Games for Core ML

Thomas Cuvillier

July 21, 2014

WORK IN PROGRESS

1 Introduction

We introduce in this paper a new game semantics for a toy language that combines several interesting features, in order to pave the way through the first ever fully complete model of a realistic programming language, ML. We furthermore rely on the nominal set theory to obtain a clean treatment of names and resource creations, and avoiding the use of bad variables.

ML is a particularly relevant language to study, both due to its wide-use through the computer scientist community, and its large number of programming features. These include general reference, (weak) polymorphism, exception, product, sum and recursive type. Hence, this work faces a double challenge; accommodate each of these feature with one another and places them in the nominal context. Because of the difficulty, the task was splitted in two. This first part is therefore restricted to a complete fragment of ML containing only non general references and no exceptions.

First of all, it should be noticed that this work aims to be fully self-contained, and as complete as possible. That is, every proof of property is included in it and is fully original, and no proofs (or part of it) is left aside as straightforward.

The first part of the paper is a description of the language, its operationnal semantics. Following is a brief introduction to nominal set theory. Relying on that is a description of the nominal games category, the construction of sum, product and function space, together with the nominal translation from types to arenas. We introduce some new conditions for legal sequences that take care of the polymorphic types. They differentiate themselves from the one present in the literature in the sense that, for ML, the $\forall$ symbol can only be in the leftmost part of a type. As a result, we used names together with some simple player-opponent conditions, instead of copycat links, to highlight the relations between the variables in a term of polymorphic type.

The second step consisted in investigating the properties of the defined category, and more particularly in showing that the subcategory of open terms of it formed indeed a Freyd category. That includes defining a subcategory of it with products, and the adjunction between the both that yields partial exponentials.
Finally, we explore the way substitution can be defined, as well as its (pre) system-\(\mathcal{I}\) structure. Relying on these results, and the CPO structure, we give a full description of the solution arena for recursive types, and prove that this arena is indeed a minimal invariant of the functor in the pre-CPO enriched category of values.

2 Language and operational semantics

2.1 Types and terms

We define a complete fragment of ML that have sum, product, recursive types and weak polymorphism. However, we restrict references to a set of basic types, introduced along the lines. The types of our languages are defined by induction below.

\[
\begin{align*}
\mathit{b} & ::= \text{unit} \mid \text{int} \mid \text{ref} \ b \\
\mathit{T} & ::= X \mid b \mid T \to T \mid T \times T \mid T + T \mid \mu X. T \\
\Sigma & ::= T \mid \forall X. \Sigma
\end{align*}
\]

The types \(b\) will be referred to as basic types and the types \(T\) as open types. Though, it should be notice that \(X\) might appear as a closed variable in an open type, as the operator \(\mu\) might close it. Also note that even if we did not include the type bool, it is easily encodable as the type \(\text{unit} + \text{unit}\).

As we rely on nominal set theory to deal with references, we therefore need some brief definitions before introducing the grammar of terms. We assume a set \(\mathbb{A} = \biguplus_b \mathbb{A}_b \uplus \mathbb{N}\) of disjoint union of infinite countable set of reference names, together with two infinite countable sets \(\text{Var}\) and \(\text{TVar}\) where \(\text{Var}\) stands for the set of variable symbols \(x,y,\ldots\), and \(\text{TVar}\) the one of type variables \(X,Y,Z,\ldots\). The set \(\mathbb{L} = \biguplus_b \mathbb{A}_b\) will be called the set of locations, and the set \(\mathbb{N}\) the set of variable distinguishers. We will give some additional structure to the set \(\mathbb{A}\) in the next section, highlighting its nominal structure. We call “names” the elements of \(\mathbb{A}\).

The terms of our language are given by the following grammar:

\[
\begin{align*}
\mathit{M} ::= & \ x \mid \ * \mid i \mid l \mid () \mid \lambda x^T. M \mid \Lambda X. M \mid \langle M, M \rangle \mid \pi_1 M \mid \pi_2 M \mid \text{case}(x : M, M, M) \mid \text{inr}(M) \mid \text{inl}(M) \mid \\
& \text{fix } x^S. M \mid M(\Sigma) \mid MM \mid \text{fold}_{\mu X,S} M \mid \text{unfold}_{\mu X,S} M \mid \text{ref } M \mid !M \mid M = M \mid M := M \mid M \oplus M \\
& \text{let } x = M \ \text{in } M
\end{align*}
\]

where \(i\) ranges over the integers, \(x\) over the set of variables, \(X\) over the set of type variables and \(l\) over the set of locations. \((\cdot)\) is the only element of unit type. To distinguish them for references, we call constant the terms \(i\) and \((\cdot)\). Formally, the set \(\mathbb{C}\) of constants is defined by \(\mathbb{C} = \mathbb{N} \uplus \text{unit}\). We now define the subset of the set of terms that will be referred to as values, and denoted by \(V\).

\[
\begin{align*}
\mathit{V} ::= & \ x \mid i \mid l \mid () \mid \lambda x. M \mid \langle V, V \rangle \mid \text{inr}(V) \mid \text{inl}(V) \mid \Lambda X. M \mid \text{fold}_{\mu X,S} V.
\end{align*}
\]
The type of each term is given by the following set of rules:

\[
\begin{align*}
& (,) : \text{unit} \quad \vdash i \in \mathbb{Z} \quad l \in \mathbb{L} \cap A_b \quad \Gamma \vdash M : T_1 \quad \Gamma \vdash M : T_2 \quad \Gamma \vdash \pi_1 M : \Sigma_{i} \times \Sigma_{i} \quad i = 1, 2 \\
& \Gamma \vdash M : T_1 \quad \Gamma \vdash M : T_2 \quad \Gamma, x : T_1 \vdash M_1 : T \quad \Gamma, x : T_2 \vdash M_2 : T \quad \Gamma \vdash (M_1, M_2) : T_1 \times T_2 \quad \Gamma \vdash \pi_1 M : \Sigma_{i}, i = 1, 2 \\
& \Gamma \vdash M : T \quad \Gamma \vdash x : M \vdash T_1 \quad \Gamma, x : T_2 \vdash M : T \quad \Gamma \vdash \text{case}(x : M, M_1, M_2) : T \\
& \Gamma \vdash \text{in}(M) : T_2 \oplus T_1 \quad \Gamma \vdash \text{inr}(M) : T_2 \oplus T_1 \quad \Gamma \vdash \text{inl}(M) : T_1 \oplus T_2 \\
& \Gamma, x : T_1 \vdash M_1 \quad \Gamma, x : T_2 \vdash M_2 \quad \Gamma \vdash \text{fix} x : T_1 \to T_2 \vdash M_1 : T_1 \rightarrow T_2 \quad \Gamma \vdash \text{fix} x : T_1 \to T_2 \vdash M_2 : T_1 \rightarrow T_2 \\
& \Gamma \vdash \mu X. T \quad \Gamma \vdash \lambda x : T. M : T_1 \to T_2 \quad \Gamma \vdash \mu X. T \quad \Gamma \vdash \Sigma[T/X] \quad \Gamma \vdash \Sigma[T/X] \\
& \Gamma \vdash \text{fold}_{\mu X. T} M : T[\mu X. T/X] \quad \Gamma \vdash \text{unfold}_{\mu X. T} M : T[\mu X. T/X] \quad \Gamma \vdash \text{fold}_{\mu X. T} M : \mu X. T \quad \Gamma \vdash \text{unfold}_{\mu X. T} M : \mu X. T \\
& \Gamma \vdash M_1 : \text{int} ; \Gamma \vdash M_2 : \text{int} \quad \Gamma \vdash M_1 : \text{int} \quad \Gamma \vdash M_2 : \text{int} \quad \Gamma \vdash M_1 \parallel M_2 : \text{int} \quad \Gamma \vdash M = N : \text{int} \\
& \Gamma, x : T_1 \vdash M_1 : T_2 \quad \Gamma \vdash N : T_1 \quad \Gamma \vdash \text{let} x = \text{Nin} M : T_2 \\
& \Gamma \vdash \text{let} x = \text{Nin} M : T_2 \\
\end{align*}
\]

2.2 Operational Semantics

We will call state $S$ a finite partial function from locations to values of base types, such that if $a \in A_b$ and $a \in \text{dom}(S)$ then $S(a) \in V_b$, for $b$ a specific base type, and $V_b$ is the set of value terms of type $b$. More formally, the set of state is defined by

\[
\{ f : \mathbb{L} \rightarrow \text{partial} \ V_{\text{basic}} \mid a \in A_b \Rightarrow f(b) \text{ of type } b \land f \text{ finite } \},
\]

and the set of basic values terms $V_{\text{basic}}$ is defined by

\[
V_{\text{basic}} ::= c \mid l \mid (\)
\]

where $c \in \mathbb{C}$ and $l \in \mathbb{L}$.

A program is a pair $(S, M)$ where $M$ is a closed type of our language, and $S$ is a state such that each name that appears in $M$ is in the domain of $S$. $S[a \mapsto V]$ stands for the state whose mapping is the same as $S$ expect for $a$ that is now mapped to $V$. The operational semantics is given under the form of small-step reduction from program $(S, M)$ to $(S', M')$ such that $\text{dom}(S) \subseteq \text{dom}(S')$. 

3
\[(S, \text{case}(x : \text{inl}(V), M_1, M_2)) \rightarrow (S, M_1 [V/x]) \]

\[(S, \text{case}(x : \text{inr}(V), M_1, M_2)) \rightarrow (S, M_2 [V/x]) \]

\[(S, \text{ref}(V)) \rightarrow (S [a' \rightarrow V], a', a' \notin \text{dom}(S)) \]

\[(S, \text{fix}_x S.M) \rightarrow (S, M [\text{fix}_x S.M/x]) \]

\[(S, a = a) \rightarrow (S, 1) \]

\[(S, a = b) \rightarrow (S, 0) \]

\[(S, (\Lambda X.M)(T)) \rightarrow (S, M [T/X]) \]

And finally

\[(S, M) \rightarrow (S', M') \Rightarrow (S, E [M]) \rightarrow (S', E [M']) \]

Where \(E\) is an evaluation context, that is defined just below.

\[
E := (\lambda x.M)_\_ | _\_ N | \pi_1_\_ | _\_ N | (\text{case}(_\_, M_1, M_2)) | \text{inl}(_\_) | \text{inr}(_\_) | \text{fix}_x S.\_ \]

\[
\text{ref} _\_ | _\_ N | V \text{=} _\_ | _\_ N | V \text{=} _\_ | _\_ V := _\_ | _\_ (T) | \text{unfold}_{\mu X.S} _\_ \]

\[
\text{fold}_{\mu X.S} _\_ \]

It is also interesting to note that we do not have the new operator as part of our language. As a result, we cannot create empty references, that are references that have not yet being initialized with a value. This is reflected in our store structure by not allowing names to be part of the domain of the store without being linked to a value of basic type.

3 The nominal sets

As the element we underpin for the creation of a reference is irrelevant, we rely on the notion of names. The nominal sets slightly differ from usual ones in the sense that we add to them the notion of name-permutation. We endow \(A\) with name-permutations \(\text{Perm}(A)\), that are endo-bijections \(\pi : A \xrightarrow{\sim} A\) that have finite support, which means they only permute a finite number of elements, and moreover such that each \(A_b\) and \(N\) remains stable under their action.

**Definition.** The set \(\text{Perm}(A)\) of name-permutations is defined by:

\[
\text{Perm}(A) = \{\pi : A \rightarrow A | \pi \text{ bijective} \}
\]

\[
\land \exists S \subseteq A.(S \text{ fini} \land \forall x.(x \notin S \Rightarrow \pi(x) = x))
\]

\[
\land \forall a \in A.(a \in A_b \Rightarrow \pi(a) \in A_b)
\]

\[
\land \forall a \in A.(a \in N \Rightarrow \pi(a) \in N)
\]

A nominal set \(\hat{X}\) is pair \((X, \bullet)\) where \(X\) is a set, and \(\bullet : \text{Perm}(A) \times X \rightarrow X\) a group action from \(\text{Perm}(A)\) to \(X\) that has finite support (defined below).
Definition. The support $\nu(x)$ of an element $x$ is

$$\nu(x) = \bigcap \{ S \subseteq \text{fin } A \mid \text{For infinitely many } b, \text{ for } a \in S, (a, b) \circ x \neq x \} \bigcap \{ S \subseteq \text{fin } A \mid \forall \pi, (\pi \mid S = \text{id}_S) \Rightarrow \pi \circ x = x \}$$

Where $(a, b)$ denotes the permutation that swaps $a$ and $b$, and $\pi \mid S$ denotes the restriction of $\pi$ to $S$.

That is, $\nu(x)$ is the smallest set $S \subseteq A$ such that, if, $\forall a \in S, \pi(a) = a$ then $\pi \circ x = x$. Here we will be using strong nominal sets, enforcing the previous implication about support to be an equivalence.

Definition. An element of a nominal set has strong support if

$$\exists S, \forall a \in S, \pi(a) = a \Leftrightarrow \pi \circ x = x.$$ 

Of course, in that case, $S = \nu(x)$.

The set $A$ can be considered as a strong nominal set with the natural action from $\text{Perm}(A)$ to it. Restricted to this set, the $\nu$ function acts just as the singleton function, $\nu(x) = \{x\}$. However, as a simple example, the power set of $A$ together with the canonical action is not a strong nominal set, as, for two elements $a, b$ in $A$, the support of $\{a, b\}$ is the set $\{a, b\}$ itself, and $(a, b) \circ \{a, b\} = \{b, a\} = \{a, b\}$ (as there is no ordering in Set). So $(a, b)$ fixes the element, but does not fix the elements of the support. On the other hand, the set $\text{list}(A)$ is a strong nominal set because of the ordering.

Moreover, there is an injection from Set to strong nominal Sets that endows each set with the constant action $id$. In particular, we can consider the constants $C$ as a nominal set, but such that the name permutations have no effect on it. For $i$ a constant element, we have $\nu(i) = \emptyset$.

Two elements of a nominal set $x$ and $y$ are equivalent if there is a permutation $\pi$ that swaps them. For example, two elements of $\text{list}(A)$ are equivalent. However, if we consider list, $\{a, a\}$ is not equivalent to $\{a, b\}$, if $b \neq a$. For $x$ element of a nominal set, we say that $x$ contains the names $a, a \in A$, if $a \in \nu(x)$. In the following, games will take place inside the nominal-set theory, which means each move will contain some names. Therefore, we will have to check all properties relatively to name-invariance, that is under invariance of the action of $\pi$, for $\pi$ a name-permutation.

We can define a category of nominal sets by specifying the concept of nominal function. A function between two sets is nominal if $f \circ \pi(x) = \pi \circ f(x)$ for any $\pi$ in $\text{Perm}(A)$. This definition extends straightforwardly to relations. For two nominal Sets $X$ and $Y$, their disjoint-sum (coproduct) $X + Y$ is well defined and the action of $\pi$ on it is defined canonically, eg $\pi(\text{inl}(x)) = \text{inl}(\pi(x))$.

A subset of a nominal is well defined if and only if it is stable under name-permutations, that is if the injection function is a nominal one.

In the following, all sets will be nominal, as well as all notions related.
notations. Let \( \pi \) be the equivalence class of \( \pi \). Lemma.

Proof. Suppose \( \pi \). Lemma. Strong support lemma: \( \pi \) \( \pi \). Let \( \pi \) \( \pi \) to replace by \( \pi \), making the group action implicit.

Lemma. \( \pi \) commutes with \( \nu \), \( \pi(\nu(x)) = \nu(\pi(x)) \).

Proof. In the following, \( S \) finite will be \( \pi \), and \( \rho \) will be a name permutation.

\[
\pi(\nu(x)) = \pi \bigcap \{ S \mid \forall \pi, (\forall a \in S, \rho(a) = a) \Rightarrow \rho(x) = x \}
\]

\[
= \bigcap \{ \pi(S) \mid \forall \rho, (\forall a \in S, \rho(a) = a) \Rightarrow \rho(x) = x \}
\]

\[
= \bigcap \{ S \mid \forall \rho, (\forall a \in S, \rho(\pi^{-1}(a)) = \pi^{-1}(a)) \Rightarrow \rho(x) = x \}
\]

\[
= \bigcap \{ S \mid \forall \rho, (\forall a \in S, \rho(a) = a) \Rightarrow \rho(\pi(x)) = \pi(x) \}
\]

\[
= \nu(\pi(x))
\]

In particular, if \( \pi(x) = y \), then \( \pi(\nu(x)) = \nu(y) \).

Lemma. Strong support lemma:

Suppose \( X \) a nominal set, and six elements \( x_i, y_i, z_i \), \( i = 1, 2 \) of \( X \) such that

\[
\bot x, y \downarrow = \bot x, y \downarrow
\]

\[
\bot x, z \downarrow = \bot x, z \downarrow
\]

\[
\nu(y_i) \cap \nu(z_i) \subseteq \nu(x_i), i = 1, 2.
\]

then \( \bot x, y, z \downarrow = \bot x, y, z \downarrow \).

