy $T = [\![a.\heartsuit]\!]$ on a single player knowing one channel a. Taking I to consist of the sole channel a, the pushout Z as in Definition 23 consists of two players, say x for the observed strategy and y for the test strategy, sharing the channel a. Now, assuming that Ω loops deterministically, the global behaviour $G = \mathsf{Gl}([[\![P]\!], T])$ has exactly one state on the identity play, and again exactly one state on the play π_1 consisting of only one fork move by x. Thus, G reaches a position with three players, say x_1 playing Ω, x_2 playing $\overline{a}$, and y playing $a.\heartsuit$. The play with infinitely many silent moves by x_1 is not maximal: we could insert (anywhere in the sequence of moves by x_1) a synchronisation move by x_2 and y, and then a tick move by the avatar of y. Essentially: our notion of play is more fair than just traces.

To get more intuition about must testing equivalence in our setting, we prove that it actually coincides with the testing equivalence generated by the following criterion:

Definition 27 *The* spatially fair *criterion* $\perp\!\!\!\perp^{sf}$ *contains all closed-world behaviours* G *such that any state* $\sigma \in G(U)$ *admits a successful extension.*

This criterion is almost like the fair criterion, except that we do not restrict to finite plays. The key result to show the equivalence is:

Theorem 4 *For any innocent strategy* F *on* X*, any state* $\sigma \in \mathsf{Gl}(F)(U)$ *admits a* $\mathsf{Gl}(F)$*-maximal extension.*

The proof is in Appendix B. Thanks to the theorem, we have:

Lemma 6 *For all* $F \in \mathsf{S}_X$, $\mathsf{Gl}(F) \in \perp\!\!\!\perp^m_X$ *iff* $\mathsf{Gl}(F) \in \perp\!\!\!\perp^{sf}_X$.

Proof: Let $G = \mathsf{Gl}(F)$.

(⇒) By Theorem 4, any state $\sigma \in G(U)$ has a G-maximal extension $\sigma' \in G(U')$, which is successful by hypothesis, hence σ has a successful extension.

(⇐) Any G-maximal $\sigma \in G(U)$ admits by hypothesis a successful extension which may only be on U by G-maximality, and hence U is successful. □

(Note that U is not necessarily finite in the proof of the right-to-left implication, so that the argument does not apply to the fair criterion.)

Now comes the expected result:

Theorem 5 *For all* $F, F' \in \mathsf{S}_X$, $F \sim_{\perp\!\!\!\perp^m} F'$ *iff* $F \sim_{\perp\!\!\!\perp^{sf}} F'$.

Proof: *(⇒)* Consider two innocent strategies F and F' on X, and an innocent strategy G on Y (as in the pushout (8)). As in spatial decomposition (Proposition 5), copairing induces an isomorphism $\mathsf{S}_X \times \mathsf{S}_Y \to \mathsf{S}_Z$, and we have, using Lemma 6:

$$\begin{aligned} \mathsf{Gl}[F, G] \in \perp\!\!\!\perp^{sf} \quad & \text{iff} \quad \mathsf{Gl}[F, G] \in \perp\!\!\!\perp^{m} \\ & \text{iff} \quad \mathsf{Gl}[F', G] \in \perp\!\!\!\perp^{m} \\ & \text{iff} \quad \mathsf{Gl}[F', G] \in \perp\!\!\!\perp^{sf} \end{aligned}$$

(⇐) Symmetric. □

Intuitively, must testing only considers spatially fair schedulings, in the sense that all players appearing in a play should be given the opportunity to play: no one should starve.

However, this is not the only source of unfairness, so that must testing and fair testing differ. To see this, consider the CCS process $P = \nu b.\mathtt{rec}\ x(a,b)\ \mathtt{:=}\ \overline{b}|(b.(x(a,b)) + \overline{a})\ \mathtt{in}\ x(a,b)$, that can repeatedly perform synchronisations on the private channel b, until it chooses to perform an output on a. We have $[\![\Omega]\!] \sim^{sf} [\![P]\!]$ while $[\![\Omega]\!] \not\sim^{f} [\![P]\!]$. Indeed, since the choice between doing a synchronisation on b or an output on a is done by a single player, the infinite play where the output on a is never performed is maximal: no player starve, we just have a player that repeatedly chooses the same branch, in an unfair way.

