Tutorial on concurrent games

Simon Castellan

March 24, 2015

Contents

Introduction. This document is a presentation of (linear) concurrent games which are a new foundation for game semantics based on event structures. The framework allows

for strategies to be event structures thus permitting to model behaviours (for instance concurrency and non-determinism), and for games to be event structures to have more general games that represent more behaviours.

1 Games and pre-strategies

To allow for more complex rules in the games, our games will be modelled by event structures, the so-called "concurrent analogue of tree". By featuring concurrency and non-determinism they allow to express more complex behaviours than for instance usual arenas in HO games.

1.1 Event structures

***Definition* 1.** — *An* event structure *is a tuple* $(A, \leq, \mathrm{Con}_A)$ *where* $(A, \leq)$ *is a partial order of* events *and* $\mathrm{Con}_A \subseteq \mathscr{P}_f(A)$ *is a set of* finite *subsets of* A *called the* consistent *subset of* A*, satisfying the following axioms:*

- Con_A *is down-closed with respect to inclusion (a subset of a consistent set is consistent)*
- *Singleton sets are consistent*
- *Whenever* X *is consistent in* A*, the down-closure of* X*,*

$$[X]_A = \{a \in A \mid \exists a' \in X, a \leq a'\}$$

  is a consistent set.

- *The set* $[a] = [\{a\}]$ *is finite for every* $a \in A$*.*

The order of an event structure represent causality: $a \leq a'$ whenever for a' to happen, a must have occurred before. In particular, if $a_0 \leq a$ and $a_1 \leq a$, then the occurrence of both a_0 and a_1 is necessary for that of a. If $\mathrm{Con}_A = \mathscr{P}(A)$, A is called *an elementary event structure*.

Two events that are consistent and not causally related are said to be *concurrent*. Concurrent events can occur together in any order.

Configurations. We define a *configuration* of A to be a finite set of events of A down-closed for $\leq_A$ and consistent. The set of configurations of A will be written $\mathscr{C}(A)$. The two last axioms imply that for any $a \in A$, $[a]$ is a configuration of A, called a *prime configuration* of A. We will write $[a)$ for $[a] \setminus a$ which is also a configuration of A. Configurations are naturally ordered by the inclusion.

Causalities. From $\leq_A$, we extract the relation $\rightarrowtriangle_A$ which is the least relation whose transitive closure is $\leq_A$, in other terms $a \rightarrowtriangle_A a'$ if and only if $a < a'$ and there is no

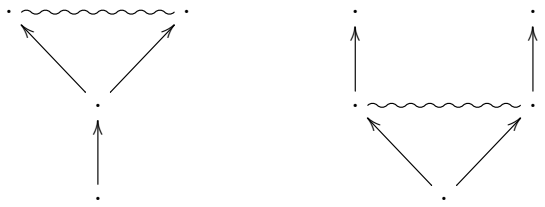
$c \in A$ such that $a < c < a'$. A pair (a, a') such that $a \rightarrowtriangle a'$ will be called a *causality* of A. Moreover, if $x \in \mathscr{C}(A)$ and $a \in A$ such that $x \cup \{a\} \in \mathscr{C}(A)$ without $a \in x$, we say that x extends by a and we write $x \overset{a}{\text{—}\subset}$.

Event structures with binary conflict. An event structure $(A, \leq_A, \mathrm{Con}_A)$ is said to have *binary conflict* when Con_A is generated by a binary relation $\sharp \subseteq A^2$ in the following way:

$$\mathrm{Con}_A = \{X \in \mathscr{P}(A) \mid \forall a_1, a_2 \in X, \neg(a_1 \sharp a_2)\}$$

The axioms on Con_S are equivalent to asking that $\sharp$ is irreflexive, and if $a \sharp a'$ with $a' \leq a''$ then $a \sharp a''$ (the conflict is said to be inherited). A minimal conflict (notation $a \sim\!\!\sim a'$) is a conflict which is not inherited, meaning that $a \sharp a'$, along with $[a, a'] \setminus \{a\}$ and $[a, a'] \setminus \{a'\}$ being consistent.

Drawing event structures with binary conflict. The only event structures we will draw have binary conflict. To draw such event structures, we will draw $\rightarrowtriangle$ and $\sim\!\!\sim$ (from which $\leq$ and $\sharp$ are easily recovered). For instance here is the classic example of coffee machine:

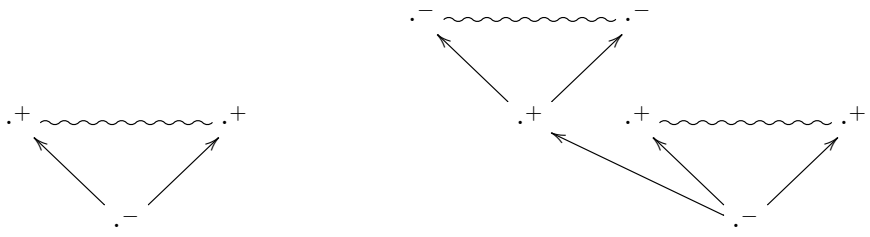


When events do not have meaningful names we draw them as $\cdot$. In the following we only draw examples that have binary conflict however the theory does **not** restrict to that case.