Proof. Let \( \pi_1 \) such that \( \pi_1(x_1) = x_2, \pi_1(y_1) = y_2 \), and \( \pi_2 \) such that \( \pi_2(x_1) = (x_2) \) and \( \pi_2(z_1) = z_2 \). Let \( \pi \) such that

\[
\pi \vdash \nu(x_1) = \pi_1 : (\nu(x_1) \to \nu(x_2))
\]

\[
\pi \vdash (\nu(y_1) \setminus \nu(x_1)) = \pi_1 : ((\nu(y_1) \setminus \nu(x_1)) \to (\nu(y_2) \setminus \nu(x_2)))
\]

\[
\pi \vdash (\nu(z_1) \setminus \nu(x_1)) = \pi_2 : ((\nu(z_1) \setminus \nu(x_1)) \to (\nu(z_2) \setminus \nu(x_2))).
\]

Let \( X = \nu(x_1) \cup \nu(y_1) \cup \nu(z_1) \) and \( Y = \nu(x_2) \cup \nu(y_2) \cup \nu(z_2) \). First let us show that \( \pi : X \to Y \) is a bijection and indeed realizes the desired property. This comes from, the fact that \( \nu(y_i) \cap \nu(z_i) \subseteq \nu(x_i) \) so the graph of the three function defined above
does not intersect, and each one of them is a bijection. Moreover, if \( \rho, \rho' \) two name-permutations, a element of a nominal set, such that \( \rho \upharpoonright \nu(a) = \rho' \upharpoonright \nu(a) \) then \( \rho(a) = \rho'(a) \) as \( \rho'^{-1} \circ \rho \upharpoonright \nu(a) = id \) and so \( \rho'^{-1}(\rho(a)) = a \Rightarrow \rho(a) = \rho'(a) \). Now we define \( \pi' \in \text{Perm}(\mathbb{A}) \) by \( \pi' \upharpoonright (\mathbb{A} \setminus (Y \cup X)) = id, \pi' \upharpoonright X = \pi \) and \( \pi' \upharpoonright (Y \setminus X) = \pi^{-1} \). Then as \( \pi' \upharpoonright (\mathbb{A} \setminus (Y \cup X)) = id \) it has finite support, and as \( \pi' \upharpoonright X = \pi \) it gives us the required bijection.

**Lemma.** Let \( X \) a nominal set, \( x, y \in X \) such that \( \nu(x) \cap \nu(y) = \emptyset \). Then \( \exists \pi \in \text{Perm}(\mathbb{A}), \pi \upharpoonright \nu(x) = id \) and \( \pi(\nu(y)) \cap (\nu(x) \cup \nu(y)) = \emptyset \).

**Proof.** For each \( b, \mathbb{A}_b \) is infinite, and \( \nu(x) \cap \mathbb{A}_b, \nu(y) \cap \mathbb{A}_b \) are finite. So for each \( a \in \nu(y) \cap \mathbb{A}_b \) we can pick an new element in \( \mathbb{A}_b \setminus (\nu(x) \cup \nu(y)) \) and then define a injection \( \pi : \nu(y) \hookrightarrow \mathbb{A}_b \). Then, let denote \( X = \nu(x) \cup \nu(Y), Y = \nu(x) \cup \pi(\nu(Y)) = \pi' : X \to Y \) such that \( \pi' \upharpoonright \nu(X) = id \) and \( \pi' \upharpoonright \nu(Y) = \pi \), then we do exactly the same as in the final part of the just below proof to obtain a bijection in \( A_b \). Then we do that over all \( A_b \) (using the axiom of choice in necessary), and obtain the required bijection.

4 The category of games

4.1 The nominal games

**Definition.** A strong nominal arena \( A = \langle M_A, I_A, \lambda_A, \vdash_A, \phi_A, n \rangle \) is given by :

- A strong nominal set of moves \( M_A \)
- A strong nominal set of initial moves \( I_A \), such that \( I_A \subseteq M_A \)
- A nominal labelling function \( \lambda_A : M_A \to \{O, P\} \times \{Q, A\} \)
- A nominal justification relation \( \vdash_A : M_A \times M_A \setminus I_A \)
- An natural number \( n \).

such that

- \( m \in I_A \Rightarrow \lambda_A(m) = (P, A) \)
- \( m \vdash m' \land \lambda_A(m') = A \Rightarrow \lambda_A(m) = Q \)
- \( m \vdash m' \Rightarrow \lambda_A^{O,P}(m) \neq \lambda_A^{O,P}(m') \)

An arena whose sixth element is \( n \) is called an \( n \)-arena, and we the 0-arena are said to be open.

The arenas will correspond to types in our denotational semantics. Therefore, the same we build our set of type out of ground types, we are going to construct a set of ground arenas, and define the whole universe in top of them. We will present five way to combine the arenas with each other. The universe \( U \) will be defined in the last subsection to be the closure of the set of base arenas under the operations below, subject to their restrictions.
4.1.1 Structure of moves

In the following, we will notice that every move that will appear in the arenas that we are going to construct will have some structure induced by the structure of the arena. More precisely, in the universe that we will obtain, each move from any arena will be obtained from two ingredients: patterns and atomic moves \( a \). The atomic moves will come from the ground arenas that we will define. The pattern from the operations that we will define on them.

The set of moves and patterns are defined by mutual-recursive definition as follows.

\[
\text{Definition.} \quad \\
\begin{align*}
\text{• } m_{\text{atomic}} & := (X, n) \mid (i, n) \mid c \mid a \\
\text{where } X & \in TVar, \ n \in \mathbb{N}, \ i \in \mathbb{N}, \ c \in \mathbb{C}, \ \text{and } a \in L
\end{align*}
\]

\[
\begin{align*}
\text{• } m & := m_{\text{atomic}} | p(m) \\
\text{• } p & := _- | \text{inr } p | \text{inl } p | \langle p, m \rangle | \langle m, p \rangle
\end{align*}
\]

Using this structure, we can define projection over a pattern \( p \) to be the function \( m \upharpoonright p = n \) if \( \exists n \) such that \( m = p(n) \) where \( n \) is a move, \( m \upharpoonright p = \epsilon \) otherwise.

Unfortunately, given a move \( m \) where \( (X, n) \) is part of \( m \), there is no canonical instantiation of \( m' \) into \( m \). This comes from the fact that \( (X, n) \) might appear in several places in \( m \), due to the product structure. Therefore, we will define the notion of extended pattern afterwards, in the section dedicated to the system F structure.

4.1.2 The base arenas

We will define some base arenas. We would like to have our arena to correspond to type. We will start by defining the arenas corresponding to variable types.

\[
\text{Definition. For each } X \in TVar \text{ we have a 0-arena } X \text{ such that } M_X = I_X = \{(X, n) \mid n \in \mathbb{N}\}. \text{ The function } \lambda \text{ is specified by the fact that every move is initial, and the enabling relation is empty.}
\]

Of course, we need an arena that represents the natural numbers.

\[
\text{Definition. The closed arena } \mathbb{Z} \text{ has as moves the integers, and they are all initial moves.}
\]

The unit will be represented as an arena with a unique move.

\[
\text{Definition. The arena } 1 \text{ has a unique move } \ast \text{ that is initial.}
\]

And finally, we need an arena to represent the type \( \text{ref } b \), for each ground type \( b \) . They will be defined using the nominal set \( \kappa_b \).

\[
\text{Definition. For each ground type } b, \text{ the arena } \kappa_b \text{ has as moves } M_{\kappa_b} = I_{\kappa_b} = \kappa_b
\]
There is then a straightforward denotational function, denoted \([\_\_]\) from ground types to base arenas. We have \([\text{int}] = \mathbb{Z}\), \([\text{unit}] = 1\), \([\_X] = X\), and finally \([\text{refl}] = A_b\).

In the following, by abuse of notation, we will use the same notation \(X\), for both the types and the arenas, and \(A_b, Z\) for both the set and the arenas.

We furthermore write \(\emptyset\) for the empty arena.

### 4.1.3 Operation on arenas

**Definition.** The closed arena \(A \Rightarrow B\) is defined only in the case \(n = 0\) for both \(A\) and \(B\). In that case, it is defined by

- \(M_{A \Rightarrow B} = \{\_\} \cup M_A \cup M_B\)
  
  A move \(m\) of \(M_{A \Rightarrow B}\) that is not \(\_\) will then be of the form \(\text{inl}(m_A)\) or \(\text{inr}(m_B)\).

- \(I_{A \Rightarrow B} = \{\_\}\)

- \(\lambda_{A \Rightarrow B}(\_\) = \(P\)
  
  \(\lambda_{A \Rightarrow B}(\text{inl}(m))) = \text{OQ}\) if \(m \in I_A\)
  
  \(\lambda_{A \Rightarrow B}(\text{inr}(m))) = \lambda_B(m)\)
  
  where \(\lambda\) is \(\lambda\) with the OP-part reversed. For example, if \(\lambda(m) = \text{OQ}\) then \(\lambda(m) = \text{PQ}\).

- \(\vdash_{A \Rightarrow B} = (\{\_\} \times \text{inl}(I_A)) \cup (\text{inl}(I_A) \times \text{inr}(I_B)) \cup \text{inl}(\vdash_{\_A}) \cup \text{inr}(\vdash_{\_B})\)

- \(n = 0\)

**Definition.** The closed arena \(A \otimes B\) is defined only in the case \(n = 0\) for both \(A\) and \(B\), to be

- \(M_{A \otimes B} = (I_A \times I_B) \cup (M_A \setminus I_A) \cup (M_B \setminus I_B)\)
  
  A move \(m\) of \(M_{A \otimes B}\) will then be of the form \(\langle i_A, i_B \rangle, \text{inl}(m_A)\) or \(\text{inr}(m_B)\).

- \(I_{A \otimes B} = I_A \times I_B\)

- \(\lambda_{A \otimes B}(i_A, i_B) = P\)
  
  \(\lambda_{A \otimes B}(\text{inl}(m)) = \lambda_A(m)\) if \(m \in M_A \setminus I_A\)
  
  \(\lambda_{A \otimes B}(\text{inr}(m)) = \lambda_B(m)\) if \(m \in M_B \setminus I_B\)

- \(\vdash_{A \otimes B} = \{(i_A, i_B) \mid i_A \vdash_{A} m_A\} \cup \{(i_A, i_B), \text{inr}(m_B)\} \cup \text{inl}(\vdash_{\_A} \cap (M_A \setminus I_A, M_A)) \cup \text{inr}(\vdash_{\_B} \cap (M_B \setminus I_B, M_B))\)

- \(n = 0\)

**Definition.** The HOA \(A + B\) is defined only for closed arenas \(A\) and \(B\) by:

- \(M_{A + B} = M_A \cup M_B\)
  
  A move \(m\) of \(M_{A + B}\) will then be under the form \(\text{inl}(m_a)\) or \(\text{inr}(m_B)\).
• \( I_{A \otimes B} = I_A \uplus I_B \)
• \( \lambda_{A+B}(\text{inl}(m)) = \lambda_A(m) \) if \( m \in M_A \)
  \( \lambda_{A+B}(\text{inr}(m)) = \lambda_B(m) \) if \( m \in M_B \)
• \( \vdash_{A+B} = \text{inl}(\vdash_A) \cup \text{inr}(\vdash_B) \)
• \( n = 0 \)

**Definition.** The arena \( \forall X.A \) is defined as follows:

• \( M_{\forall X.A} = \psi(M_A) \) where \( \psi \) is defined by induction by:
  1. \( \psi(X,a) = (n_A, a) \) for any \( (X,a) \) atomic move of \( X \).
  2. if \( m_{\text{atomic}} \) not under the form \( (X,a) \) then \( \psi(m_{\text{atomic}}) = m_{\text{atomic}} \)
  3. \( \psi(\text{inl}(m)) = \text{inl}(\psi(m)) \) \( \psi(\text{inr}(m)) = \text{inr}(\psi(m)) \) \( \psi(m_1,m_2) = (\psi(m_1), \psi(m_2)) \)
• \( I_{\forall X.A} = \psi(I_A) \)
• \( \lambda_{\forall X.A} = \lambda_A \circ \psi^{-1} \)
• \( \vdash_{\forall X.A} = \vdash_A \circ \psi^{-1} \), that is \( a \vdash_{\forall X.A} b \iff \psi^{-1}a \vdash_A \psi^{-1}b \)
• \( n_{\forall X.A} = n_A + 1 \).

**Definition.** The HOA \( \mu X.A \) will be defined in section 4.5 using domain theory. We will only say here that \( n_{\mu X.A} = n_A = 0 \).

**Remark.** In the following, \( \bullet \) will stand for any operation ranging over \( \Rightarrow, \otimes, + \)

Then of course we can extend the denotational function from types to arenas by induction \( [A \bullet B] = [A] \bullet [B] \), \( [\forall X.A] = \forall X.[A] \), and finally \( [\mu X.A] = \mu X.[A] \).

For sake of completeness, we briefly define instantiation now, even if it will be more investigated later. We define it for type, but the translation to arenas is straightforward.

**Definition.** We define instantiation the following way, where \( A \) is an open type.

• \( X [A/X] = A \)
• \( Y [A/X] = Y \)
• \( (\text{unit} \mid \text{int} \mid \text{ref} b) [A/X] = (\text{unit} \mid \text{int} \mid \text{ref} b) \)
• \( (B \bullet B') [A/X] = B [A/X] \bullet B' [A/X] \)
• \( \forall X.B [A/X] = \forall X.B \)
• \( (\forall Y.B) [A/X] = \forall Y.(B [A/X]) \)
• \( \mu X.B [A/X] = \mu X.B \)

10
• \((\mu Y.B)[A/X] = \mu Y.(B[A/X])\)

Lemma.  
• \(n_{B[A/X]} = n_B\).
• Generity principle

\[\forall X.B = \forall Y.B[Y/X]\]

Proof. By induction on the structure of the type.  

4.2 Plays, legal sequences and strategies

Before beginning this section, we will introduce some notations.

• We say that \(s' = s'_1...s'_i\) is a subsequence of \(s = s_1....s_n\) if it exists a growing function \(f : [1...i] \to [1,...,n]\) such that \(s'_i = s'_j(i)\). We write \(s' \subseteq s\) in that case.

• We say that \(s' = s'_1...s'_i\) is a strict subsequence of \(s = s_1....s_n\) if it is a subsequence of \(s\) such that \(f : [1,...,i] \to [1,...,i] = id\). We write \(s' \preceq s\) in that case.

4.2.1 Prearenas

We define a subtle variant of the arena \(A \Rightarrow B\), with two main modifications. The game starts on the left arena, and is opened by opponent. Basically, it boils down to erase the first player move. Moreover, we do not restrict \(A\) and \(B\) to be 0-arenas any-more. An arena in which the opponent starts is called a pre-arena. The notion of closed or open does not make sense anymore in that case.

Definition. The pre-arena \(A \Rightarrow B\) is defined as follow:

• \(M_{A \Rightarrow B} = M_A + M_B\)
• \(I_{A \Rightarrow B} = inl(I_A)\)

\[
\begin{align*}
\lambda_{A \Rightarrow B}(inl(i_A)) &= OQ, i_A \in I_A, \\
\lambda_{A \Rightarrow B}(inl(m_A)) &= \overline{\lambda}_{A}(m_a), m_A \in M_A \setminus I_A \\
\lambda_{A \Rightarrow B}(inr(m_B)) &= \lambda_B(m_b) \\
\vdash_{A \Rightarrow B} &= \text{inl}(\vdash_A) \cup \text{inr}(\vdash_B) \cup \{(\text{inl}(i_A), \text{inr}(i_B)) \mid i_A \in I_A \land i_b \in I_B\} \\
\end{align*}
\]

We denote \(A \Rightarrow B \Rightarrow C = A \Rightarrow (B \Rightarrow C)\). For \(s\) a sequence in \(A_1 \Rightarrow A_2.... \Rightarrow A_n\), \(s \mid A_iA_j\) is the subsequence of \(s\) whose move are in either \(A_i\) or \(A_j\), where the initial move of \(A_j\) is relabelled in answer. We furthermore looked at it as injected in \(Ai \Rightarrow A_j\), by doing the necessary adaptations. Explicitly, for \(m\) in \(A_i\), \(\lambda_{A_i,A_j}(s) = \overline{\lambda}_{A_i}(s)\), and \(\lambda_{A_i,A_j}(s) = \lambda_{A_j}(s)\) if \(m\) in \(A_j\). Furthermore, we add a justification pointer from the initial move \(s_{A_i}\) of \(A_i\) to the one of \(A_j\), \(s_{A_j}\), and keep the pointers of \(A_i\) and \(A_j\) as they were in the original sequence. With these modifications, \(s_{A_i,A_j} \in A_i \Rightarrow A_j\).
4.2.2 Move with store

In the future, we will adjoin a store to each move, that will represent a snapshot of the memory at runtime. The definition of the store is almost the same as the one presented in the operational semantics. For sake of completeness, we remind it here. The store is written $S$ and is an element of the following set

$$\{ f \in \mathcal{A} \rightarrow \text{partial } V_{\text{basic}} \mid a \in \mathcal{A}_b \Rightarrow f(a) \text{ of type } b \land f \text{ finite} \},$$

where the set $V_{\text{basic}}$ of values of basic types is defined by

$$V_{\text{basic}} ::= \_ | c | l | ()$$

where $c \in \mathbb{C}$ and $l \in \mathbb{L}$. Note that the store is not a (partial) nominal function.

The names-permutations act canonically on the store, that is if $(a,v)$ is in $S$ then, $(\pi \bullet a, \pi \bullet v)$ will be a couple of the graph of $\pi \bullet S$. A sequence with store is a sequence of moves such that each of them have an associated store, and that moreover satisfies the following growing condition:

$$\text{if } s'm^S_m n^S_n \text{ is a subsequence of } s, \text{ then } \text{dom}(S_m) \subseteq \text{dom}(S_n).$$

We are not interested in the ability to erase something from the memory. The inability of programs to access local variables once they are out of their scope will be further implemented as part of the semantics of composition.

In the following we will write $m$ for $m^S_m$ a move with store, writing the store explicitly only when required. In the opposite direction, for a move with store $m$, when we will want to emphasize move and exclude the store, we will write $\overline{m}$.

4.2.3 Plays

**Definition.** A legal sequence $l$ over a pre-arena $A \rightarrow B$ is a sequence of moves $(m_{A \rightarrow B})$ from $M_{A \rightarrow B}$ subject to certain conditions.