We leave for future work the investigation of such unfair scenarios and their correlation to the corresponding behaviours in classical presentations of CCS.

A Temporal decomposition

This section is a proof of Theorem 2. Let us first review the general equivalences mentioned in the proof sketch. The product of a family of presheaf categories is isomorphic to the category of presheaves over the corresponding coproduct of categories:

Lemma 7 *We have* $\prod_{M \in \mathcal{M}_n} \mathsf{S}_{\mathrm{cod}(M)} \cong [\sum_{M \in \mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}]$.

Furthermore, let the functor $\Delta \colon \mathsf{Set} \to \widehat{\mathbb{C}}$ map any set X to the constant presheaf mapping any $C \in \mathbb{C}$ to X. We have:

Lemma 8 *For any small category* $\mathbb{C}$, $\mathrm{Fam}(\widehat{\mathbb{C}}) \simeq (\widehat{\mathbb{C}} \downarrow \Delta)$.

Proof: A generalisation of the more well-known $\mathsf{Set}^X \simeq \mathsf{Set}/X$. $\square$

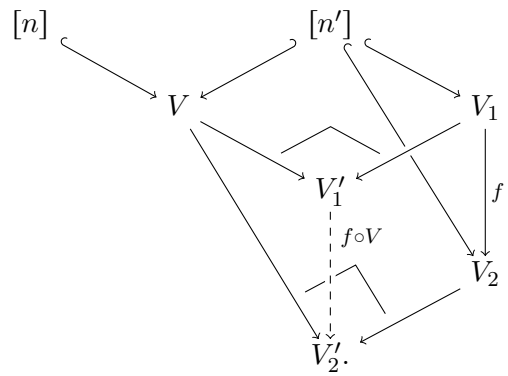
Corollary 1 *We have:*

$$\mathrm{Fam}\left(\prod_{M \in \mathcal{M}_n} \mathsf{S}_{\mathrm{cod}(M)}\right) \simeq ([\sum_{M \in \mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}] \downarrow \Delta).$$

We now construct the lax pushout (6). A first step is the construction, for each move $[n] \hookrightarrow M \hookleftarrow [n']$, of a functor $(- \circ M) \colon \mathbb{V}_{[n']} \to \mathbb{V}_{[n]}$ given by precomposition with M in $\mathsf{Cospan}(\widehat{\mathbb{C}})$. This functor maps any $V_1 \colon [n'] \hookrightarrow V_1$ to the view $V_1 \circ M$, i.e., the view $[n] \hookrightarrow V_1'$ defined by the colimit

$$\begin{tikzcd}
{[n]} \arrow[dr, hook] & & {[n']} \arrow[dl, hook] \arrow[dr, hook] & \\
& M \arrow[dr] & & V_1 \arrow[dl] \\
& & V_1'. \arrow[ur, dashed] &
\end{tikzcd}$$

This of course relies on the choice of such a colimit for every V and V_1. Any morphism $f \colon V_1 \to V_2$ in $\mathbb{V}_{[n']}$, letting $V_2' = V_2 \circ V$, is mapped to the dashed morphism induced by universal property of pushout in



Once the choice has been made on objects, the map for morphisms is determined uniquely.

This family of functors allows us to decompose $\mathbb{V}_{[n]}$ as follows:

Lemma 9 *The diagram*

$$\begin{tikzcd} \sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)} \arrow[r, equal] \arrow[d, "!"'] & \sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)} \arrow[d, "{[-\circ M]_{M\in\mathcal{M}_n}}"] \arrow[dl, Rightarrow, "\lambda"'] \\ 1 \arrow[r, "\ulcorner id_{[n]} \urcorner"'] & \mathbb{V}^{op}_{[n]} \end{tikzcd} \tag{9}$$

is a lax pushout, where $\lambda_{M,V}\colon id_{[n]} \to M \circ V$, seen in $\mathbb{V}_{[n]}$, is the obvious inclusion.