1.2 Games

***Definition* 2. —** *A game, or* event structure with polarities *is a pair* $(A, \lambda_A : A \to \{\ominus, \oplus\})$.

Games are drawn similarly to event structures with the polarities drawn in exponent of the events: for instance, here are the arenas Bool and Bool $\Rightarrow$ Bool of HO game semantics:



We also will write $\oplus$ for $\cdot^+$ and $\ominus$ for $\cdot^-$. By reversing the polarities of a game A, one gets the dual game $A^\perp$.

Parallel composition. Given two event structures A and B we can form the event structure $A \parallel B$ (that extends the usual product of arenas) defined as follows:

- Events: $\{1\} \times A \cup \{2\} \times B$ (the disjoint union of A and B)
- Causal order: $(i, a) \leq (j, b)$ iff $i = j = 1$ and $a \leq_A b$ or $i = j = 2$ and $a \leq_B b$
- Consistency: $X \in \mathrm{Con}_{A\parallel B}$ iff $\{a \in A \mid (1, a) \in X\} \in \mathrm{Con}_A$ and $\{b \in B \mid (2, b) \in X\} \in \mathrm{Con}_B$.

In a binary conflict $(i, a)\sharp(j, a')$ if and only if $i = j$ and $a\sharp a'$ in the corresponding component. A direct consequence of this definition is the fact that configurations of the parallel composition are made of a configuration of each component:

***Lemma* 1.** — *Let A and B be event structures. There is an order-isomorphism*

$$(\mathscr{C}(A \parallel B), \subseteq) \cong (\mathscr{C}(A) \times \mathscr{C}(B), \subseteq \times \subseteq)$$

Proof. The bijection sends a configuration $z \in \mathscr{C}(A \parallel B)$ to (x, y) where $x = \{a \in A \mid (1, a) \in z\}$ and $y = \{b \in B \mid (2, b) \in z\}$ and its inverse sends a pair (x, y) to the configuration

$$z = \{(1, a) \mid a \in x\} \cup \{(2, b) \mid b \in y\}$$

It is routine to check those define the desired isomorphism. □

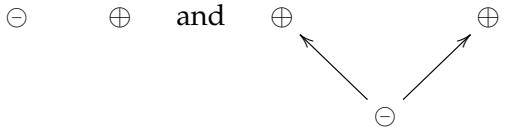
Through this isomorphism we will often identify configurations of $A \parallel B$ as pairs of configurations of A abd B. If A and B are game, we define $\lambda_{A\parallel B}(1, a) = \lambda(a)$ and $\lambda_{A\parallel B}(b) = \lambda_B(b)$ so that $A \parallel B$ is a game.

Now that we defined what games are, we explain how we play on them.

1.3 Simple pre-strategies

Usually, a strategy is described as a set of plays satisfying some closure properties. In a sequential settings, plays are sequences of moves, but as noticed here a better notion of plays are *configurations* of a game. A first definition of strategy would be then: a set of configurations of the game, down-closed for the inclusion. Doing so would lose us direct access to causality. A better approach is: wonder what we want allow our strategies to do, in a concurrent world.

For instance, what could be the possible strategies on the following games:



Intuitively for the first game, there are three possible behaviours:

- play the $\oplus$ no matter what Opponent does
- play the $\oplus$ after Opponent played his $\ominus$ event, hence adding a causality $\ominus \to \oplus$
- do nothing

For the second game, since causalities are already in place, there is not much we can do:

- play as the game specifies: wait for Opponent and then answer concurrently the two $\oplus$
- wait for Opponent, and then only answer one of the $\oplus$ thus dropping the other one
- wait for Opponent, and then make a non-deterministic choice between the two positive events to play (adding a minimal conflict to the game)
- do nothing

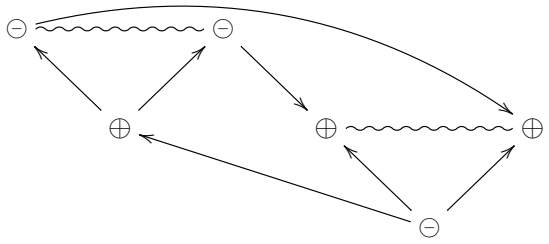
Thus a strategy should be allowed to:

- ignore some positive events
- add some causalities
- add minimal conflict.