- (Alternating) For every two consecutive moves $m,n$ of $l$ the polarity has to be different: $\lambda_{O,P}(m) \neq \lambda_{O,P}(n)$
- (Well-opened) The first move $i_A$ of the sequence belongs to $I_A$ and it is the only one: $l \cap I_A = i_A$.
- (Justified) Every move, expect the first one, has to be enabled by a move before. For any move $n$ in $l$, $n$ not in first position, $\exists m \in l$, such that $m$ appears before $n$ in $l$, and $m \vdash_A n$. We say that $m$ is the justifier of $n$.
- (Well bracketed) If $m_a$ answers a question $m_q$, then $m_q$ should be the last unanswered question that appears in $l$. 

12
• (Visibility) Let \( s = s'.m.n \) a subsequence of \( l \). Then the justifier of \( n \) should be in \( \Gamma(s'.m)^n \) where the \( \Gamma \) denotes the view function: \( \Gamma_e = e \mid \Gamma_i_A = i_A \mid \Gamma_sms'n = \Gamma_s'mn \) if \( m \) justifies \( n \).

• (Variable Freshness) In the following, \( \chi \) will stand for "either \( X \) a Type variable or \( i \) an integer". If \( p(\chi,a) \in l \land p'(\chi',b) \in l \) and \( (\chi,a) \neq (\chi',b) \), where \( a,b \in \mathbb{N} \), then \( a \neq b \). In other term, if a name \( a \in \mathbb{N} \) is played in \( l \) once with \( \chi \), then it can only be played coupled with \( \chi \) afterwards.

• (Left copycat for closed variables) For the sequence \( l_{\text{left}} = l^{\uparrow_{\text{ind}}} \) opponent can play \( p(i,a) \) only if player played \( p'(i,a) \) before, where \( i \in \mathbb{Z} \) and \( a \in \mathbb{N} \).

• (Right copycat for closed variables) For the sequence \( l_{\text{right}} = l^{\uparrow_{\text{inr}}} \) player can play \( p(i,a) \) only if opponent played \( p'(i,a) \) before, where \( i \in \mathbb{Z} \) and \( a \in \mathbb{N} \).

• (Copycat for open variables) Player can play \( p(X,a) \) only if opponent played \( p'(X,a) \) before, where \( X \in TVar \) and \( a \in \mathbb{N} \).

The first four conditions are the standard ones for legal sequences in game semantics for call-by-value programming language. In our case, we will need three more, stating who is introducing names in \( X \). The global idea is basically that player can only copycat in \( X \), as he does not know more about \( X \) that what opponent reveals to him, that is, he can not introduce new names. Of course, for the left side of the arena the polarity is reversed. Two type variables played with the same variable name distinguisher is equivalent to a copy-cat links in Laird papers on polymorphism. The variable freshness condition states that a there is no possible copycat-links between two different type variables.

Following (some quote missing,...), we would like to add some conditions on the store. The store must contain only names that have been played so far, and each name that have been played must be contained in the store. This condition is named frugality. For technical reasons that will be highlighted while studying the composition of strategies, we have to differentiate legal sequences that satisfies this conditions from the other ones. Such legal sequence will be called played.

**Definition.** The set of revealed or global names is defined by induction as follows

\[
\text{Glog}(\epsilon^e) = \epsilon \quad \text{Glog}(s'.m^S) = \cup_{i \in \mathbb{N}} (\nu \circ S)^i(G\text{log}(s') \cup \nu(m)) \cap \mathbb{L}
\]

where \((\nu \circ S)^0\) is the identity.

We do not consider the names using for variables distinguishers, that are those in \( \mathbb{N} \) when we speak about global names.

**Definition.** A play is a legal sequence \( l \) that moreover satisfies the frugality condition

• (Frugality) For any sub-sequences \( s = s'm^S \in l \) we have \( \text{dom}(S) = \text{Glog}(s) \).

We define by \( \mathcal{P}_A \) the set of plays in a pre-arena \( A \).
Definition. A (nominal) strategy $\sigma$ is a subset of even-length sequences of $P_A$ such that

- (prefix-closure) if $\exists smn, smn \in \sigma \Rightarrow s \in \sigma$
- ($\sim$-saturation) if $s \in \sigma$, and $s \sim t$ then $t \in \sigma$.
- ($\sim$-determinacy) if $sm \in \sigma \land s'm' \in \sigma \land s \sim s'$ then $sm \sim s'm'$.

There is a really simple way to turn a legal sequence into a play, we just need to restrict the domain of store function to the names that have appear so far. Moreover, if a name is played without appearing in the store, we add it, mapping it to nothing.

Definition. The garbage collector function $\gamma$ is defined by induction from the legal store to plays by:

$$\gamma(s.m^S_n) = \gamma(s).m^{S'}_n$$

where $S'_i = S_i \setminus \text{glob}(\gamma(s).S_i)$.

Then, for any legal sequence $s$ such that $\text{glob}(s) \subseteq \text{dom}(S)$, $\gamma(s)$ satisfies the frugality condition and is therefore a play. In the following, we will denote by $s \mid_\gamma A$ the sequence $\gamma(s \mid A)$, for $A$ an (pre)-arena. We introduce some more notation in the following definition.

Definition. Concatenation of move/sequence with store

For two moves $a^S_a$ and $b^S_b$, we denote $a \circ b$ the sequence $a^S_a b^S_b \cup (\text{dom}(S_a) \setminus \text{dom}(S_b))$.

For two sequences $s = s_1...s_m$ and $t = t_1...t_n$ we define $s \circ t$ by induction:

$$s \circ \epsilon = s \mid s \circ t_1 = s_1...s_{m-1}.(s_m \circ t_1) \mid s \circ (t_1.t') = (s \circ t_1) \circ t'$$

For a sequence $t$, we define $\circ t = \epsilon \circ t$

Note that this turns every sequence of move with store into one satisfying the store-growing condition.

4.3 Category of Games

We would like to define a category with arenas as objects, and strategies in the pre-arena $A \to B$ as morphisms. Hence, the composition between strategies has to be defined, and, more generally, the composition of plays, from which the composition of strategies will follow straightforwardly.

Definition. We denote $O$ and $P$ the set of names introduce by respectively the opponent and the player.

$O(\epsilon) = \epsilon$

$O(sm) = O(s) \cup \nu(m)$ if $m$ played by the opponent, or $O(s)$ otherwise

$P(s) = \nu(s) \setminus O(s)$
Definition. The set of interaction sequences $I_{ABC}$ of $A \rightarrow B \rightarrow C$ is a subset of the set of justified and frugal sequences on $(A \rightarrow B) \rightarrow C$ that satisfy

- $s \rightharpoonup \gamma A \rightarrow B \in \mathcal{P}_{A \rightarrow B} \land s \rightharpoonup \gamma B \rightarrow C \in \mathcal{P}_{B \rightarrow C} \land s \rightharpoonup \gamma A \rightarrow C \in \mathcal{P}_{A \rightarrow C}$

- name-flow condition 1 $P(s \rightharpoonup \gamma A \rightarrow B) \cap P(s \rightharpoonup \gamma B \rightarrow C) = \emptyset$

- name-flow condition 2 $O(s \rightharpoonup \gamma AC) \cap (P(s \rightharpoonup \gamma A \rightarrow B) \cup P(s \rightharpoonup \gamma B \rightarrow C)) = \emptyset$

- store condition if $t.m_{S.nS}$ subsequence of $s$, and if $n$ is a
  1. player move in $A \rightarrow B$ then $S' \rightharpoonup \nu(s) \setminus \nu(\gamma(s \rightharpoonup A \rightarrow B)) = S \rightharpoonup \nu(s) \setminus \nu(\gamma(s \rightharpoonup A \rightarrow B))$
  2. player move in $B \rightarrow C$ then $S' \rightharpoonup \nu(s) \setminus \nu(\gamma(s \rightharpoonup B \rightarrow C)) = S \rightharpoonup \nu(s) \setminus \nu(\gamma(s \rightharpoonup B \rightarrow C))$
  3. opponent move in $A \rightarrow C$ then $S' \rightharpoonup \nu(s) \setminus \nu(\gamma(s \rightharpoonup A \rightarrow C)) = S \rightharpoonup \nu(s) \setminus \nu(\gamma(s \rightharpoonup A \rightarrow C))$

Using that, we define the set of interaction sequences with respect to two strategies; given $\sigma : A \rightarrow B$ and $\tau : B \rightarrow C$, we define $\sigma \mid \tau$ in $I_{ABC}$ as follows:

$$\sigma \mid \tau = \{ s \in I_{ABC}, s \rightharpoonup \gamma A \rightarrow B \in \sigma, s \rightharpoonup \gamma B \rightarrow C \in \tau \}.$$

From that, the composition $\sigma; \tau$ turns to be

$$\sigma; \tau = \{ s \in \mathcal{P}_{A \rightarrow C} | \exists t \in \sigma | \tau, \gamma(t \rightharpoonup A \rightarrow C) = s \}.$$

We need to prove that these definition indeed make sense and that the arenas together with such morphisms indeed forms a category, that is to check that composition of strategies yield to strategy, and that composition is associative.

Theorem 1. The composition of two strategies is a strategy.

For that we need a few lemmas, that will be proved along the lines.

Lemma. 1. Let $s \in \sigma \mid \tau$ an interaction sequence. Then either $s$ terminates in $A$ by a $P$-move or terminates in $C$ by an $O$-move.

2. Only $P$ can move from $A$ to $B$ and reversely and only $O$ from $B$ to $C$ and reversely.

Proof. The last move of $s$ has to be the last move of a sequence that belongs to $\sigma$ or $\tau$. Suppose $\sigma$, then either it is a $P$-move in $A$, or it is a $P$-move in $B$. But then, this move is an $O$-move for $\tau$, that therefore has to answer it. So this can not be the last move of the interaction sequence. So it terminates either in $A$ with a $P$-move for $\sigma$, or ends in $C$ with a $P-\text{move}$ from $\tau$, that is an $O$-move relatively from the whole sequence.

The second part comes from the following diagramm. The nodes represent who played last in $A, B$), and the edge the move that is going to happen. The underlined part highlight where the last move was. For example $(Q,P)$ means that the last move in $A$ was
by O, the last move in B was by P, and the last move occurred in the arena A. \((O, O)\) means equivalently that the last move in the right hand side was by O or that there was no move in the arena B. \(\text{change}_p\) denotes a move by \(P\) that happens in a different part of the arena than the move preceding it. We can see that the only moves that change from arena to another are by P. As in the part \(B \to C\) the polarity is reversed, only \(O\) will be able to make the change on this side of the interaction sequence.

\[
\begin{array}{c}
\text{stay}_O \quad \xymatrix{ & (O, O) \ar[dl]^{\text{change}_p} \ar[dr]_{\text{stay}_O} & } \\
(P, O) & & (O, P) \\
\text{change}_p \quad \xymatrix{ & (O, O) \ar[dl]^{\text{change}_p} \ar[dr]_{\text{stay}_P} & } \\
& (P, O) & \\
\end{array}
\]

\[\square\]

**Lemma.** For an interaction sequence \(s \in I_{A,B,C}\), we have the following equality.

\[
\nu(s) = P(s \mid_{\gamma} AB) \uplus P(s \mid_{\gamma} BC) \uplus O(s \mid_{\gamma} AC)
\]

**Proof.** We already know that each two of these three sets have empty intersection. Furthermore, by splitting the sequence \(s\) into the \(AB\) and the \(BC\) part, we know that \(\nu(s) \subseteq P(s \mid_{\gamma} AB) \cup O(s \mid_{\gamma} AB) \cup P(s \mid_{\gamma} BC) \cup O(s \mid_{\gamma} BC)\). So we just need to prove that \(x \in O(s \mid_{\gamma} AB)\) or \(x \in O(s \mid_{\gamma} BC)\) then \(x \in P(s \mid_{\gamma} AB) \cup P(s \mid_{\gamma} BC) \cup O(s \mid_{\gamma} AC)\). We will prove that in the case \(x \in O(s \mid_{\gamma} AB)\), the other case being proven along the same lines. Suppose \(x\) have been played before in \(C\), then it is either by opponent \((x \in O(s \mid_{\gamma} AC))\), or by player \((x \in P(s \mid_{\gamma} BC))\). So the \(C\) case is done. Suppose now, the first time \(x\) appears in the sequence is in \(AB\), then by definition it is by opponent. If it appears first in \(B\), we have \(x \in P(s \mid BC)\), and in \(A\) it entails \(x \in O(s \mid AC)\), which ends the proof. \[\square\]

We are going to prove carefully the four key properties for \(\sigma; \tau\) being a strategy.

**Proof.**

- **prefix-closure** Let \(smt'n \in \sigma; \tau\) then there exists \(t, t'\) such that \(tmt'n \in \sigma \mid \tau\) and \(smt'n \mid_{\gamma} A \to C = smn\). We can see that two consecutive moves of an interaction sequence belongs either to \(B \to C\) or to \(A \to B\). Indeed, say \(m\) is an O-move in \(C\), then as \(s \mid_{\gamma} B \to C \in \tau\), P will answer in \(B\) or \(C\). Say that \(m\) belongs in \(A\) and \(n\) in \(C\). Then, \(m\) has to be an O-move (cf preceeding lemma), and \(n\) as well, as the sequence \(t'\) belongs in \(B\), and then begins by a P-move. By using the prefix closure of \(\tau\), we can erase moves two by two in the part \(t'n\), until there is nothing left. Then, as \(m\) is an O-move, and only P can change, it means that the move preceeding \(m\) (if there is one) is a P-move in \(A\). So we indeed have \(t \mid_{\gamma} A \to B \in \sigma\) and \(t \mid_{\gamma} B \to C \in \tau\) by prefix closure of \(\sigma\) and \(\tau\). So \(t\) is indeed an interaction sequence such that \(t \mid_{\gamma} A \to C = s\), so \(s \in \sigma; \tau\).
• **saturation** Let $s \in \sigma; \tau$, $\pi$ name-permutation. Then, let $s' \in \sigma | \tau$ such that $s = s' |\gamma A \rightarrow C$. Then $\pi s' \in \sigma | \tau$ by saturation of $\sigma$ and $\tau$, and as the set intersections are preserved under a bijection, the properties about the name flow are preserved. Therefore $\pi(s) = \pi(s') |\gamma A \rightarrow C$ is in $\sigma; \tau$.

• **determinacy** This is the most difficult property to prove. We will rest on the strong support lemma, with the notation used in it. Let $\uplus s_1 = \uplus t_2$ two plays of $A \rightarrow C$, such that both belong to $\sigma; \tau$. Then let $s'$ be an interaction sequence for $s$, and $t' = \pi(s')$ be an interaction sequence for $t$. There are three cases to consider.

1. Suppose $a,b$ are P-moves in $A \rightarrow B$. For $X$ an arena let denote by $S_X$ (resp. $S_X'$) the part of the store of $a$ (resp $b$) that corresponds to $sa$ (resp $tb$) |$\gamma X$. By $\sigma$ saturation $\uplus s' |\gamma A \rightarrow B, a^{S_{AB}}_\gamma = \uplus t' |\gamma A \rightarrow B, b^{S_{AB}}_\gamma$. Moreover, $\uplus s' |\gamma A \rightarrow B, s' \rightarrow C_{\uplus j} = \uplus t' |\gamma A \rightarrow B, t' \rightarrow C_{\uplus j}$. In addition, suppose $x \in \nu(a^{S_{AB}}) \cap \nu(s' |\gamma B \rightarrow C)$. We have to show $x \in \nu(s' |\gamma A \rightarrow B).$ If this is not the case, then $x$ just appears in $C$, that means $x \in O(s' |\gamma A \rightarrow C) \cup P(s' |\gamma B \rightarrow C)$. But contradiction, as $x \in P(s' a^{S_{AB}} |\gamma A \rightarrow B)$. Furthermore, as $a$ and $b$ are P-move in $A \rightarrow B$, they do not change the part of the store that consists of name just appearing in $C$. Let call $SC$ this part (caution, $SC \neq S_C$, as names may flow from $A,B$ to $C$). But this part is under the $B \rightarrow C$ bijection, that maps $s' |\gamma B \rightarrow C$ to $t' |\gamma B \rightarrow C$. So by the strong support lemma $\uplus s' |\gamma A \rightarrow B, s' \rightarrow C_{\uplus j}, \nu(SC), a^{S_{AB}}_\gamma = \uplus t' |\gamma A \rightarrow B, t' \rightarrow C, \nu(SC')$, $b^{S_{AB}}_\gamma$ or more simply $\uplus s', a_{\uplus j} = \uplus t', b_{\uplus j}$ that implies $\uplus s, a_{\uplus j} = \uplus t, b_{\uplus j}$.

2. for $a,b$ P-moves in $B \rightarrow C$, the proof goes the same way but this time using $\nu(s' |\gamma B \rightarrow C)$ (resp $\nu(t' |\gamma B \rightarrow C)$) as $X_1$ (resp $X_2$), and $\nu(s' |\gamma A \rightarrow B)$ (resp $\nu(t' |\gamma A \rightarrow B)$) as $Y_1$ (resp $Y_2$).

3. for $a,b$ O-moves in $A \rightarrow C$, we take $X_1 = \nu(s' |\gamma A \rightarrow C), Y_1 = \nu(s'), X_2 = \nu(t' |\gamma A \rightarrow C), Y_2 = \nu(t')$. Then we have to show that $\nu(a^{S_{AC}}) \cap Y_1 \subseteq X_1$. Suppose this is not the case, then $\exists x \in \nu(a^{S_{AC}})$, $x$ only appears in $s' |\gamma B$. That means, $x \in P(s' |\gamma A \rightarrow B) \cap x \in P(s' |\gamma B \rightarrow C)$. But $x \in O(s' a^{S_{AC}} |\gamma A \rightarrow C)$, so this is a contradiction. We end the proof just as in the first case.