Proof: For any category $\mathbb{C}$, taking such a lax pushout of $id_{\mathbb{C}}$ with 1 just adds a terminal object to $\mathbb{C}$. The rest is an easy verification. A dual result of course holds with $\mathbb{V}_{[n]}$, reversing the direction of λ. $\square$

Now, it is well-known that, in any small 2-category $\mathbb{K}$, any contravariant hom-2-functor, i.e., 2-functor of the shape $\mathbb{K}(-, X)$ for $X \in K$, maps weighted colimits in $\mathbb{K}$ to weighted limits in Cat. For an introduction to weighted limits and colimits in the case of enrichment over Cat, see Kelly [26]. Here, for any 2-category P, and 2-functors $G\colon P \to \mathbb{K}$ and $J\colon P^{op} \to \mathsf{Cat}$, any colimit $L = J \star G$ of G weighted by J with unit $\xi\colon J \to \mathbb{K}(G(-), L)$ in $[P^{op}, \mathsf{Cat}]$ is mapped, for any object $X \in \mathbb{K}$, by the hom-2-functor $\mathbb{K}(-, X)$ to a limit of $\mathbb{K}(G(-), X)\colon P^{op} \to \mathsf{Cat}$ weighted by J in Cat, with unit $\mathbb{K}(\xi, X)\colon J \to \mathsf{Cat}(\mathbb{K}(L, X), \mathbb{K}(G(-), X))$, in Cat. In particular, lax pushouts are mapped to lax pullbacks. As usual, considering a larger universe, we may replace Cat with CAT and obtain the same results with $\mathbb{K} = \mathsf{Cat}$.

Recalling our lax pushout (9) and taking the hom-categories to Set, we obtain a lax pullback

$$\begin{tikzcd}
{[\sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}]} \arrow[r, equal] & {[\sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}]} \\
\mathsf{Set} \arrow[u, "!^*"] \arrow[r, leftarrow] & \mathsf{S}_{[n]} \arrow[u] \arrow[ul, Rightarrow, "\lambda^*"']
\end{tikzcd}$$

in CAT, i.e., a comma category. But observe that restriction along ! is precisely $\Delta\colon \mathsf{Set} \to [\sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}]$, so we have indeed shown that $\mathsf{S}_{[n]}$ is a comma category $[\sum_{M\in\mathcal{M}_n} \mathbb{V}^{op}_{\mathrm{cod}(M)}, \mathsf{Set}] \downarrow \Delta$.

B Maximal extensions

This section is a proof of Theorem 4.

Lemma 10 *For any position X, the category $\mathbb{W}(X)$ of closed-world plays is a preorder.*

Proof: Easy. □

In the following, we consider the quotient poset.

Lemma 11 *In $\mathbb{W}(X)$, any non-decreasing chain admits an upper bound.*

Recall $\mathcal{M}$, the graph of all basic moves, and the set $\mathcal{M}_n$ of edges from n, for each n. Let now, for each n, $\mathcal{M}^f_n$ be the analogous set with full moves, i.e., the set of isomorphism classes of full moves from $[n]$.

Lemma 12 *For each play $U \in \mathbb{E}_X$, the coproduct of all s maps from full moves*

$$\left(\sum_{n\in\mathsf{FinOrd}} \sum_{M\in\mathcal{M}^f_n} U(M)\right) \to \sum_{n\in\mathsf{FinOrd}} U[n], \tag{10}$$

is injective.

Recall here that for forking, we have also called s the common composite $l \circ s = r \circ s$ (see the discussion following Definition 3).

Proof: By induction on U. □

Lemma 13 *Any non-decreasing sequence in the poset $\mathbb{W}(X)$ admits its colimit in $\widehat{\mathbb{C}}$ as an upper bound.*

Proof: Consider any increasing sequence $U^1 \hookrightarrow U^2 \hookrightarrow \dots$ of plays in $\mathbb{W}(X)$. Let U be its colimit in $\widehat{\mathbb{C}}$. We want to prove that U is a play.