This can be formalized to define a *simple pre-strategy* on a game A as an event structure $(S, \leq_S, \mathrm{Con}_S)$ satisfying:

- S is a down-closed subset of A (wrt $\leq_A$)
- $\leq_S \supseteq \leq_A$ (to add causalities),
- $\mathrm{Con}_S \subseteq \mathrm{Con}_A$ (to add some conflicts).

Simple pre-strategies are easily drawn, just draw the event structure corresponding to it. For instance, here is the simple pre-strategy corresponding to the negation on Bool $\Rightarrow$ Bool.

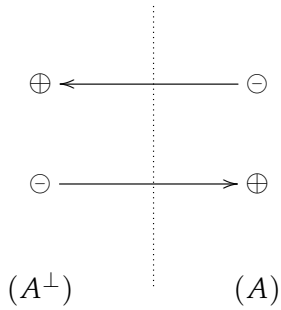


Note that the three conditions on S can be more elegantly put by saying that a configuration of S is a configuration of A.

A very important example of simple pre-strategy is copycat. Define $\mathbb{CC}_A$ to be the following event structure:

- Events: those of $A^\perp \parallel A$
- Causal order: $\leq_{\mathbb{CC}_A} = (\leq_{\mathbb{CC}_A} \cup \rightarrow)^*$ where $\rightarrow$ is the relation defined as:
  - $(1, a) \rightarrow (2, a)$ when a is positive in A
  - $(2, a) \rightarrow (1, a)$ otherwise
- Consistency: that of $A^\perp \parallel A$.

It is easy to see that $\mathbb{CC}_A$ satisfies all the condition of a simple pre-strategy written $\mathbf{c}_A$ playing on $A^\perp \parallel A$. Copycat is a pre-strategy forwarding negative moves from one side to the corresponding positive move on the other side. For instance, copycat on $\ominus \parallel \oplus$ gives:



A very important property that is going to be useful later on, is that copycat does not add immediate causal link inside A or $A^\perp$ – all extra links are between A and $A^\perp$.

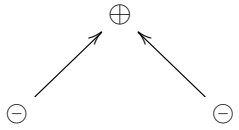
1.4 Pre-strategies on game A

However, there is a problem with this definition. What about strategies on the game:

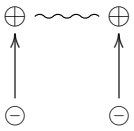
$\oplus \qquad \ominus \qquad \ominus$

Among the behaviours we want to have, are: "I want to play $\oplus$ whenever Opponent plays **both** negative events" and "I want to play $\oplus$ as *soon* as Opponent plays one his events".

The first behaviour correspond to the simple strategy:



What about the second? Because of the nature of the order, there is no way to say a depends on b **or** c. But this process contains a race: because the two Opponent events are received concurrently, two threads have to race to send the $\oplus$ event. It is a race, because we want to send only one $\oplus$ event. Making this intuition clear yields the following event structure with polarity:



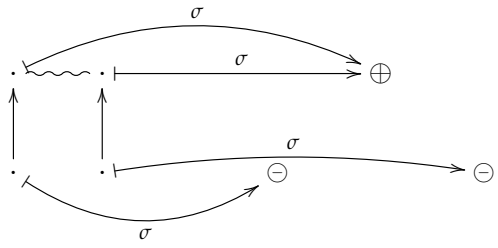
To account for that phenomenon, we define a pre-strategy as an event structure with polarity S, along with a projection function $\sigma : S \to A$. The function should satisfy that $x \in \mathscr{C}(S)$ implies $\sigma x \in \mathscr{C}(A)$ as before. Moreover, it should satisfy an injectivity condition (since an inclusion can be seen as an injection), however in our case the function is not injective since the two $\oplus$ of S are sent to the unique $\oplus$ of A. Therefore we only ask for *local* injectivity, ie. σ restricted to a configuration is injective. It turns out these maps between event structures are well-known:

***Definition* 3.** — *Let A and B be event structures. A map of event structures is a function $f : A \to B$ satisfying:*

- *For $x \in \mathscr{C}(A)$, $fx \in \mathscr{C}(B)$*
- *f restricted to a configuration is injective.*

A pre-strategy on a game A is an event structure S along with a map of event structures $\sigma : S \to A$.

Note that σ induces a unique choice of polarities on S such that σ preserves them. We can represent the second behaviour as the following pre-strategy:



In the rest of the article, to draw a pre-strategy $\sigma : S \to A$, we will draw the event structure S and instead of the $\cdot$, for each event s we will draw its projection in the game σs. The strategy will be denoted by σ, and S will be called its underlying event structure.