• **frugality** We know that for each sequence $s' a^{S_A}$, terminating in either $A$ or $C$, we have $glob(s' a) = dom(S_a)$, and in particular, $glob(s' a^{S_A} | AC) \subseteq dom(S_a)$. Then frugality follows from the fact that $\gamma$ transforms any such sequence into a frugal one.

We will now discuss the associativity of composition.

**Definition.** We define two sets of enlarged interaction sequences.

- $(\sigma | \tau) | \rho = \{s \in P((A\rightarrow B)\rightarrow C)\rightarrow D | s |\gamma A, B, C \in \sigma | \tau, s |\gamma A, C, D \in (\sigma; \tau) | \rho\}$
Lemma.
\begin{itemize}
\item \( \sigma \mid \tau \mid \rho = \{ s \in P(A \rightarrow B) \rightarrow (C \rightarrow D) \mid s \mid \gamma B, C, D \in \tau \mid \rho, s \mid \gamma A, B, D \in \sigma \mid (\tau ; \rho) \} \)
\end{itemize}

\textbf{Proof.} We will prove only the first point, the second being symmetrical.

From right to left is by definition. For \( s \) such that \( s \mid \gamma A, C, D \in (\sigma ; \tau) \mid \rho \) then, by definition, \( s \mid A, D \in (\sigma ; \tau) ; \rho \).

Let \( s \in (\sigma ; \tau) ; \rho \). Then \( \exists s' \in (\sigma ; \tau) \mid \rho \) such that \( s = s' \mid \gamma A \rightarrow D \). Now, we know that \( s' \mid A \rightarrow C \in \sigma ; \tau \). So \( \exists t \in \sigma \mid \tau \) such that \( s' \mid \gamma A \rightarrow C = t \mid \gamma A \rightarrow C \).

Note that when we are going to interleave these sequences with each other, a problem might appears in the store part. Indeed, even if the \( AC \) part of the store will be the same for the moves in \( t \) and \( s \) that are supposed to correspond, the other part might be conflictual. Therefore, we want to enforce that \( (\nu(t) \setminus \nu(t \mid \gamma AC)) \cap (\nu(s \mid \gamma) \setminus \nu(s \mid \gamma AC)) = \emptyset \), that is, the ‘other parts’ of the store of \( s \) and \( t \) being disjoint. Then, two sequences \( t \) and \( s \) that satisfies the property, it is possible to form a interaction sequence in \( (\sigma \mid \tau) \mid \rho \). This will be proved along the next lemmas. For now, let us assume this to be true.

For now on, we will consider a sequence \( s \in (\sigma \mid \tau) \mid \rho \). In order to show that the two way of composing are equal, we are going to show some intermediate lemmas. Our aim will be to prove the following lemma, which will immediately entails the theorem.

\textbf{Lemma.} \( \forall s \in (\sigma \mid \tau) \mid \rho, \exists t \in (\sigma \mid \tau \mid \rho) \) such that \( s \mid \gamma A \rightarrow C = t \mid \gamma A \rightarrow C \), and \( s \mid \gamma A \rightarrow D = t \mid \gamma A \rightarrow D \).

It is interesting to thing about what might prevent a sequence in the first set to be part of the second. These are mainly the names flow conditions, and this lemma investigate it. In order to shorten, we will denote \( P_s(AB) = \nu(P(s \mid \gamma A \rightarrow B)) \), and equivalently for \( O_s \) instead of \( P, C \) and \( D \) instead of \( A \) and \( B \).

Let sum up the properties we have about names in \( (\sigma \mid \tau) \mid \rho \).

\begin{itemize}
\item \( P_s(AB) \cap P_s(BC) = \emptyset \)
\item \( O_s(AC) \cap (P_s(AB) \cup P_s(BC)) = \emptyset \)
\item \( P_s(AC) \cap P_s(CD) = \emptyset \)
\item \( O_s(AD) \cap (P_s(AC) \cup P_s(CD)) = \emptyset \)
\end{itemize}

Now, we would like to enforce \( s \mid \gamma ABD \in I_{ABD} \) and \( s \mid \gamma BCD \in I_{BCD} \)

\textbf{Lemma.} Let \( s \in (\sigma \mid \tau) \mid \rho \).

Let \( E_1 = (P_s(BC) \cap P_s(sCD) \cup P_s(BC) \cap O_s(BD) \cup (P_s(CD) \cap (O_s(BD)))) \)

and \( E_2 = (P_s(AB) \cap P_s(BD) \cup P_s(AB) \cap O_s(AD) \cup (P_s(BD) \cap (O_s(AD)))) \)

Then \( E_1 \cap \nu(s \mid \gamma AC) = \emptyset \).

And \( E_2 \cap \nu(s \mid \gamma AC) = \emptyset \).

18
Proof. We proof only for $E_1$, but the proof for $E_2$ goes along the same lines. Let $x$ be in $\nu(s \upharpoonright_\gamma A \rightarrow C) \cap E$. As $x \in \nu(AC)$, $x$ belongs either in $P(AC)$ or in $O(AC)$. Suppose $x$ in $P(AC)$, that implies $x \notin P(CD)$. So $E_1 \cap \nu(s \upharpoonright_\gamma AC) \subseteq P(BC) \cap O_\delta(BD)$. We are going to do a case study depending on where $x$ appears the first time in the whole interaction sequence $s$.

$x$ appears first in :

- **A**: By opponent : $x \in P_\delta(AB) \Rightarrow x \notin P_\delta(BC) \Rightarrow x \notin E$.
- **B**: $x \in P_\delta(BC)$ implies that opponent introduces it, whereas $x \in O_\delta(BD)$ says it is player. So this case is not possible.
- **D**: By opponent : $x \in O_\delta(AD) \Rightarrow x \notin P_\delta(AC)$; contradiction. By player $\Rightarrow x \in P_\delta(CD)$ : impossible.
- **C**: There is three cases to analyse, depending on in which arena it will appear next. If in $A$, then either it is by opponent, in which case $x \in P_\delta(AB) \Rightarrow x \notin P_\delta(BC) \Rightarrow x \notin E$. So suppose player introduced it in $A$, then $x \in O_\delta(AD)$, so $x \notin P_\delta(AC)$, contradiction. So let us examine the $B$ case. As $x \in O_\delta(BC)$ it has to be introduced by player in $B$. So $x \in P_\delta(AB)$ : impossible as $x \in P_\delta(BC)$. It remains only the case where $x$ appears in second in $D$. Then, as $x \in O_\delta(BC)$ it has to be played there by opponent. So $x \in O_\delta(AD)$ which leads, as we have seen, to a contradiction. Let us finish by noticing that if $x$ just appears in $C$, it will not belongs to $O_\delta(BD)$, and so is not in $E$.

Now we do the same in the case $x \in O(AC)$, obtaining this time $E_1 \cap \nu(AC) \subseteq O(BD) \cap P(CD)$. $x$ appears first in

- **A**: then $x \in O(AD) \Rightarrow x \notin P(CD) \Rightarrow x \notin E_1$
- **B**: then it is by player as $x \in O(BD)$ so $x \in P(AB) \Rightarrow x \notin O(AC)$ contradiction
- **D**: As $x \in O(BD)$, it also has to be that $x \in O(AD) \Rightarrow x \notin P(CD)$.
- **C**: then we need to do a second case analysis, depending on where it will appear next. It has to appear in $B$ or $D$ otherwise $O(BD) = \emptyset$. Wherever it is, the proofs (for the A,B,D cases) just above works, even if $x$ was played in $C$ first.

\[\Box\]

So we know that the names that are going to prevent the whole sequence $s$ to belong $s; (\tau; \rho)$ will only appear in $B$ and $D$. Actually, this is the whole point of the problem. The name-flow condition in an interaction sequence precisely say that if a name appears in the external part, then either it has to be the opponent who plays it, either it has to flow through the internal part. Here, in the whole sequence $ABCD$, the opponent can only play in $A$ and $D$. Then every names that appears in $B$ either has to comes from $A$, or from $D$, in which case in has to go through $C$. So $(\nu(s \upharpoonright_\gamma ABC) \setminus \nu(s \upharpoonright_\gamma AC)$
and $\nu(s \upharpoonright D)$ should be empty. Let us also note that $E_1 \subseteq \nu(s \upharpoonright \gamma \ BCD) \setminus \nu(s \upharpoonright AC)$, and $E_2 \subseteq \nu(s \upharpoonright \gamma \ ABD) \setminus \nu(s \upharpoonright AC)$, so Let call $E$ this set, and let us prove that it is the right set to consider.

**Lemma.** Let $x$ in $E_1$ (resp $E_2$), then $x$ appears both in $B$ and $D$.

**Proof.** Knowing that $x \notin \nu(s \upharpoonright \gamma \ AC)$, and looking at the structure of $E_1$ gives us directly the answer. For example, let us say $x \in P_s(BC) \cap P_s(CD)$, then as $x$ does not appears in $C$, it has to appears both in $C$ and $D$. For $P_s(BC) \cap O_s(\text{BD})$, the argument is if $x$ is played only in $B$, then $x \in P_s(BC) \Rightarrow x \in P_s(BD)$. So $x$ has to be played by opponent in $D$. And so on and so forth.

As a result, we know that

$$E_1 \cup E_2 \subseteq (\nu(s \upharpoonright \gamma \ ABC) \setminus \nu(s \upharpoonright \gamma \ AC)) \cap \nu(s \upharpoonright \gamma \ D) = E,$$

and introduce $E$ by the same occasion.

**Lemma.** Let $s (\sigma; \tau) \mid \rho$, and $s' \in \sigma \mid \tau$, such that $s \upharpoonright \gamma \ AC = s' \upharpoonright \gamma \ AC$ and $(\nu(s' \upharpoonright \gamma \ ABC) \setminus \nu(s \upharpoonright \gamma \ D)) \cap \nu(s \upharpoonright \gamma \ D) = \emptyset$, then we can construct a sequence $t = (s' \mid s \upharpoonright CD) \in (\sigma \mid \tau) \mid \rho$ such that $t \upharpoonright \gamma \ ABC = s'$ and $t \upharpoonright ACD = s$ (resp $t \upharpoonright ABD = s$).

**Remark.** Note that the name $t = (s' \mid s \upharpoonright CD)$ is actually inducing the reader in mistake, letting think that we only look at the CD part of $S$. This is actually false, as some names of $D$ might appear in the store when $s$ is playing in $ABC$. If we do not really care about the part of the play in $BC$, as the interaction sequence will prevent the sequence from modifying their value in this part, they might be in $A$.

**Proof.** We prove the lemma by constructing the sequence $t$. We can interleave the two interaction sequences $s'$ and $\bar{s}$ such that the interaction sequence $s'$ is included inside the $\bar{s}$ one. Basically, if $s_i$ in $A$ or $C$ and $s_i+1$ in $A$ or $C$, $t'_k$ move corresponding to $s_i$, $t'_{k+1}$ the move corresponding to $s'_i+1$ (so $s_i = s'_i \land s_i+1 = s'_i+1$), the new sequence $t$ is going to be of the form $\ldots s_i \cup s'_i \cdots s_{i+1} \cup s'_{i+1}, \ldots$, with the necessary change of justification pointer associated. Now, we adjoin a store to each move $t_i$ by associating the store of its corresponding move $s_i$ or $s'_i$, and the union of the both whenever $t_i$ is in $A$ or $C$. Re-using the previous notation, the moves with store will have the following structure $\ldots s_i \cup s'_i \cdots s_{i+1} \cup s'_{i+1}$, whenever $St_i \cap \text{dom}(S_1) \cap \text{dom}(S_2) = S_2 \cap \text{dom}(S_1) \cap \text{dom}(S_2)$.

Here, the store conditions allowing us for creating the union of the two stores are clearly satisfied, as if a name appears both in $S_{s_i}$ and in $S'_{s_i}$ then it is in $\nu(s \upharpoonright \gamma \ AC) = \nu(s' \upharpoonright \gamma \ AC)$, and $s \upharpoonright \gamma \ AC = s' \upharpoonright \gamma \ AC$, so in particular $S_{s_i} \upharpoonright \gamma \ AC = S_{s'_i} \upharpoonright \gamma \ AC$.

Now, in order to satisfies the growing condition, we take $ct$, and the obtained sequence satisfies the lemma.

Another totally equivalent way to construct it is by considering two slightly modified sequences $s$ and $s'$, such that their labeling corresponds to $t$. :
Lemma. Let \( \pi \) and \( \sigma \) be sequence in \( \Sigma = (\nu(s \mid \gamma ABC)) \cap \nu(t \mid \gamma D) = \emptyset \).

\[
S_i = \{ s_j \mid j = \max_{j \in \mathbb{N}, j \leq i} \text{ such that } s_j \in t \} \\
S'_i = \{ s'_j \mid j = \max_{j \in \mathbb{N}, j \leq i} \text{ such that } s'_j \in t \}
\]

We construct the store sequence of \( t \) by induction:

\[
S(t_i) = S(s'_i) \cup (S(s_i) \setminus (\nu(\text{dom}(S(s'_i))) \setminus \nu(\text{dom}(S(s'_i)))))) \text{ if } s_i \text{ in } A, C \\
S(t_i) = S(s'_i) \cup (S(t_{i-1}) \setminus (\nu(\text{dom}(S(s'_i))) \setminus \nu(\text{dom}(S(s'_i)))))) \text{ if } s_i \text{ in } B \\
= S(s_i) \cup (S(s'_i) \setminus (\nu(\text{dom}(S(s'_i))) \setminus \nu(\text{dom}(S(s'_i)))))) \text{ if } s_i \text{ in } C,D
\]

\[\square\]

**Lemma.** For \( s, s' \) as before, we have \( s' \mid (s \mid CD) \in (\sigma \mid \tau) \mid \rho \).

**Proof.** \( t \mid ACD = s \) and \( s \in (\sigma \mid \tau) \mid \rho \), and \( t \mid ABC = s' \) and \( s' \in \sigma \mid \rho \).

This proves the second part of the lemma...

Now our aim is to prove that given any sequence in \( (\sigma \mid \tau) \mid \rho \), we can transform it into a sequence in \( \sigma \mid (\tau \mid \rho) \).

**Lemma.** Let \( s \in (\sigma \mid \tau) \mid \rho \). Let \( \pi \) be a name permutation such that, \( \pi \mid \gamma E^c \cap \nu(s) = \text{id} \) and \( \pi(E) \cap (E^c \cap \nu(s)) = \emptyset \). Let \( s' = \pi(s \mid \gamma ABC) \).

As \( s, s' \) satisfies the condition of the previous lemma, we can form the sequence \( t = s' \mid s \mid CD \). Then we have \( t \mid BCD \in I_{BCD} \)

**Proof.** As a first step, we need to prove that \( \Sigma = (\nu(t \mid \gamma ABC) \setminus \nu(t \mid \gamma AC)) \cap \nu(t \mid \gamma D) = \emptyset \).

\[
\Sigma = (\nu(\pi s \mid \gamma ABC) \setminus \nu(\pi s \mid \gamma AC)) \cap \nu(s \mid \gamma D) \\
= \pi(\nu(s \mid \gamma ABC) \setminus \nu(s \mid \gamma AC)) \cap \nu(s \mid \gamma D) \\
= (\pi(\nu(s \mid \gamma ABC) \setminus \nu(s \mid \gamma AC)) \cap \nu(s \mid \gamma D)) \\
\cup \pi(\nu(s \mid \gamma ABC) \setminus \nu(s \mid \gamma AC)) \cap \nu(s \mid \gamma D)^c) \cap \nu(s \mid \gamma D) \\
\subseteq (\pi(E_1) \cup \pi(\nu(s \mid \gamma D)^c) \cap \nu(s \mid \gamma D)) \\
\subseteq (\pi(E_1) \cap \nu(s)) \cup (\pi(\nu(s \mid \gamma D)^c) \cap \nu(s \mid \gamma D)) \\
= \emptyset
\]

For the last step we used the fact that \( \pi(E_1) \cap \nu(s) = \emptyset \) and \( \pi \mid \nu(s \mid \gamma D)^c = \text{id} \)

This proved that the names flow conditions are satisfied for \( I_{BCD} \) and \( I_{ABD} \).

We now need to prove the three conditions on the store:

1. We want to prove that if \( m \) is a P-move in \( BC \), then forall \( x \) such that \( x \in \nu(s \mid BCD) \setminus \nu(s \mid \gamma BC) \), \( S(x) \) is unchanged in the store. If \( m \) in \( B \), this comes from the way we define the store of the interaction sequence \( t \). Indeed, in this case, \( x \in (\nu(\text{dom}(S(s'_i))) \setminus \nu(\text{dom}(S(s'_i)))) \) and so, by definition, \( S(x) \) is the same as in the preceding move. If \( m \) is in \( C \), using \( I_{ACD} \) and \( I_{ABC} \), one need to establish that
(\nu(s \mid BCD) \setminus \nu(s \mid \gamma BC)) \subseteq ((\nu(s \mid ACD) \setminus \nu(s \mid \gamma AC)) \cup (\nu(s \mid ABC) \setminus \nu(s \mid \gamma BC))), to ensure the property. It is equivalent to.

\iffalse (\nu(s \mid D) \setminus \nu(s \mid \gamma BC)) \subseteq ((\nu(s \mid D) \setminus \nu(s \mid \gamma AC)) \cup (\nu(s \mid A) \setminus \nu(s \mid \gamma BC))
\fi

\Rightarrow (\nu(s \mid D) \setminus \nu(s \mid \gamma BC)) \subseteq ((\nu(s \mid D) \setminus \nu(s \mid \gamma AC)) \cup (\nu(s \mid A) \setminus \nu(s \mid \gamma BC))

We know from \nu(s \mid \gamma D) \cap \nu(s \mid \gamma B) \cap \nu(s \mid \gamma AC) \cap \nu(s \mid \gamma D) = \emptyset. So if x \in (\nu(s \mid D) \setminus \nu(s \mid \gamma BC)) \cap (\nu(s \mid A) \setminus \nu(s \mid \gamma BC)) then x \in \nu(\mid \gamma D) \cap \nu(\mid \gamma AC)cap\nu(\mid \gamma BC). So in particular x \in nu(\mid A) \cap \nu(\mid BC) = \nu(\mid A) \setminus \nu(\mid BC).