First, observe that U satisfies joint injectivity of s-maps as in Lemma 12: indeed, if we had a player p and two full moves M and M' such that $s(M) = s(M') = p$, then all of M, M', and p would appear in some U^i, which, being a play, has to satisfy joint injectivity.

For each n, U^n comes with a sequence of compatible (closed-world) extended moves

$$X = X_0^n \hookrightarrow M_1^n \hookleftarrow X_1^n \hookrightarrow \dots \hookleftarrow X_{i-1}^n \hookrightarrow M_i^n \hookleftarrow X_i^n \hookrightarrow \dots$$

which are also (by the colimit cocone) morphisms over U in $\widehat{\mathbb{C}}$. For each $i \geqslant 1$, taking the colimit of the i first moves yields a finite play $X \hookrightarrow U_i^n \hookleftarrow X_i^n$. By convention, letting $U_0^n = X$ extends this to $i \geqslant 0$. Similarly, we may consider all the given plays infinite, by accepting not only extended moves, but also identity cospans.

We consider the poset of pairs $(N, n) \in \{(0,0)\} \uplus \sum_{N \in \mathsf{FinOrd}^*} N$, with lexicographic order, i.e., $(N, n) \leqslant (N', n')$ when $N < N'$ or when $N = N'$ and $n \leqslant n'$.

We will construct by induction on (N, n) a sequence of composable closed-world moves, with colimit U', such that for all (N, n), $U_{N-n+1}^n \subseteq U'$ in $\mathbb{W}(X)/U$. More precisely, we construct for each (N, n) an integer $K_{N,n}$ and a sequence

$$X = X_0^{N,n} \hookrightarrow M_1^{N,n} \hookleftarrow X_1^{N,n} \hookrightarrow \dots \hookleftarrow X_{K_{N,n}-1}^{N,n} \hookrightarrow M_{K_{N,n}}^{N,n} \hookleftarrow X_{K_{N,n}}^{N,n},$$

(again, if $K_{N,n} = 0$, we mean the empty sequence) such that

- for all $(N', n') < (N, n)$, we have $K_{N',n'} \leqslant K_{N,n}$ and the sequence $(M_i^{N',n'})_{i \in K_{N',n'}}$ is a prefix of $(M_{i \in K_{N,n}}^{N,n})$;
- and the colimit, say $U_{N,n}$, of $(M_i^{N,n})_{i \in K_{N,n}}$ is such that for all $(N', n') \leqslant (N, n)$, $U_{N-n'+1}^{n'} \subseteq U_{N,n}$ in $\mathbb{W}(X)/U$.

For the base case, we let $K_{0,0} = 0$, which forces $M^{0,0}$ to be the empty sequence on X.

For the induction step, consider any $(N, n) \neq (0, 0)$, and let (N_0, n_0) be the predecessor of (N, n). The induction hypothesis gives a K_{N_0,n_0} and a sequence $(M_i^{N_0,n_0})_{i \in K_{N_0,n_0}}$ satisfying some hypotheses, among which the existence of a diagram

$$\begin{array}{ccccccccc}
X & \longrightarrow & U^n_{N-n} & \longleftarrow & X^n_{N-n} & \longrightarrow & M^n_{N-n+1} & \longleftarrow & X^n_{N-n+1} \\
\| & & \downarrow & & & & & & \\
X & \longrightarrow & U_{N_0,n_0} & \longleftarrow & X^{N_0,n_0}_{K_{N_0,n_0}} & & & &
\end{array}$$

over U.

Now, if $M^n_{N-n+1} \to U$ factors through U_{N_0,n_0}, then we put $K_{N,n} = K_{N_0,n_0}$ and $(M^{N,n}_i)_{i \in K_{N,n}} = (M^{N_0,n_0}_i)_{i \in K_{N_0,n_0}}$, and all induction hypotheses go through.