2. The case of a P-move in CD is dealt with the same way. If played in D, then it comes from the definition of the interaction sequence we made. In C, it comes just as above using I_{ACD} and I_{BCD}.

3. We now treat the case of an O-move in BD. That is, check that the set \nu(s \mid \gamma BCD) \setminus (s \mid \gamma BD) is fixed. The case B and D are symmetrical, and we will therefore just proof for a move in D. As before, we have \nu(s \mid \gamma ACD) \setminus \nu(s \mid \gamma AD) stay unchanged. So the x that might changed are in \nu(s \mid \gamma C) \setminus (\nu(s \mid \gamma BD) \cap (\nu(s \mid \gamma AD)) \subseteq \nu(s \mid \gamma C) \cap \nu(s \mid \gamma A) \cap \nu(s \mid \gamma B) = (\nu(ABC) \setminus \nu(AC)) \cap \nu(s \mid \gamma ABC) = \pi(E_1) which is, by definition of the store in the interaction sequence t, fixed.

We already know that t \mid \gamma AB \in \mathcal{P}_{AB}, t \mid \gamma BC \in \mathcal{P}_{BC}, t \mid \gamma CD \in \mathcal{P}_{CD}. Still remaining is t \mid \gamma BD \in \mathcal{P}_{BD} and that will be proved in the following lemmas.

**Lemma.** Let s a sequence in A \rightarrow B \rightarrow C that is justified, frugal, alternating, well opened, and such that s \mid \gamma AB \in L_{AB} and s \mid \gamma BC \in L_{BC}. Then s \mid \gamma BD is well bracketed.

In order to do that, we will use a straightforward characterisation of well bracketing. Let \#Q(t) denotes the number of question in t, where t is a subsequence of s, and \#A the number of answer. Then:

s well-bracketed \iff \forall m, n \in s such that m question, n answer, m justifies n, we got

\#Q(s \mid m...n) = \#P(s \mid m...n)

where s\# \mid m...n denotes the subsequence of s starting at m and finishing at n.

We now prove the lemma.

**Proof.** We first prove it for m, n in B. The sequence between the points have the form \alpha_1\beta_1,\eta_1,\beta_1,\eta_2,\ldots\beta_1,\alpha_2,\eta_2,\ldots,\eta_{\ell-1}\beta_1,\alpha_{\ell+1}, where \alpha belongs in A, \beta in B, and \eta in C, and m is the first move of \alpha_1, and n the last move of \alpha_1 + 1. We call \beta_i = \cup_{j=0..k_i}\beta_{i,j}.

Then, by s \mid AB well bracketed, we can conclude that:

\sum_{i=1}^{l} |Q(\alpha_i) + \sum_{j=1}^{k_i} |Q(\beta_{i,j})| + |Q(\alpha_{l+1}) = \sum_{i=1}^{l} |P(\alpha_i) + \sum_{j=1}^{k_i} |P(\beta_{i,j})| + |P(\alpha_{l+1}), and by well bracketness of s \mid AC we have:

22
\[
\sum_{i=1}^{l} (\#Q(\beta_{i,1}) + \sum_{j=1}^{k_i-1} \#Q(\beta_{i,j+1})) + \#Q(\eta_{i,j}) = \sum_{i=1}^{l} (\#P(\beta_{i,1}) + \sum_{j=1}^{k_i-1} \#P(\beta_{i,j+1}) + \#P(\eta_{i,j})).
\]

There is a small trick in the case where the first move of \( \beta_1 \) is initial, as then when looking at it in \( AB \) it appears as an answer, whereas it is a question in \( BC \). However, this does not cause problems. Let suppose the first move of \( \beta_1 \) is not initial. We would like to prove that \( \sum_{i=1}^{l} \#Q(\beta_{i}) - \#A(\beta_{i}) = 0 \). Let us suppose, it is positive, then the bracketing condition implies that some question in \( B \) might be answered in \( A \), which is not possible. If it is negative, it means that they are question that appears before \( m \) that are answered in the sequence, and this breaks the bracketing condition as well.

And now, summing the two equations coming from \( L_{CD} \) and \( L_{AB} \) together with erasing the part lying in \( B \) with this new equation we obtain:
\[
\sum_{i=1}^{l} (\#Q(\alpha_{i}) + \sum_{j=1}^{k_i-1} \#Q(\eta_{i,j})) = \sum_{i=1}^{l} (\#P(\alpha_{i}) + \sum_{j=1}^{k_i-1} \#P(\eta_{i,j})),
\]
that is, \( s \upharpoonright BD \) is well bracketed.

**Lemma.** Let \( s \) be such as in the previous lemma. Then \( s \upharpoonright BD \) satisfies the three names conditions related to polymorphism.

**Proof.** The fact that the left and right copycat conditions for closed variables are true come from the fact that \( s \upharpoonright BC \in L_{BC} \) and \( s \upharpoonright CD \in L_{CD} \). In particular, the left copycat for closed variable condition is true for \( s \upharpoonright BD \) as it is for \( s \upharpoonright BC \), and symmetrically for the right one.

Now let us say player play \( p(X,a) \) in \( BD \). We need to prove that opponent played it first. Suppose \( p(X,a) \) is in \( D \). Let consider the first time it appears in the sequence. \( s \upharpoonright BCD \). If it was in \( C \), then either from the \( BC \) point of view or the \( CD \) point of view it is a player move, so opponent has to play it first, so this case is impossible. So the first time it appears was either in \( B \) or in \( D \). If it was in \( D \), then by the fact that \( s \upharpoonright \gamma \in L_{CD} \), we know that opponent played it first, and the proof in that case is done. If it was in \( B \), we know from \( s \upharpoonright BC \in L_{BC} \) that it was also by an opponent move, and then it is also an opponent move from the \( BD \) point of view, so opponent played it first in \( BD \).

**Lemma.** Let \( s \) such as in the previous lemma and such that furthermore \( \nu(s \upharpoonright \gamma BC) \setminus \nu(s \upharpoonright \gamma C) \cap \nu(s \upharpoonright \gamma D) = \emptyset \). Then \( s \) satisfies the variable freshness condition.

**Proof.** Let \( a \) be a name distinguisher played with \( \chi \) first, where \( \chi \) stands for either \( i \) or \( X \). Suppose without lost of generality it is played in \( B \). Now, \( a \) can not be played with another name that \( X \) in \( B \) because \( s \upharpoonright \gamma BC \in L_{BC} \), so the condition holds in \( B \). Suppose \( a \) is played with a type variable, or a numeral, in \( D \). We will prove it is \( \chi \) as well. Suppose \( a \) is played in \( C \) first. Then because of \( s \upharpoonright BC \in L_{BC} \) it is coupled with \( \chi \) in \( C \), and furthermore as \( s \upharpoonright CD \in L_{CD} \), that enforces \( a \) to be played with \( \chi \) in \( D \) as well. Suppose \( a \) is not played in \( C \). Then \( a \in \nu(s \upharpoonright \gamma BC) \setminus \nu(s \upharpoonright \gamma C) \) so \( a \) cannot appear in \( D \).
Lemma. Suppose $s$ as in lemma ... , then $s \upharpoonright BD$ satisfies the visibility condition.

Proof. For $m$ in $B$, this comes from $s \upharpoonright BC \in L_{BC}$ and the fact that the justifier of a move in $B$ is in $B$. In particular, for a sequence $s$ ending with an move in $B$, $\bar{r}s \upharpoonright B = \bar{r}s \upharpoonright B^\upharpoonright B$. Therefore, if $m$ is a move in $B$, then it justifiers appears in $\bar{r}s \upharpoonright BC \upharpoonright B = \bar{r}s \upharpoonright B^\upharpoonright B$. Now, if $m$ is in $D$, then either it is justifies by a move of $D$, or by the initial move of $C$. If we restrict to $BD$, the moves that were originally modified by the initial move of $C$ are now justified by the initial move of $B$. So let $\phi : i_C \mapsto i_D$, and $\phi = id$ otherwise. For a sequence $m$ that ends in $D$, we have $\bar{r}s \upharpoonright CD \upharpoonright i_C, D) = \bar{r}s \upharpoonright BD \upharpoonright i_B, D$ just as above. This proves the lemma for $D$.

Claim. $s \upharpoonright BD \in L_{BD}$

Each point was prove along the previous lemmas, and that ends the prove of associativity.

4.4 CPO enrichment

4.4.1 CPO structure on strategies

For a type arena $A$, two strategies $\sigma_A$, $\tau_A$, we say that $\sigma_A \preceq \tau_A$ if $\sigma \subseteq \tau$ when we view the strategies as a subset of $P(P_A)$. Let us notice that $\preceq$ is a partial order, in the sense that it is a transitive reflexive antisymmetric relation on strategies. The least element is the empty strategy and the least bound of the infinite growing chain $\sigma_1 \preceq \sigma_2 \preceq \sigma_3$... is define by $\bigcup_{i \in \omega} \sigma_i$. Let us check briefly the three required properties that ensure that $\sigma_\omega$ is indeed a strategy.

- saturation : For $s \in \sigma_\omega$, $\pi(s) \in \pi(\bigcup_{i \in \omega} \sigma_i) = \bigcup_{i \in \omega} \pi(\sigma_i) = \bigcup_{i \in \omega} \sigma_i = \sigma_\omega$
- prefix-closure : $s \in \sigma_\omega$, $\exists i, s \in \sigma_i$, and so the property comes from the prefix closure of $\sigma_i$.
- nominal - determinacy : $sa, tb \in \sigma_\omega$. Then $\exists i, sa, tb \in \sigma_i$, and this comes from nominal-determinacy of $\sigma_i$.

Actually, we furthermore claim that $G$ is enriched over CPO, that is, the composition is a monotonous bi-continuous operation.

- $\sigma \preceq \sigma', \tau \preceq \tau' \Rightarrow \sigma; \tau \preceq \sigma'; \tau'$

Proof. Let $s \in \sigma; \tau$, $\sigma; A \to C$, then $\exists s' \in \sigma \upharpoonright \tau$, $s = s' \upharpoonright_\gamma AC$. As $\sigma \subseteq \sigma'$, $s' \upharpoonright_\gamma AB \in \sigma \Rightarrow s' \upharpoonright_\gamma AB \in \sigma'$, and respectively for $\tau$, that thus leads to $s' \in \sigma' \upharpoonright \tau'$, and therefore $s = s' \upharpoonright_\gamma AC \in \sigma'; \tau'$. $

- \left(\bigcup \sigma_i\right); \tau = \bigcup (\sigma_i; \tau)$

24
Proof. We need to prove, $(\cup \sigma_i) ; \tau \preceq \cup_{i \in \omega} (\sigma_i ; \tau)$, the other sense following from the continuity of the composition. Let $s \in (\cup \sigma_i) ; \tau$ then $\exists s' \in (\cup \sigma_i) | \tau, s = s' |_{\gamma} \text{AC}$. Then $s' |_{\gamma} AB \in \cup_{i \in \omega} \sigma_i$, and, as a consequence $\exists i$ such that $s' |_{\gamma} AB \in \sigma_i$, and $s' \in \sigma_i | \tau$ that finally implies that $s \in \sigma_i ; \tau$, so $s \in \cup_i (\sigma ; \tau)$.

\[ \bullet \sigma ; (\cup \tau_i) = \cup (\sigma_i ; \tau_i) \]

Proof. Just as before.

\[ \square \]

Theorem 2. G is a CPO enriched category.

4.4.2 CPO structure on nominal arenas

This aim of this section is to pave the road towards the solution of the recursive types equation. Before trying to solve it by domain theory, we hereby shortly show that the arena themselves form a CPO.

We embed the domain of nominal arenas with a CPO-structure, giving a partial order, and a least element, the empty arena. For each two elements, we write $A \triangleleft B$ if there is an injection between from $A$ to $B$ that respect the structure, that is, more explicitly :

\[ \bullet M_A \text{ nominal subset of } M_B \]
\[ \bullet I_A = I_B \cap M_A \]
\[ \bullet \lambda_A = \lambda_B \upharpoonright A. \]
\[ \bullet \vdash_A = \vdash_B \upharpoonright A, \]
\[ \bullet n_A = n_B. \]

We write $inj$ for the nominal injection from $A$ to $B$, and $proj$ for the partial nominal function from $B$ to $A$. Let us notice that $\triangleleft$ is a partial order, in the sense that it is a transitive reflexive antisymmetric relation on nominal arenas.

Definition. Two morphisms $e : A \rightarrow B, p : B \rightarrow A$ forms a section retraction pair if $p \circ e = id_A$ and $e \circ p \preceq id_B$.

In particular, when $A \triangleleft B$ we define :

\[ e = \{ s \in P_{A \rightarrow B} | inj(s \upharpoonright A) = s \upharpoonright B \} = \{ s \in P_{B \rightarrow A} | s \upharpoonright B = inj(s \upharpoonright A) \} \]

There is also a constructivist way to define them, by taking :

\[ e = \{ s = ...x_i.y_i, i \in 1...n | s \in L_{A \rightarrow B}, n \in \mathbb{N} \land (i \simeq 0[2] \Rightarrow x \in A, y \in B, y = inj(x)) \]
\[ \lor (i \simeq 1[2] \Rightarrow x \in B, y \in A, x = inj(y)) \}. \]

\[ p = \{ s = ...x_i.y_i, i = 1...n | s \in P_{B \rightarrow A}, n \in \mathbb{N} \land (i \simeq 0[2] \Rightarrow x \in B, y \in A, x = inj(y)) \]
\[ \lor (i \simeq 1[2] \Rightarrow x \in A, y \in B, y = inj(x)) \}. \]

This construction proofs the following claim :
Proposition. \( A \triangleleft B \iff \exists e : A \rightarrow B, p : B \rightarrow A \) such that \( A \rightleftharpoons_B e^e_p \) section retraction pair.

Lemma. Embedding and retraction compose.
For \( A \rightleftharpoons_A e_1 B \rightleftharpoons_B e_2 C \) such that \( e_i, p_i \) section retraction pair, we have that \( A \rightleftharpoons_A e_2 \circ e_1 B \rightleftharpoons_B p_1 \circ p_2 C \) section retraction pair.

Proof. The composition of \((x, \text{inj}_1(x))\) with \((y = \text{inj}_1(x), \text{inj}_2(y))\) will give you a pair \((x, \text{inj}_1(\text{inj}_2(x)))\), and symmetrically the other way around. That proves that both \( e_1 \circ e_2 \) and \( p_1 \circ p_2 \) are the strategies \( e \) and \( p \) associated with the injection \( i_2 \circ i_1 \) and therefore are a section retraction pair.

For a growing chain \((A_i)_{i \in \mathbb{N}}, \) such that \( \forall i, j \; j > i \Rightarrow A_i \triangleleft A_j, \) the least upper bound of this chain, written \( M_\omega = \bigcup_{i \in \mathbb{N}} A_i \) is defined by increasing union.

4.5 Freyd category structure
The aim of this section is to show that the full subcategory of arenas that have not the \( \forall, \) in their description, that is, the closed arenas, form a Freyd category. More explicitly, we restrain the category \( G \) to the arenas that does not have any move of the form \( p(i, a) \) where \( p \) is a pattern, \( i \) a numeral and \( a \) a name distinguisher. We write \( G_c \) for this category.

4.5.1 Premonoidal structure
We begin this section by interesting ourselves in the \( \times \) defined in the upper section. We would like to investigate its functoriality. First, we can notice that \( 1 \times A \simeq A \times 1 \simeq A. \) So \( 1 \) is the unit of the \( \times \) operation. Let us denote \( \lambda_A : 1 \times A \simeq A \) and \( \rho_A : A \times 1 \simeq A. \) However, given two morphisms \( f : A \rightarrow A' \) and \( g : B \rightarrow B' \) there is no canonical way we can form a morphism \( f \times g : A \times B \rightarrow A' \times B'. \) Indeed, some order conditions appears : after having played \((i_A, i'_A)\) should we react according to \( f \) or \( g \) ?

However, we can see that for each object \( A, \) we can create two functors \( A \times \_ \) and \( \_ \times A. \) They are defined by the \( \times \) operation on objects, and for \( f : B \times B', A \times f : A \times B \rightarrow A \times B' \) is defined to be :

\[
A \times f = \{ s \in L_{A \rightarrow B} \mid s \gamma B \rightarrow B' \in f \land s \gamma A \rightarrow A \in \text{id}_A \\
\land (P(s \gamma B \times B') = P(s)) \}
\]
\( f \times A \) is defined symmetrically. In order to ensure this indeed defines a strategy, we first have to prove that this is indeed well defined. For that, we need the following lemma.

**Lemma.** For a strategy \( \sigma : A \times B \rightarrow C \times D \), only player can switch between \( A \) and \( B \), and only opponent between \( C \) and \( D \).

The proof goes exactly the same as in lemma ..., therefore we will not reproduce it here.