Otherwise, M^n_{N-n+1} is played by players in X^n_{N-n} which are not in the joint image of all s maps (10) in U_{N_0,n_0}, otherwise s maps in U could not be jointly injective, contradicting Lemma 12. Technically, the diagram

$$X^n_{N-n} \to M^n_{N-n+1} \leftarrow X^n_{N-n+1}$$

is obtained by pushing some (non-extended) closed-world move $Y \to M \leftarrow Y'$ along some morphism $I \to Z$ from an interface I, and the induced morphism $Y \to X^n_{N-n} \to U^n_{N-n} \to U_{N_0,n_0}$ factors through $X^{N_0,n_0}_{K_{N_0,n_0}}$. We consider the subposition $Z' \subseteq X^{N_0,n_0}_{K_{N_0,n_0}}$ making

$$\begin{array}{ccc}
I & \hookrightarrow & Y \\
\downarrow & & \downarrow \\
Z' & \hookrightarrow & X^{N_0,n_0}_{K_{N_0,n_0}}
\end{array}$$

a pushout; Z' consists of the players in $X^{N_0,n_0}_{K_{N_0,n_0}}$ that are not in the image of Y, plus their names, plus possibly missing names from I.

Then, pushing $Y \to M \leftarrow Y'$ along $I \to Z'$, we obtain an extended move $X^{N_0,n_0}_{K_{N_0,n_0}} \hookrightarrow M' \hookleftarrow X'$. We let $K_{N,n} = K_{N_0,n_0} + 1$ and define $(M^{N,n}_i)_{i \in K_{N,n}}$ to be the extension of $(M^{N_0,n_0}_i)_{i \in K_{N_0,n_0}}$ by M'. This induces a unique map $U_{N,n} \to U$ by universal property of $U_{N,n}$ as a colimit. All induction hypotheses go through; in particular, U^n_{N-n+1} is a union $U^n_{N-n} \cup M^n_{N-n+1}$ in $\mathbb{W}(X)/U$, and actually a union $U^n_{N-n} \cup M$; similarly, $U_{N,n} = U_{N_0,n_0} \cup M$; so, since we have $U^n_{N-n} \subseteq U_{N_0,n_0}$ by induction hypothesis, we obtain $U^n_{N-n+1} \subseteq U_{N,n}$.

The sequences $M^{N,n}$ induce by union a possibly infinite sequence of closed-world extended moves, i.e., a closed-world play U', such that for all (N,n), $U^n_{N-n+1} \subseteq U'$, hence, for all n, $U^n \subseteq U' \subseteq U$, i.e., $U' \cong U$. Thus, U is indeed a play. $\square$

We are almost ready for proving Theorem 4. We just need one more lemma. Consider any innocent strategy F on X, play $U \in \mathbb{W}(X)$, and any state $\sigma \in \mathsf{Gl}(F)(U)$. Consider now the poset F_σ of $\mathsf{Gl}(F)$-extensions of σ (made into a poset by choosing a skeleton of $\mathbb{W}(X)$), where $\sigma' \in F(U') \leqslant \sigma'' \in F(U'')$ iff $U' \leqslant U''$. This poset is not empty, since it contains σ. Furthermore, we have:

Lemma 14 *Any non-decreasing sequence in F_σ admits an upper bound.*

Proof: Any such sequence, say $(\sigma_i)_{i \in \mathsf{FinOrd}}$, induces a non-decreasing sequence of plays in $\mathbb{W}(X)$, say $(U_i)_i$, which by Lemma 13 admits its colimit, say U', as an upper bound. Now, any view inclusion $j \colon V \hookrightarrow U'$, factors through some U_i, and we let $\sigma_j = (\sigma_i)_{|V}$ (this does not depend on the choice of i). This assignment determines (by innocence of F and by construction of the right Kan extension as an end) an element $\sigma' \in F(U')$, which is an upper bound for $(\sigma_i)_{i \in \mathsf{FinOrd}}$. $\square$

Proof of Theorem 4: Consider any innocent strategy F on X, play $U \in \mathbb{W}(X)$, and any state $\sigma \in \mathsf{Gl}(F)(U)$. Consider as above the poset F_σ of $\mathsf{Gl}(F)$-extensions of σ. By the last lemma, we may apply Zorn's lemma to choose a maximal element of F_σ, which is a $\mathsf{Gl}(F)$-maximal extension of σ. $\square$

References

[1] Emmanuel Beffara. *Logique, réalisabilité et concurrence*. PhD thesis, Université Paris 7, December 2005.

[2] Marcello M. Bonsangue, Jan J. M. M. Rutten, and Alexandra Silva. A Kleene theorem for polynomial coalgebras. In Luca de Alfaro, editor, *FOSSACS*, volume 5504 of *Lecture Notes in Computer Science*, pages 122–136. Springer, 2009.

[3] Ed Brinksma, Arend Rensink, and Walter Vogler. Fair testing. In Insup Lee and Scott A. Smolka, editors, *CONCUR*, volume 962 of *Lecture Notes in Computer Science*, pages 313–327. Springer, 1995.

[4] Albert Burroni. Higher-dimensional word problems with applications to equational logic. *Theoretical Computer Science*, 115(1):43–62, 1993.

[5] Aurelio Carboni and Peter Johnstone. Connected limits, familial representability and artin glueing. *Mathematical Structures in Computer Science*, 5(4):441–459, 1995.

[6] Aurelio Carboni and Peter Johnstone. Corrigenda for 'connected limits, familial representability and artin glueing'. *Mathematical Structures in Computer Science*, 14(1):185–187, 2004.

[7] Rocco De Nicola and Matthew Hennessy. Testing equivalences for processes. *Theor. Comput. Sci.*, 34:83–133, 1984.

[8] Olivier Delande and Dale Miller. A neutral approach to proof and refutation in mall. In *LICS '08* [30], pages 498–508.

[9] H. Ehrig, H.-J. Kreowski, Ugo Montanari, and Grzegorz Rozenberg, editors. *Handbook of Graph Grammars and Computing by Graph Transformation, Volume 3: Concurrency, Parallelism and Distribution*. World Scientific, 1999.

[10] Marcelo P. Fiore. Second-order and dependently-sorted abstract syntax. In *LICS '08* [30], pages 57–68.

[11] Fabio Gadducci, Reiko Heckel, and Mercè Llabrés. A bi-categorical axiomatisation of concurrent graph rewriting. *Electronic Notes in Theoretical Computer Science*, 29, 1999.

[12] Fabio Gadducci and Ugo Montanari. The tile model. In Gordon D. Plotkin, Colin Stirling, and Mads Tofte, editors, *Proof, Language, and Interaction*, pages 133–166. The MIT Press, 2000.

[13] Jean-Yves Girard. Locus solum: From the rules of logic to the logic of rules. *Mathematical Structures in Computer Science*, 11(3):301–506, 2001.

[14] Yves Guiraud and Philippe Malbos. Higher-dimensional categories with finite derivation type. *Theory and Applications of Categories*, 22(18):420–278, 2009.

[15] André Hirschowitz and Marco Maggesi. Modules over monads and linearity. In Daniel Leivant and Ruy J. G. B. de Queiroz, editors, *WoLLIC*, volume 4576 of *Lecture Notes in Computer Science*, pages 218–237. Springer, 2007.

[16] André Hirschowitz and Marco Maggesi. Modules over monads and initial semantics. *Information and Computation*, 208(5):545–564, 2010.

[17] André, Michel, and Tom Hirschowitz. Contraction-free proofs and finitary games for linear logic. *Electronic Notes in Theoretical Computer Science*, 249:287–305, 2009.

[18] Tom Hirschowitz. Cartesian closed 2-categories and permutation equivalence in higher-order rewriting. Preprint, 2010.

[19] Tom Hirschowitz and Damien Pous. Innocent strategies as presheaves and interactive equivalences for CCS. In Alexandra Silva, Simon Bliudze, Roberto Bruni, and Marco Carbone, editors, *ICE*, volume 59 of *EPTCS*, pages 2–24, 2011.

[20] Martin Hyland. *Semantics and Logics of Computation*, chapter Game Semantics. Cambridge University Press, 1997.