**Theorem 3.** \( A \times f \) is a strategy

**Proof.** Prefix closure and saturation are straightforward so we will just investigate determinacy. Say \( s,t \in A \times f \), and \( \downarrow s \cdot j = \downarrow t \cdot j \). Moreover, \( s,a,t,b \in A \times f \). We want to show \( \downarrow s \cdot a \cdot j = \downarrow t \cdot b \cdot j \). For that, we want to first ensure that if \( f \) (resp \( id_A \)) played \( a \) then \( f \) (resp \( id_A \)) played \( b \) as well. For that, we will examine what can happen after the first move \((i_a,i_b)\) in \( A \times B \). Either \( f \) plays directly in \( B' \), in which case the second move will be \((i_a,i'_b)\), either it explores \( B \) first. In that case, it follows the \( f \) behaviour, and every move stay in \( B \) as opponent can not switch. In the point where \( f \) is ready to answer in \( B' \), the move \((i_a,i'_b)\) finally happens. Afterwards if opponent plays in \( A \) then it reacts as copycat playing the same in the left hand side, as if it would play in \( B \), then, from the \( B \rightarrow B' \) point of view, it would be two consecutive player moves. We can prove straightforwardly that if it is opponent to play, the last move of both \( s \upharpoonright A \rightarrow A \) and \( s \upharpoonright B \rightarrow B' \) are by players. So if opponent plays in one of them(\( BB' \)), then player can not answer in the other as it would break the alternance condition of it (here \( L_{AA} \)). So player play reacts either by \( id \) or by \( f \) depending on where is the last move of the opponent.

So in our case, if \( a \) is played by \( id_A \) (resp \( f \)) then \( b \) as well. Let us say \( a \) played by \( id_A \). Then, \( a \) does not introduce any new name, so if \( \downarrow s \cdot j = \downarrow t \cdot j \), then the same bijection between \( s \) and \( t \) yields to \( \downarrow s \cdot a \cdot j = \downarrow t \cdot b \cdot j \). Now suppose \( a \) is played by \( f \). If \( a \) introduces any new name for \( s \upharpoonright BB' \), it is new for \( s \) as well. This respectively holds for \( t \) and \( b \) as well. Then by \( \sigma \) determinacy we have \( \downarrow s \upharpoonright BB',a \cdot j = \downarrow t \upharpoonright BB',b \cdot j \), and \( \downarrow s \upharpoonright BB',s \cdot j = \downarrow t \upharpoonright BB',t \cdot j \). To apply the strong support lemma and prove the theorem, we need to prove that if \( x \in \nu(a) \cap \nu(t) \cap \nu(BB') \) then \( x \in \nu(t \upharpoonright BB') \). That comes from the fact that if \( x \in \nu(a) \) and \( x \notin \nu(t \upharpoonright BB') \) then \( x \in P(s \upharpoonright BB') \), and therefore \( x \in P(s) \) that is \( x \notin \nu(t) \). So we have \( \downarrow s \upharpoonright BB',s,a \cdot j = \downarrow t \upharpoonright BB',b,t \cdot j \) and in particular \( \downarrow s \cdot a \cdot j = \downarrow t \cdot b \cdot j \).

**Proposition.** \( A \times f :\)

\[
B \rightarrow A \times B
\]

\[
f : (B \rightarrow C) \rightarrow A \times f : (A \times B) \rightarrow (A \times C)
\]

is a functor

**Proof.** We need to prove that \( A \times id_C = id_{A \times C} \) and \( A \times (\sigma; \tau) = (A \times \sigma); (A \times \tau) \).

27
\[ A \times id_C = \epsilon \cup \{ s \mid s \in L_{(A \times C)} \rightarrow (A \times C) \wedge s \mid_\gamma A \rightarrow A \in id_A, s \mid_\gamma C \rightarrow C \in id_C \wedge P(s \mid_\gamma C \rightarrow C) = P(s) \}. \]

As if \( s \in id_{A/C} \) then \( s \) does not introduces new names, and moreover that by covering \( A \rightarrow A, C \rightarrow C \) we covered every moves that can happen in \( A \times C \rightarrow A \times C \), we know that \( P(s) = \emptyset \). So we can drop the last condition about names. It only remains to prove that if \( A \) that can happen in \( s \) stop the sequence at this point, (by prefix closure), \( s \mid A \rightarrow A \) would end by an opponent move and therefore would not be part of \( id_A \). So \( P \) will answer in \( A \) by copycat just as \( id_{A \times C} \) would have done. So \( A \times id_C = id_{A \times C} \).

We prove it by induction on the length of the sequence. Let \( s \in (A \times f) \cap (A \times g) \). Then \( \exists t \in (A \times f) \cap (A \times g) \) such that \( g = t \mid_\gamma (A \times B) \rightarrow (A \times D) \). Now suppose the property is true for \( s \) and we are trying to prove it for \( s \cdot m \cdot n \). So \( s \in (A \times (f; g)) \). So \( \exists u \in f \mid g \) such that \( u \mid_\gamma B \rightarrow D = s \mid B \rightarrow D \). Suppose \( m \) in \( A \), then \( smn \in (A \times f) \cap (A \times g) \) as for the both cases \( n = m \), the strategies just act as copycat. Now suppose without lost of generalities that \( m \) in \( B \), the case \( D \) being identical.

We will first prove the inclusion \( (A \times f) \cap (A \times g) \subseteq (A \times (f; g)) \). Let \( t \) be the interaction sequence such that \( t \mid_\gamma (A \times B) \rightarrow (A \times D) = smn \). As \( t \in I_{(A_1 \times B_1), (A_2 \times C_2), (A_3 \times D_3)} \), \( t \mid_\gamma BCD \in I_{B,C,D} \). Moreover \( t \mid_\gamma BC \in \sigma \) and \( t \mid_\gamma CD \in \tau \). So \( t \mid_\gamma BCD \in \sigma \mid \tau \), and therefore \( t \mid_\gamma B \rightarrow D \in \sigma \mid \tau \). Moreover, \( t \mid_\gamma A_1 \rightarrow A_3 \in id_A \). So it remains to prove that \( P(t \mid_\gamma B, D) = P(t \mid_\gamma (A \times B), (A \times D)) \). As \( t \) is an interaction sequence, we know that every name that appears in the sequence is either from \( P_t(A \times B, A \times C) = P_t(BC) \), or from \( P_t(A \times C, A \times D) = P_t(CD) \), or from \( O_t(A \times B, A \times D) \). So we can deduce that \( P_t(A \times B, A \times D) = (P_t(A \times B, A \times C) \cup P_t(A \times C, A \times D)) \cap \nu(t \mid_\gamma A \times B \rightarrow A \times D) = (P_t(B, C) \cup P_t(C, D)) \cap \nu(t \mid_\gamma B \rightarrow D) = P_t(BD) \), which proves that \( t \mid_\gamma A \times B \rightarrow A \times D \) is indeed in \( A \times (\sigma \mid \tau) \). The other way around is a little bit trickier. Indeed, given an interaction sequence \( t \) in \( A \times (f \mid g) \), to transform it into an interactions sequence \( (A \times f) \mid (A \times g) \) boils down to prove that \( P_t(A \times B, A \times D) = P_t(BD) \Rightarrow P_t(A \times B, A \times C) = P_t(BC) \cup P_t(A \times C, A \times D) = P_t(CD) \), and this is unfortunately wrong. Just as in the proof of associativity, we have to consider the problematic names and move them apart using a name-permutation and the name-saturation of strategies. So let \( tml'n \) be the interaction sequence in \( \sigma \mid \tau \) such that \( tml'n \mid_\gamma B \rightarrow D = s \cdot m \cdot n \mid_\gamma B \rightarrow D \), and moreover, as \( s \in A \times \sigma \mid A \times \tau \), by induction hypothesis there exists an interaction sequence \( u \in A \times \sigma \mid A \times \tau \) such that \( s = u \mid_\gamma A \times B \rightarrow A \times D \), and \( u \mid B, C, D = t \). Let \( S = P(u.m.t.'n \mid_\gamma BC) \cap (P(u.m.t.'n \mid_\gamma A \times B, A \times C) \cup P(u.m.t.'n \mid_\gamma CD) \cap P(u.m.t.'n \mid_\gamma A \times B, A \times C)) \). Let us note that the induction hypothesis entails \( \nu(t) \cap S = \emptyset \), and moreover as \( m \) played by opponent \( \nu(m) \cap S = \emptyset \) as well. We know that \( P(u.m.t.'n \mid BD) = P(t.m.t.'n \mid BD) = P(t.m.t.'n \mid A \times B, A \times D) = P(u.m.t.'n \mid A \times B, A \times D) \). In particular, this proves that \( \nu(n) \) is not part of \( S \), using the same equation about names as in the part above. So let \( \pi \) such that \( \pi S \cap S = \emptyset \) and \( \pi \mid \nu(u.m.n) = id \). We apply \( \pi \) to
m.t'\cdot n to construct the interaction sequence \( v = ov_\cdot (\pi (m.t'\cdot n)) \in A \times \sigma \mid A \times \tau \).
Finally, this proves that \( s.m.n = \pi (u.m.t'\cdot n) \mid A \times B \), \( A \times D \in A \times f; A \times g \).

We can prove exactly the same way that \( \cdot \times A \) is a functor, and we obtain the following theorem.

**Theorem 4.** \( \times \) provides \( G \) with a premonoidal structure whose unit is \( \text{id} \).

**Proof.**
- Pentagon law: missing
- Naturality of \( \lambda \) and \( \rho \): missing
- Naturality of the associativity morphisms: missing

4.5.2 Total morphisms and products

We are looking for a subcategory of \( G_c \) such that the restriction of \( \times \) to it provides product. What prevented it from being so in the previous part was that \( f \) could explore \( B \) before playing in \( B' \). We had therefore to implicitly "force" the strategy to act as \( f \) before acting as \( id_A \). This notion of order will be highlighted in the next section. For now, we will restrict the strategy to morphisms that answer directly in the co-domain, in order to avoid this restriction.

**Definition.** A strategy \( A \rightarrow B \) is total if:
- for any plays in it, the second move belongs to \( B \).
- for \( i_A^{S_1}, i_B^{S_2} \) in \( \sigma \) we have \( S_1 \subseteq S_2 \).

We name \( G_t \) the lubb subcategory of \( G_c \) restricted to total morphisms.

**Theorem 5.** \( G_t, \times, \text{id} \) is a symmetric monoidal category

**Proof.** We need to prove that every morphism is central that is, for every \( f : A \rightarrow A' \) and \( g : B \rightarrow B' \) we have

\[
A \times B \xrightarrow{f \times B} A' \times B \\
\downarrow A \times g \quad \downarrow A' \times g \\
A \times B' \xrightarrow{f \times B'} A' \times B'
\]

By induction on the length of the play. First, we need to prove it for the second move. Let us consider \( s = (i_A, i_B)^{S_1}(i'_A, i'_B)^{S_2}(i''_A, i''_B)^{S_3} \in f \times B \mid A' \times g \) and \( t = (j_A, j_B)^{T_1}(j'_A, j'_B)^{T_2}(j''_A, j''_B)^{T_3} \in A \times g \mid f \times B' \) such that \( (i_A, i_B)^{S_1} \simeq (j_A, j_B)^{T_1} \). As \( f \) only add names to the store, we have \( S_2 \mid \nu(s \mid \gamma \cdot B) = S_1 \mid \nu(s \mid \gamma \cdot B) \). So we have \( i_B^{S_1} \mid \gamma \cdot B \simeq i_B^{S_2} \mid \gamma \cdot B \), and by \( g \) nominal determinacy, we have \( i_B^{S_1} \mid \gamma \cdot B \simeq i_B^{S_2} \mid \gamma \cdot B \).
There is at most one move justified by $a$. A sequence is single threaded if only if some moves happened in $A$ before, then all the play will take place in $A''$, and symmetrically with $B$. Following ..., we introduce the notion of thread, that corresponds to a path of a single execution of the function.

**Definition.** For a total sequence $qas$, a thread of a sequence is an equivalence class of moves in the sequence.

$$s_i \sim_{\text{thread}} s_j \iff \exists s_k \text{ such that } a \vdash s_k \text{ and } (s_k \vdash^* s_i \land s_k \vdash^* s_j)$$

A sequence is single threaded if it has at most one thread-equivalence class.

**Proposition.** A play $s \in L_{A \rightarrow B} = qas^l$ that is total is single threaded if and only if there is at most one move justified by $a$ in $s'$.

We are almost done, but a property allowing us to separate the store is still needed. For example, imagine a morphism $\text{unit} \rightarrow \text{ref}(\text{int}) \times \text{ref}(\text{int})$ that introduces twice the same new name, that is has as play $\star.(a^a,a^a)$. Then we can not decompose this
morphism into two ones \( f : \text{unit} \to \text{refint} \) and \( g : \text{unit} \to \text{refint} \) as those, by nominal determinacy, could introduce two different names.

**Definition.** We say that a morphism \( \sigma : A \to B \) is strongly total if it is total and moreover if \( i_A^S \cdot j_B^S \in \sigma \) then \( S_A = S_B \).

**Proposition.** A total morphism is strongly total if \( P(q.a) = \emptyset \).

We call \( G_{\text{sd}} \) the luff category of \( G \) of strongly total single threaded morphisms. The composition of two strongly total single threaded morphisms yield indeed a strongly total single threaded morphism, but it might be too restrictive to force the left hand side morphism to be single threaded. Indeed, the single threaded condition imposes some properties on moves on \( B \) (for a morphism \( A \to B \) ), however, if we compose two strategies \( A \to \pi B \to \tau C \), the fact that \( \tau \) is single threaded does not prevent it from dealing with many threads in \( B \). Therefore, we define a new way of composing the morphisms, by transforming the left hand side morphism into one multi-threaded, but such that each thread is "independent" from the other ones.

**Definition.** We define the function \( \text{thread} : L_{A\to B} \to L_{A\to B} \) such that for each \( s \), \( \text{thread}(s) \) is single threaded by

\[
\begin{align*}
\text{thread}(sm) &= \gamma(s.m \cap (A \cup \{ n \mid n \in s \mid B \land n \sim \text{thread \, m} \})) \text{ if } m \in B \\
&= \gamma(\text{thread}(s).m) \text{ otherwise}
\end{align*}
\]

**Definition.** A play \( s \in L_{A\to B} \) is thread independent, written \( s \in \text{Ind}(A, B) \) if, whenever player introduces a name new for the sequence, it is new for the whole play, and both opponent and player cannot change names affectation that lies outside the thread. That is, \( s \in \text{tot}(A, B) \) if \( s \in L_{A\to B} \) and, for all \( s'.m^{S_m}.n^{S_n} \) subsequence of \( s \)

\[
P(\text{thread}(s'.m^{S_m}.n^{S_n})) \subseteq P(s'.m^{S_m}.n^{S_n})
\]

\[
\forall x \in \nu(s'.m^{S_m}.n^{S_n}) \setminus \nu(\text{thread}(s'.m^{S_m}.n^{S_n})), S_m(x) = S_n(x)
\]

This concept extends straightforwardly to strategies.

**Definition.** Given a single threaded strategy \( \sigma \), we define its bang \( \sigma^\dagger \) by

\[
\sigma^\dagger = \{ s \in \text{tot}(A, B) \mid \forall s'.m \subseteq s, \text{thread}(s'.m) \in \sigma \}.
\]

**Theorem 6.** \( \sigma^\dagger \) is a strategy.

**Proof.** The prefix closure is proved the usual way. The saturation comes from the one of \( \sigma \) and the fact that if \( P(\text{thread}(s'.m)) \subseteq P(s'.m) \) then, for every name permutation \( \pi \) the property is conserved by translation over \( \pi \) that is \( P(\text{thread}(\pi(s'.m))) \subseteq P(\pi(s'.m)) \). About nominal determinacy, let us pick two sequences \( s \) and \( t \) in \( \sigma^\dagger \) such that \( s \simeq t \), and such that last move of \( s \) and \( t \) are by opponent. Then let \( a_S \) and \( b_S \) such that \( s.a_S \) and \( t.b_S \) in \( \sigma^\dagger \). We know by \( \sigma \) determinacy that \( \text{thread}(s.a_S) \simeq \text{thread}(t.b_S) \). And moreover, player can not change thread, that is \( \text{thread}(s.a_S) \setminus a_S = \text{thread}(s) \). So \( \text{thread}(s), s.\downarrow = \text{thread}(t), t.\downarrow \) and \( \text{thread}(s), \text{thread}(s.a_S) \downarrow = \text{thread}(s) \).
\documentclass{article}
\usepackage{amsmath}
\usepackage{amsthm}
\usepackage{amssymb}
\usepackage{mathrsfs}
\usepackage{enumerate}
\usepackage{url}
\usepackage{hyperref}
\usepackage{color}

\newtheorem{lemma}{Lemma}
\newtheorem{definition}{Definition}
\newtheorem{proof}{Proof}

\begin{document}

Lemma. Let $u$.

\begin{proof}
\end{proof}

Lemma. For $\sigma$, $\tau$ two morphisms of $G_{st}$, we have
\begin{itemize}
\item $\sigma^\dagger; \tau \in G_{st}$
\item $\text{der}^\dagger = \text{id}_A$
\item $\tau^\dagger; \text{der} = \tau$
\item $\text{der}^\dagger; \tau = \tau$
\item $(\sigma^\dagger; \tau)^\dagger = \sigma^\dagger; \tau^\dagger$
\item $(\tau^\dagger; \sigma)^\dagger; \rho = \tau^\dagger; (\sigma^\dagger; \rho)$
\end{itemize}

The fifth point is a little bit tricky and we need some more tools to prove it nicely.

Definition. The thread functions and definitions are extensible to interaction sequences.
\begin{itemize}
\item A interaction sequence $s \in I_{A \rightarrow B \rightarrow C}$ is single threaded if there is only one move justified by the initial move in $C$. In other words, $s$ single threaded $\iff s \mid A \rightarrow C$
\end{itemize}

Lemma. Let $\sigma : A \rightarrow B$ and $\tau : B \rightarrow C$ be two strategies.

\begin{proof}
\end{proof}

\end{document}
Let $\pi$ such that $\pi \upharpoonright \nu(u) \setminus S = id$ and $\pi(S) \cap (\nu(t) \cup \nu(u)) = \emptyset$. Finally, let $t := t' \circ m \circ \pi(u_{j+1} \ldots u_{k-1}, n)$. We claim that $t$ now satisfies the lemma. Let us first prove that $t$ is an interaction sequence. As the sequence we add is in $B$, we have to intereset ourselves in $P_I(AB)$ and $P_I(BC)$. $P_I(AB) = P_I(AB) \cup \pi((P(u |_\gamma A \rightarrow B)) \setminus P(u_1 \ldots u_{j-1}, m, n |_\gamma A \rightarrow B)))$ and this union is disjoint as we picked $\pi$ such that $\pi(S) \cap \nu(u(t')) = \emptyset$. The same holds for $P_I(BC)$. Then the intersection of both are disjoint as $t'$ as both $t'$ and $u'$ are interaction sequence, and we furthermore took care that $\pi((P(u |_\gamma A \rightarrow B)) \setminus P(u_1 \ldots u_{j-1}, m, n |_\gamma A \rightarrow B))) \cap P_I(BC) = \emptyset$. The condition on the store are proved equally. Let us also note that $\pi(t'') = t''$, by definition of $\pi$, so basically we have applied $\pi$ to the whole thread $u = t''m.u_{j+1} \ldots u_{k-1}, n$. Therefore $\text{thread}(t) = \pi(u) \in \sigma^\dagger \upharpoonright \tau$ by saturation of $\sigma \upharpoonright \tau$. Furthermore, when player $u$ brings new names for the thread, they are new for the all sequences. So $t \upharpoonright AB \in \sigma^\dagger$, as required. The same applies also in the BC part, that is $t \upharpoonright BC \in \tau^\dagger$. 

So this lemma entails $(\sigma^\dagger; \tau)^\dagger \subseteq \sigma^\dagger; \tau^\dagger$. We now need to prove the other way around.

**Lemma.** If $s \in \sigma^\dagger; \tau^\dagger$ then $s$ is thread independent.

**Proof.** Let us first focus on the case where player introduce a new name $a$ for the thread in $C$. As the thread covers all names in $A$, we know that $a$ never appears in $A$ before. We would like to use the thread independance of $\tau^\dagger$ to say that $a$ never appears in $C$ before. It would proved the lemma. However, we need to ensure that $a$ is indeed new in the $B \rightarrow C$ sequence, as it could appears in $B$ without never appearing in $A$. The visibility together with the strongly total condition will help us solve the problem.

Let $t.m$ be a sequence ending with a P-move in $C$ that brings new names for the sequence, $a \in \nu(m) \setminus \nu(\text{thread}(t.m))$. Let $t'$ be the interaction sequence in $\sigma^\dagger \upharpoonright \tau^\dagger$ such that $t'.m$ legal and $t' \upharpoonright \gamma A, C = t$. Suppose $a$ never appears in $B$, that is $a \notin \nu(t' \upharpoonright \gamma B)$. Then $a \in P(\text{thread}(t'.m \upharpoonright \gamma BC))$, and, by thread independance of $\tau^\dagger$, $a \in P(t'.m \upharpoonright \gamma BC)$. As, moreover, $a$ does not appears in $A$, we have that $m$ indeed introduces $a$ in $t'$, and, in particular, in $t' \upharpoonright \gamma BC = t$.

Now suppose $a$ appears in $B$, brought by the move $m'$. Let call it $t''.m'$ the subsequence of $t'$ that finishes by $m'$. Let us say that $m'$ was by opponent, from the $BC$ point of view. Then, it can only be "used" by the thread of $\tau$ that encompass $\text{thread}(t''.m' \upharpoonright BC)$, as long as opponent does not play it in $A$. Indeed, let us consider that opponent start a new thread in $C$. Then, player can not see the names that have appeared in the preceding thread, except the initial moves. But these ones do not bring any new names, and $a \notin A$, so $a$ does not belong in any initial move. Therefore, if the new thread want to fetch some names in $B$, it has to start explore $B$ all over again. As $a$ was brought by opponent in $BC$, it means it was brought by player in $AB$. Furthermore, $a$ does not appear in $A$, so $a \in P(AB)$. As $\sigma^\dagger$ thread independant, when asked a second time, $\sigma^\dagger$ will not bring the same $a$.

Now suppose that the first time $a$ appears in $B$ brought by $m'$, it was by player from the $BC$ point of view. So by thread independance of $\tau^\dagger$, we know that $m$ does not introduce $a$ in $\text{thread}(t'.m \upharpoonright \gamma BC)$. So there is two cases to consider. Either $m'$
is in \text{thread}(t'.m \mid \gamma, BC) or it was brought by an opponent move. We will exclude the second case, leading to the conclusion of the proof. If it was brought by an opponent move in \text{BC} to the thread, then this would be a P-move in \text{AB}. As, as we have seen in the previous paragraph, a new thread in C will begin new threads in B, there is two possibilities. Let us call \( u = \text{thread}(t'.m) \). Either \( a \in O_u(AC) \mid A \) or \( a \in P_u(AB) \). The first case is impossible as \( a \) does not appears in \( A \). The second is also problematic by the thread independance of \( \sigma^\dagger \) that yields \( P_u(AB) = P_t^\dagger(AB) \), but as \( a \in P_t^\dagger(BC) \) and \( P_t^\dagger(BC) \cap P_t^\dagger(AB) = \emptyset \), we obtain a contradiction.

We can finally prove our six points lemma about composition of thread independance strategies.

Proof. \( 1. \) We need to prove that \( \sigma^\dagger; \tau \) strongly total single threaded strategy. The strongly total property is straightforward. The fact that this is has a single thread comes from the fact that if \( s \in \sigma^\dagger; \tau : A \rightarrow C \) then \( s \mid C \in \sigma \mid C \) that is has only one move justified by its first move, which entails the property.

\( 2. \) The first point is true because player never introduces a name in \( id_A \), therefore two different threads are automatically independant. So \( \text{der}^\dagger = \{ s \mid s \in \text{tot}(A, A) \land \forall s'.m \subseteq s, \text{thread}(s'.m) \in \text{thread}(id_A) \} = \{ s \mid s \in L_{A \rightarrow A} \land \forall s'.m \subseteq s, \text{thread}(s'.m) \in \text{thread}(id_A) \} = \{ s \mid s \in id_A \} = id_A \).

\( 3. \) This comes from the fact that if \( \text{der}_A \) is left single threaded in the sense that for any sequences \( q_1a_1q_2a_2s' \) in \( id_A \) there is only one move justified by \( a_2 \) in \( s' \). So for any interaction sequences \( s \in \tau^\dagger \mid \text{der}_B, \tau : A \rightarrow B, \) we have \( s \mid \gamma, A \rightarrow B \) single threaded and therefore \( a \mid \gamma \in \tau \). Then \( \text{der}_B \) just does copycat and therefore \( \tau^\dagger; \text{der} = \tau \).

\( 4. \) Comes from \( 2) \)

\( 5. \) We already know the left inclusion; \( (\sigma^\dagger; \tau)^\dagger \subseteq (\sigma^\dagger; \tau^\dagger) \). The other sense is pretty easy once we know that \( \sigma^\dagger; \tau^\dagger \) is thread independant. It is just enough to notice that for any \( s \in \sigma^\dagger \mid \tau^\dagger \) we have \( \text{thread}(s) \in \sigma^\dagger \mid \tau \), which comes directly from the definition of thread of interaction sequence.

\( 6. \) We use both the previous point and the associativity of \( ; \).

\( (\sigma^\dagger; \tau^\dagger); \rho = (\sigma^\dagger; \tau^\dagger); \rho = \sigma^\dagger; (\tau^\dagger; \rho) \)

\( \square \)

Definition. \( G_{st} \) has strongly total single threaded strategies as morphisms, and composition of morphisms is defined by \( \sigma : \tau = \sigma^\dagger; g \) where the ";" is the composition of strategies in \( G_c \).

Theorem 7. \( G_{st} \) is a category. That is composition of strongly total stringle threaded strategies yields indeed strongly total single threaded strategies. It has an identity on all objects, and the composition is associative.
Proof. The theorem is a direct consequence of the lemma.

Theorem 8. \( \times \times \) is a product on \( G_{st} \), and \( I \) is the terminal object. The two projections are defined by

\[
\pi_1 : A \times B \mapsto A = \{ s \in L_{A \times B \to A}, s \mid_\gamma A \mapsto A \in id_A \}
\]

Proof. Let \( C \) be an object, \( \sigma : C \to A \) and \( \tau : C \to B \). Then we define

\[
\langle \sigma, \tau \rangle = \epsilon \cup \{ c^S.\langle i_A, i_B \rangle^S \mid c^S.i_A^S \in \sigma \land c^S.i_B^S \in \tau \}
\]

\[
\cup \{ c^S.\langle i_A, i_B \rangle^S.j_A^S.j_B^S \mid c^S.i_A^S \in \sigma \land c^S.i_B^S \in \tau \}
\]

\[
\cup \{ c^S.\langle i_A, i_B \rangle^S.j_B^S \mid c^S.i_A^S \in \sigma \land c^S.i_B^S.j_B^S \in \tau \}
\]

\[
= \{ s \mid s \in L_{C \to A \times B}, s \mid C \mapsto A \in \sigma \land s \mid C \mapsto B \in \tau \}
\]

Then we have \( \langle \sigma, \tau \rangle ; \pi_1 = \sigma \) and \( \langle \sigma, \tau \rangle ; \pi_2 = \tau \) by definition. For the uniqueness, we have to prove that for any \( \rho : C \to A \times B \), we have \( \langle \pi_1(\rho), \pi_2(\rho) \rangle = \rho \). This comes from the fact that we can gives to \( \rho \) a similar decomposition that the one we gave to \( \langle \sigma, \tau \rangle \). If \( c^S.\langle i_A, i_B \rangle^S.j_A^S \) in \( \rho \), then \( c^S.i_A^S \) in \( \rho ; \pi_1 \) (and respectively for \( \rho \) \( \pi_2 \)), so we have the following equations:

\[
\rho = \epsilon \cup \{ c^S.\langle i_A, i_B \rangle^S \mid c^S.i_A^S \in \pi_1(\rho) \land c^S.i_B^S \in \pi_2(\rho) \}
\]

\[
\cup \{ c^S.\langle i_A, i_B \rangle^S.j_A^S.j_B^S \mid c^S.i_A^S \in \pi_1(\rho) \land c^S.i_B^S \in \pi_2(\rho) \}
\]

\[
\cup \{ c^S.\langle i_A, i_B \rangle^S.j_B^S \mid c^S.i_A^S \in \pi_1(\rho) \land c^S.i_B^S.j_B^S \in \pi_2(\rho) \}
\]

\[
= \{ s \mid s \in L_{C \to A \times B}, s \mid C \mapsto A \in \pi_1(\rho) \land s \mid C \mapsto B \in \pi_2(\rho) \}
\]

that proves the uniqueness of the product.

4.5.3 Exponential and strong monad

The aim of this section is to prove that the two category \( G \) and \( G_{st} \) together with the tensor product \( \times \) on \( G_{st} \) (almost) yields a closed Freyd category.

Definition. A Freyd category consists of a category \( C \), with finite products, a symmetric premonoidal category \( (C, \otimes) \), and an identity on objects strict symmetric premonoidal functor \( F : C_v \to C \) such that for every \( A \) the functor \( F(\_ \otimes \_ \otimes A) \) has right adjoint.

But there is a slight modification that we have to impose. Indeed, in order to yield to exponential, we need the player to answer first without read the store. This new condition for strategies is formally expressed as:

- \( \forall i_A \in I_A, \forall S compatible with i_A, \exists i_B \) such that \( i_A^S.i_B^S \in \sigma \).
We call $G_{sst}$ the category that have strongly total single threaded with strong thread condition strategies as morphisms. The strong thread condition composes.

**Theorem 9.** $G_{sst}$ is a category.

**Proof.** (proof missing).

By proving that $(G_c, G_{sst}, \times, 1)$ is a Freyd category, we will know (ref) that $G_c$ is a \textit{direct $\lambda_c$-model}, and that $G_{sst}$ is its sub-category of values.

**Proposition.**

$\downarrow \cdot : A \rightarrow A$

$f : (A \rightarrow B) \rightarrow f^\uparrow : A \rightarrow B$

is a functor from $G_{sst}$ to $G$.

**Proof.** $(f : g)^\uparrow = (f^\uparrow ; g)^\uparrow = f^\uparrow ; g^\uparrow$, and $der^\uparrow_A = id_A$. 

**Proposition.** In $G$, for every $A$, the operation $A \Rightarrow \_ :$

$B \rightarrow (A \Rightarrow B)$

$f : (B \rightarrow C) \rightarrow (A \Rightarrow f) : (A \Rightarrow B) \rightarrow (A \Rightarrow C)$

where $A \Rightarrow f$ defined by

$\epsilon \cup \{ t \cdot A \Rightarrow B \cdot A \Rightarrow C, \ s \mid t \in L_{(A \Rightarrow B) \Rightarrow (A \Rightarrow C)} \wedge s \mid \gamma A \Rightarrow A \in id_A, s \mid \gamma B \Rightarrow C \in f \wedge P(s \mid \gamma B \Rightarrow C) = P(s) \}$

is a functor from $G$ to $G_{sst}$

Note that for every $f$ and $A$, $A \Rightarrow f$ is strongly total.

**Lemma.** $A \Rightarrow f$ is a strategy.

**Proof.** The proof goes just the same as for $A \times f$. 

**Lemma.** $A \Rightarrow \_ is a functor.

**Proof.** The proof that it is indeed a functor is sensibly the same as for $A \times \_$ It is straightforward to see that the yield strategy is indeed strongly total and satisfies the no-read condition. Moreover, the fact that $F(s) \mid A \in id_A$ ensures the uniqueness of the thread.

**Theorem 10.** $A \Rightarrow \_ is right adjoint to $(\_)^\uparrow \otimes A$. Equivalently,
\[ G_{ssl}(B, A \Rightarrow C) \simeq G(B \otimes A, C) \]

and the bijection is natural in \( B \) and \( C \).

**Proof.** Let \( \Lambda_{A,B,C} : G_{ssl}(A, B \Rightarrow C) \to G(A \otimes B, C) \)

\[ \Lambda_{A,B,C}(i_A^{S_a}, *_{S_a}) = \epsilon \]

\[ (i_A^{S_a} \cdot *_{S_a} i_B^{S_a} s) = (i_A, i_B)^{S_a} f(s) \]

where,

\[ f : \text{inl}(m) \mapsto \text{inl}(m) \mid \text{inr}(\text{inl}(m)) \mapsto \text{inl}(\text{inr}(m)) \mid \text{inr}(\text{inr}(m)) \mapsto \text{inr}(m), \]

and a justification pointer is added from the \((i_A, i_B)\) to the initial move of \( C \). Then \( \Lambda_{A,B,C} \) is a bijection, with \( \Lambda_{A,B,C}^{-1}(i_A^{S_a}, *_{S_a} i_B^{S_a}) = i_A^{S'_a} \cdot *_{S'_a} i_B^{S_a} f^{-1}(s) \), with \( S' \) such that \( i_A^{S'_a} = \gamma(i_A^{S_a}) \). and \( \Lambda_{A,B,C}^{-1}(i_A^{S'_a}, *_{S'_a} i_B^{S_a} | i_A \in I_A) = \epsilon. \)

Proof of the naturality missing. \( \square \)

### 4.6 Instantiation and recursive type

#### 4.6.1 Structure of the sum

**Theorem 11.** \( \oplus \) is a coproduct in \( G_C \)

**Proof.** proof missing \( \square \)

**Lemma.** For every \( A \), we can form the functor \( A \oplus - \) and \( - \oplus A \). For \( \sigma : B \to C \), we define :

\[ A \oplus \sigma = \{ s \in P_{(A \rightarrow B) \rightarrow (A \rightarrow C)} \mid s \upharpoonright (A \rightarrow A) \in id_A \wedge s \upharpoonright (B \rightarrow C) \in \sigma \} \]

**Proof.** Proof missing \( \square \)

**Lemma.**

- \(- \oplus - \) is a bifunctor

\[ \sigma \oplus \tau = \{ s \in P_{(A \rightarrow B) \rightarrow (B \rightarrow D)} \mid s \upharpoonright (A \rightarrow B) \in \sigma \wedge s \upharpoonright (C \rightarrow D) \in \tau \} \]

- The empty arena is a unit for the \( \oplus \) functor. That is, there are natural isomorphisms \( A \oplus \emptyset \simeq \emptyset \oplus A \simeq A \).

- \((G_c, \oplus, \emptyset)\) is a monoidal category.

**Proof.** missing. \( \square \)

37
4.6.2 Operations are continuous

We would like to establish the correspondence:

Type with $X$ as free variable $\Rightarrow$ Contunuous operation on $G_c$
Continuous functor on $G_{sst}$

For that, we need to prove that each operations that we have defined so far on arena is a monotone continuous operation.

**Theorem 12.** $\oplus, \times, \Rightarrow$ are continuous operations in $G_c$.

*Proof.* missing \hfill $\square$

4.6.3 Solving the recursive arena equation-the plan

We want an arena for $\mu X.F(X)$, where $F$ is a formula/type formed by induction over $X, \mathbb{N}, 1, \text{ and } \oplus, \otimes, \to$.

There are different approaches to this problem. First, we look at the structure of the moves of the arena $F(A)$, and show that the instantiation is functorial. We do that by plugging each the arena into $A$ into $F(X)$ every time there is a $X$, and look at the solution. We use the result to gives a exhaustive description of the arena $\mu X.F(X)$, by plugging the arena $F(X)$ into itself, and repetting the process. We then prove that the arena created beforehand is equated by the union of the n-application of $F$ on the empty arena, giving us the proof that indeed the arena is the least fix point on $F$ when looked at it as a class function on arenas. On a third part, we prove that this solution is indeed a minimal invariant for $F$ when looked at it as a functor in the category $G_{sst}$ that is pre-CPO-enriched.