[21] Bart Jacobs. *Categorical Logic and Type Theory*. Number 141 in Studies in Logic and the Foundations of Mathematics. North Holland, Amsterdam, 1999.

[22] Ole H. Jensen and Robin Milner. Bigraphs and mobile processes (revised). Technical Report TR580, University of Cambridge, 2004.

[23] P. T. Johnstone, S. Lack, and P. Sobociński. Quasitoposes, quasiadhesive categories and Artin glueing. In *CALCO*, volume 4624 of *LNCS*, pages 312–326. Springer Verlag, 2007.

[24] André Joyal, Mogens Nielsen, and Glynn Winskel. Bisimulation and open maps. In *LICS '93*, pages 418–427. IEEE Computer Society, 1993.

[25] Stefano Kasangian and Anna Labella. Observational trees as models for concurrency. *Mathematical Structures in Computer Science*, 9(6):687–718, 1999.

[26] G. M. Kelly. Elementary observations on 2-categorical limits. *Bulletin of the Australian Mathematical Society*, 39:301–317, 1989.

[27] Joachim Kock. Polynomial functors and trees. *International Mathematics Research Notices*, 2011(3):609–673, 2011.

[28] Jean-Louis Krivine. Dependent choice, 'quote' and the clock. *Theor. Comput. Sci.*, 308(1-3):259–276, 2003.

[29] James J. Leifer and Robin Milner. Deriving bisimulation congruences for reactive systems. In Catuscia Palamidessi, editor, *CONCUR*, volume 1877 of *Lecture Notes in Computer Science*, pages 243–258. Springer, 2000.

[30] *Proceedings of the Twenty-Third Annual IEEE Symposium on Logic in Computer Science, LICS 2008, 24-27 June 2008, Pittsburgh, PA, USA*. IEEE Computer Society, 2008.

[31] Saunders Mac Lane. *Categories for the Working Mathematician*. Number 5 in Graduate Texts in Mathematics. Springer, 2nd edition, 1998.

[32] Saunders MacLane and Ieke Moerdijk. *Sheaves in Geometry and Logic: A First Introduction to Topos Theory*. Universitext. Springer, 1992.

[33] Paul-André Melliès. Asynchronous games 2: the true concurrency of innocence. In *Proc. CONCUR '04*, volume 3170 of *LNCS*, pages 448–465. Springer Verlag, 2004.

[34] Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *LNCS*. Springer, 1980.

[35] V. Natarajan and Rance Cleaveland. Divergence and fair testing. In Zoltán Fülöp and Ferenc Gécseg, editors, *ICALP*, volume 944 of *Lecture Notes in Computer Science*, pages 648–659. Springer, 1995.

[36] Tobias Nipkow. Higher-order critical pairs. In *LICS '91*, pages 342–349. IEEE Computer Society, 1991.

[37] Gordon D. Plotkin. A structural approach to operational semantics. DAIMI Report FN-19, Computer Science Department, Aarhus University, 1981.

[38] Julian Rathke and Pawel Sobocinski. Deconstructing behavioural theories of mobility. In *IFIP TCS*, volume 273 of *IFIP*, pages 507–520. Springer, 2008.

[39] Vladimiro Sassone and Pawel Sobociński. Deriving bisimulation congruences using 2-categories. *Nordic Journal of Computing*, 10(2), 2003.

[40] Peter Sewell. From rewrite rules to bisimulation congruences. In Davide Sangiorgi and Robert de Simone, editors, *CONCUR*, volume 1466 of *Lecture Notes in Computer Science*, pages 269–284. Springer, 1998.

[41] Daniele Turi and Gordon D. Plotkin. Towards a mathematical operational semantics. In *LICS '97*, pages 280–291, 1997.

[42] Rob J. van Glabbeek. The linear time-branching time spectrum (extended abstract). In Jos C. M. Baeten and Jan Willem Klop, editors, *CONCUR*, volume 458 of *Lecture Notes in Computer Science*, pages 278–297. Springer, 1990.

[43] Angelo Vistoli. Notes on Grothendieck topologies, fibered categories and descent theory. Preprint. `http://arxiv.org/abs/math/0412512`., 2007.