4.6.4 The semantic of instantiation

We introduce the notion of extended pattern. An extended pattern is defined by induction:

$$p, p' = \_ | \text{inr}(p) | \text{inl}(p) | \langle m, p \rangle | \langle m, p \rangle | \langle p, p' \rangle.$$ Where $m$ is a move.

An extended pattern can take several moves or pattern as input. We write $p_e(A)$ to say that all these input are in the set $A$, or $p_e(X)$ to say they are all $X$. For $p_e$ an extended pattern, and $p$ a pattern, we say that $p \in p_e \uparrow X$ if, by filling all the holes of $p_e$ expect one by $X$, we can obtain the pattern $p$. As one can notice, explore inside a pair can change the structure of the pattern. For example, for the type $(A \to B) \times C$, if we explore the left hand side, the pattern for the initial move will be $(\_, i_c)$ whereas the pattern for the second move (in A) would be $\text{inl}(\text{inl}(\_))$. We automatize this transition.
from one pattern to another by the function on pattern $\pi$. We define $\pi$ by recursion on pattern.

$$
\begin{align*}
\pi(\epsilon) &= \epsilon \\
\pi(p(\langle i, m \rangle)) &= \pi(p)(\text{inl}_-) \\
\pi(p(\text{inl}_-)) &= \pi(p)\text{inl}_- \\
\pi(p(\text{inr}_-)) &= \pi(p)\text{inr}_-
\end{align*}
$$

**Lemma.** Let $m$ in $F(X)$ be a move such that $X \in m$. Then $\exists p_e$ an extended pattern such that $p_e(X, X, \ldots, X) = m$ and $X$ does not appears in $p_e$.

**Proof.** As $X \in m$, $\exists p$ pattern such that $m = p(X)$. Now we proceed by recursion on the number of time $X$ appears in the pattern. If 1 then we got the desired result. Suppose we got the result for $n$, and $X$ appears $n+1$ times in the pattern. Then, the only way a name can appears several times if is there is a product at some point. Imagine $m = p(m', m'')$. Then if either $m$ or $m'$ contains the $n+1$ $X$, we explore it to the next pair, until there is at least one $X$ in each side of the pair. Then we got, using the recursion hypothesis, $m = p'(p''(X), p'''(X))$. where $p'', p''' \in P_e$. Then $p'(p'', p''')$ in $P_e$ and satisfies the theorem.

**Lemma.** Let $p$ a pattern such that $p(X) \in M_{F(X)}$ then $X \notin \pi(p)$.

**Proof.** missing

Now, given $F$ functor on type defined as above, we would like to investigate the structure of the arena $F(A)$ where $A$ is a random arena, given the structure of $A$, and the structure of $F(X)$.

**Theorem 13.**

1. Let $P_e$ be the set of extended pattern $p_e$ such that $p_e(X) \in F(X)$ and $X \notin p_e$.

2. Let $P'$ be the set of of pattern such that $P(X) \in F(X)$, and $P = \pi P'$.

3. Let $Q$ be the set of moves of $F(X)$ such that $X \notin Q$.

Then $F(A)$ has the following structure

1. $M_{F(A)} = Q \cup P_e(I_A) \cup P(M_A \setminus I_A)$
2. $I_{F(A)} = Q \cap I_F \cup (P_e \cap I_F)(I_A)$
3. $\lambda_{F(A)}(q) = \lambda_{F(X)}(q) \land \lambda_{F(A)}(p_e(i_{A_1}, \ldots, i_{A_n}) = \lambda_{F(X)}(X) \land P_{F(A)}(m) = \lambda_{A}^P(m)\text{if } \lambda_{F(X)}(X) = P$
4. $\lambda_{F(A)}(p)(m) = \lambda_{A}^P(m)\text{otherwise}$.
5. $q \vdash_{F(A)} q' \iff q \vdash_{F(X)} q'$
6. $q \vdash_{F(A)} (p_e(i_{A_1}, \ldots, i_{A_n}) \iff q \vdash_{F(X)} p_e(X)$
7. $p(m) \vdash_{F(A)} p(m) \iff m \vdash_A m'$
8. $P_e(i_{A_1}, \ldots, i_{A_n}) \vdash_{F(A)} p(q)$ if $p = p_e \land i_{A_j} \vdash_A q$
Proof. The only way the author thinks this can be proved is by a really tedious induction on the form of the type, where we prove that this formula is stable. To do that, we decompose $F$ into a number of $F_i$, $F = \circ_{i=1..n} F_i$, such that $F_i$ are atomic functor, that is amoung the following list.

$$F_{\text{atomic}} = X + A | A + X | A \times X | X \times A | X \times X | X \to A | A \to X | X \to X$$ (1)

where $X$ does not appear in $A$. Then if we prove the following formula to be sable under each application of $F_i \in F_{\text{atomic}}$, we prove it hold for $F$. Just for sake of showing how this work, we will do the proof in the case $F(X) = \langle A, G(X) \rangle$. Then $M_F(B) = I_A \times I_{G(B)} \cup \text{inl}(A \setminus I_A) \cup \text{inr}(G(B) \setminus I_{F(B)})$. We suppose $X$ does not appear in $A$. Let investigate first what is the set $Q_F$. We can see that $Q_F = I_A \times (I_G \cap Q_G) \cup \text{inl}(M_A \setminus I_A) \cup \text{inr}(Q_G \setminus I_G)$.

On the other hand $P^F = (I_A, P^G \cap I_G) \cup \text{inr}(P^G \cap I_G^c)$, and $P_F = \text{inr}(P_G)$. Then, by using the recursion hypothesis on both $M_G$ and $I_G$ and put it in the formulas we obtained the desired result. The $\lambda$ part comes from the lemma, and the $\vdash$ from the induction hypothesis, and the fact that $(i_a, i_b) \vdash \text{inr}(m)$ if $i_b \vdash_{G(B)} m$. \qed

Let us note that we can not define an section retraction pair from $F(X)$ to $F(A)$, even if it is almost the case. There is indeed a strategy from $F(X)$ to $F(A)$, when $A$ is not empty, that plays copycat for move not related to $X$, and that plays an initial move of $A$ instead of $(X, a)$ whenever $X$ appears in the structure of the move. We can also define a section from $F(A)$ to $F(X)$ by playing twice the same name distinguisher for $X$ whenever the initial moves are the same. However, we can not ensure that the composite is the identity of $F(X)$. Such a pair can be defined only in certain case, such as $A = \mathbb{N}$, where we can encode what was the name distinguisher used inside $\mathbb{N}$. We do that more explicitely in the next paragraph.

A function $f : (X, N) \mapsto I_A$ can be extended to move simply by defining $f(p_e(m)) = p_e(f(m))$ where $p_e$ is an extended pattern such that $\text{dom}(f) \cap p_e = \emptyset$. Now, for such a function, and $s$ a sequence $A \mapsto B$, we define $f \upharpoonright B(s)$ by applying $f$ only to moves that belong in $B$, and letting the $A$ part unchanged.

$$e_{F(X) \mapsto F(A)} : F(X) \to F(A) = \{ f \upharpoonright F(A)(s) \mid s \in \text{id}_{F(X)}, f : (x, N) \mapsto I_A \}.$$  

$$p_{F(A) \mapsto F(X)} : F(A) \to F(X) = \{ g \upharpoonright F(X)(s) \mid s \in \text{id}_{F(A)}, s \cap M_A \subseteq I_A, g : I_A \mapsto (X, (N)) \}.$$  

For the $A = \mathbb{N}$ case we can choose a $f$ to be a bijection between $(X, N)$ and $I_{\mathbb{N}} = \mathbb{N}$, and $g$ to be its inverse. In this case we obtain a section retraction pair.

$F$ is now defined on objects, and we would like to extend this definition to morphisms in order to define a functor.

Definition. Let $\sigma : F(X) \mapsto G(X)$. We define $\sigma_A : F(A) \mapsto G(A)$ by doing copycat between the different copies of $A$ that have been instatied using the same name.
\[
\sigma_A = \{ s \in L_{F(A)} \rightarrow G(A) \mid \exists t \in \sigma, \exists f : (X, N) \rightarrow I_A, \text{ such that } \\
\quad s \upharpoonright \gamma (F(I_A) \rightarrow G(I_A)) = f(t) \\
\quad \land p(X, a) \in t, p'(X, a) \in t \Rightarrow s \upharpoonright \gamma p, p' \in \text{id}_A \\
\forall s' \leq s, P(s' \upharpoonright \gamma (F(\emptyset) \rightarrow G(\emptyset)) = P(s) \} 
\]

**Proposition.** Instantiation is functorial. That is for every object \( A, \) and \( \sigma, \tau \) such that the co-domain of \( \sigma \) is the domain of \( \tau \). we have

\[
\sigma_A; \tau_a = (\sigma; \tau)_A.
\]

**Proof.** The proof goes the usual way. Suppose \( \sigma : F(X) \rightarrow G(X), \tau : G(X) \rightarrow H(X) \)
First, we prove \( \sigma_A; \tau_A \subseteq (\sigma; \tau)_A \). Let \( s \in \sigma_A; \tau_A \), and \( s' \in \sigma_A \mid \tau_A \) such that \( s' \upharpoonright F(A) \rightarrow H(A) = s \). As \( s' \upharpoonright \gamma F(A) \rightarrow G(A) \in \sigma_A \), let us called \( f_\sigma \) and \( t_\sigma \) the \( t \) and \( f \) of the definition. We define as well \( t_\tau \) and \( f_\tau \). Note that we can choose \( t_\sigma \) and \( t_\tau \) such that \( t_\sigma \upharpoonright B = t_\tau \upharpoonright B \), and \( ((\nu(t_\sigma) \setminus \nu(t_\sigma \upharpoonright \gamma B)) \cap (N)) \cap ((\nu(t_\tau) \setminus \nu(t_\tau \upharpoonright \gamma B)) \cap (N)) = \emptyset \). In that case, we will have \( f_\sigma(\upharpoonright t_\sigma \cap G(X) \cap (X, N)) = f_\tau(\upharpoonright t_\tau \cap (G(X) \cap (X, N))) \). Then by composing the \( t_\sigma \) and \( t_\tau \), we obtain an interaction sequence for \( \sigma \mid \tau \). The \( \wedge \) part -names and store conditions about \( t \) comes from \( s' \), and the \( N \) from the fact that \( P(t_\tau) \cap N = P(t_\sigma) \cap N = \emptyset \). The copycat property between instation that have the same name distinguisher is conserved, as both equality and copycat are transitive. Conservation of \( \forall s' \leq s, P(s' \upharpoonright \gamma (F(\emptyset) \rightarrow G(\emptyset)) = P(s) \) is proved just as we did for \( \odot \).

Now, we prove the reverse inclusion, that is \( (\sigma; \tau)_A \subseteq \sigma_A; \tau_A \). For \( s \in (\sigma; \tau)_A \), there is \( t' \in \sigma \mid \tau \) such that \( t' \upharpoonright F(X) \rightarrow H(X) = t \). Now from \( s \upharpoonright G(A), s \upharpoonright A, t' \upharpoonright F(X) \rightarrow G(X) \) and \( f_{\sigma, \tau} \) we can almost construct a sequence in \( \sigma_A \). Let remind that player can not introduce new names in \( N \), so there is no risk that some \((X, a)\) appears in \( t' \upharpoonright \gamma F(X) \rightarrow G(X) \) and not in \( t' \upharpoonright \gamma F(X) \rightarrow H(X) \). An other problem might be that \( P(s' \upharpoonright (F(\emptyset) \rightarrow G(\emptyset)) \neq P(s) \). We define \( S \), the set that cause problem, and apply the usual transformation using a name permutation on \( t' \) to get rid of the problem. Finally, once we have obtained the two sequences in \( \sigma_A \) and \( \tau_A \), such that the names conditions are respected, we apply the \( \circ \) transformation to obtain the interaction sequence, just as in ..., and get the required inclusion. \( \square \)

### 4.6.5 Exhaustive description of \( \mu X.F(X) \)

For \( P \) a set of pattern, we define \( P^* \) to be the set of finite list of pattern of \( P \), including the empty list. If \( P \) is a set of extended pattern, we write \( P^* \) for the set of extended pattern build recursively by adding pattern of \( P \) into the holes of pattern of \( P^* \), or equivalently for the set of finite tree whose node are labelled by patterns of \( P \) and the arity of each node corresponds to the arity of the extended pattern. We write \( p^* \) for a element of the set \( P^* \).

We split the moves of \( F(X) \) into distinct sets.
• Let $P_e = \{ p_e | p_e \text{ extended pattern } \land p_e(X) \in M_{F(X)} \land X \notin p_e \}$. 
• Let $P_e^i = \{ p_e | p_e \text{ extended pattern } \land p_e(X) \in I_{F(X)} \land X \notin p_e \}$. 
• Let $Q_j = \{ p(m) | p \text{ pattern, } m \text{ atomic move, } \land p(m) \in (M_{F(X)} \setminus I_{F(X)}) \land X \notin p(m) \}$. 
• Let $Q_I = \{ (p(m)) | p \text{ pattern, } m \text{ atomic move, } \land p(m) \in I_{F(X)} \land X \notin p(m) \}$. 
• Let $P = \pi(\{ p | p \text{ pattern such that } p((X,N)) \in M_{F(X)} \})$. 
• Let $P_I = \{ p | p \text{ pattern } \land p((X,N)) \in I_{F(X)} \}$. 
• Let $P_{e}^*(Q_I) = \{ p^*_e(Q_I) | p_e \in P_e \}$. 
• Let $P_{e}^i(Q_I) = \{ p^*_e(Q_I) | p_e \in P_e^i \}$. 

$P_{e}^*(Q_I)$ can be seen as the set of tree whose node are labelled by $P_e$, the leaves by $Q_I$, and the number of edges from one node is the arity of the extended pattern, and $(P_e)^*(Q_I)$ as restriction of this set to those whose nodes are labelled only by the extended patterns that are initial. For $q$ in $P_{e}^*(Q_I)$ we define by $q_{pe}$ its extended pattern part in $P_{e}^*$, and by $p_{p} = \{ q_{pi} | i = 1...n \}$, where $n$ is the arity of $p_{p}$, its set of pattern obtained by erasing just one leaf at the time ( and replacing it by _). Finally, we define $Q = Q_j \cup P_{e}^*(Q_I)$.

1. $M_{\mu X.F(X)} := \{ p^*q | p^* \in P^* \land q \in Q \}$.  
2. $I_{\mu X.F(X)} := P_{e}^*(Q_I)$.  
3. 
   \[ p^*(m) \vdash_{\mu X.F(X)} p'^*(m') \iff ((p^* = p'^* \land (q,q' \in Q_j) \land q \vdash_{F(X)} q')) \lor (p^* = p'^* \land (q \in Q_j \land q' \in P_{e}^*(Q_I)) \land q \vdash_{F(X)} q'(X,N)) \lor (p^* \neq p'^* \land q \in P_e(Q_I) \land p'^* = p'' \land p'' \in P) \land \exists j \leq \text{arity}(q) \land p'' = \pi(q_{pj}), \land q = q_{pi}(i_{F(X)}) \Rightarrow i_{F(X)} \vdash_{F(X)} q') \]

4. $\lambda$ defined by induction by:
   - $\lambda_{\mu X.F(X)}(\pi X m) = \lambda_{F(X)}(m)$ if $m \in F(X)$.  
   - if $\lambda_{\mu X.F(X)}(p(X)) = \{ P, A \}$ then $\lambda_{\mu X.F(X)}(p(m)) = \lambda(m)$  
   - if $\lambda^Q_{\mu X.F(X)}(p(X)) = Q$ then $\lambda^Q_{\mu X.F(X)}(p(m)) = Q$ if $m$ initial ($m \in I_{\mu X.F(X)}$), $\lambda^Q_{\mu X.F(X)}(p(m)) = \lambda_{\mu X.F(X)}(m)$ otherwise.  
   - If $\lambda^O_{\mu X.F(X)}(p(X)) = O$ then $\lambda^O_{\mu X.F(X)}(p(m)) = \lambda_{\mu X.F(X)}(m)$. Otherwise $\lambda^O_{\mu X.F(X)}(p(m)) = \lambda_{\mu X.F(X)}(m)$.  

42
Let us call \( U \) the arena defined in the section below and \( V \) the arena \( \bigcup_{i \in \mathbb{N}} F^i(\bot) \). In order to prove that indeed \( U = V \), it is enough to prove that \( \forall i, F^i(\bot) \subseteq V \) and reciprocally that \( \forall m \in V \exists i, m \in F^i(\bot) \).

1. We do the first part by induction. First, let us note that indeed \( F^0(\bot) \subseteq V \). Now let us suppose the property holds until rank \( n \).

2. For \( p \in P^* \), let us denote \( \eta(p) \) the number of pattern of \( P \) in \( p \), and define it using recursion by:

\[
\eta(\emptyset) = 0 \\
\eta(p) = \min\{n \in \mathbb{N} \mid \exists p' \in P, p'' \in P^*, p = p'p'' \land n = \eta(p'') + 1\}
\]

For an extended pattern \( p_e \in P_e^* \), we extend the definition:

\[
\eta(p_e) = \max\{j \in \mathbb{N} \mid \exists i \in \mathbb{N}, i \leq \text{arity}(p_e) \land j = \eta(p_e_i)\}
\]

We claim that a move \( m = pq \) belongs in \( F^{\eta(p)+\eta(q)}(\bot) \)

4.6.6 The arena as a minimal invariant

4.7 Pre-System F structure

4.7.1 The fibration and the adjunctions

For \( X_1, X_2, \ldots, X_n \) in \( TVAR \), we define the category \( G(X_1, \ldots, X_n) \) that has as objects arenas whose free variable are inside \( (X_1, \ldots, X_n) \).