A simple pre-strategy is just a pre-strategy of the form $\mathrm{id}_{A'} : A' \to A$. Note that given pre-strategies $\sigma : S \to A$ and $\tau : T \to B$, we have a pre-strategy $\sigma \parallel \tau : S \parallel T \to A \parallel B$ defined naturally as $(\sigma \parallel \tau)(1, s) = \sigma s$ and $(\sigma \parallel \tau)(2, s) = \tau s$.

The category of event structures. Event structures and their maps form a category written $\mathcal{ES}$. Because of the local injectivity constraint, it does not have a terminal object. But it has pullbacks and products as we will see later. It also has sums (but not coequalizers), but we will not make use of that.

On maps of event structures. There is no condition in the definition of maps of event structures for preservation of order or conflict. However, because of the requirement on configurations, a map of event structures preserves consistency (thus reflects conflicts) and reflects the order on consistent events: for all consistent $a, a' \in A$ such that $fa \leq_B fa'$ then $a \leq_A a'$.

Isomorphism of pre-strategies. Since our pre-strategy can choose freely the S, we don't want to identify strategies by equality of their underlying event structures and their maps. Unless, we ask that there is an isomorphism (in the category of event structures) between the underlying event structures that commutes with the projection on the game. Two strategies $\sigma : S \to A$ and $\tau : T \to A$ are isomorphic whenever there exists an isomorphism $\varphi : S \cong T$ in $\mathcal{ES}$ such that $\tau \circ \varphi = \sigma$.

To prove isomorphism of event structures, the following lemma will be useful.

***Lemma* 2.** — *Let S and T be event structures. There is an order-isomorphism φ between $(\mathscr{C}(S), \subseteq)$ and $(\mathscr{C}(T), \subseteq)$, if and only if S and T are isomorphic. Moreover, if $\sigma : S \to A$ and $\tau : T \to A$ are pre-strategies, and φ satisfies $\varphi; \tau = \sigma : \mathscr{C}(S) \to \mathscr{C}(A)$, σ and τ are isomorphic.*

Proof. If S and T are isomorphic it is direct to see that the induced maps on configurations is an order-isomorphism.

Let $\varphi : \mathscr{C}(S) \to \mathscr{C}(T)$ be an order isomorphism. By an easy induction, this implies that φ preserves cardinality of finite sets. Thus given $a \in A$, we have that $\varphi[a] \setminus \varphi[a)$ has cardinal 1, ie. is a singleton we call $f(a)$. It easy to see that $a \mapsto f(a)$ defines an isomorphism $f : S \cong T$.

Moreover, if $\varphi; \tau = \sigma$, it is straightforward to check that we indeed have $f; \tau = \sigma : S \to A$. □

1.5 Toward a category of concurrent games?

Following the usual tradition on game semantics, we define a pre-strategy from a game A to a game B as a pre-strategy on the game $A \multimap B = A^{\perp} \parallel B$. (Note that there is no need

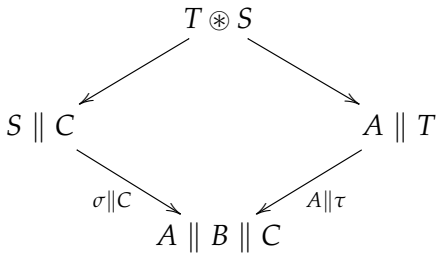
to make the initial events of A depend on those of B as we are not working in a polarized setting).

We have seen how to define the pre-strategy $\mathfrak{c}_A : \mathbb{C}_A \to A^\perp \parallel A$ that can be seen as a strategy from A to A that will be our identity. We now need to define the composition of two strategies and explain how they interact with each other.

Let $\sigma : S \to A^\perp \parallel B$ and $\tau : T \to B^\perp \parallel C$ be composable pre-strategies. We need to explain how to form their composition, denoted $\tau \odot \sigma$. As usual in game semantics the composition will be defined in two steps:

1. Interaction of τ against σ over the game B
2. Hiding of the "internal events", ie. those sent to B

Hiding is an easy operation to define on event structures, it corresponds to *projection*. Interaction though is more subtle to define, as it is usually defined, on sequences. The trick is to consider the following pullback (forgetting about polarities):



which exactly captures what the interaction should be (it will be made clear later on).

2 Pullback of event structures and stable families

2.1 Pullback of deterministic maps

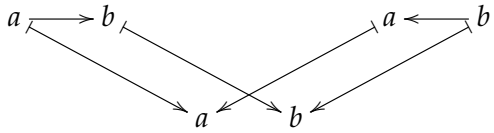
In this section, we consider two maps of event structures $f : A \to C$ and $g : B \to C$ whose pullback we would like to build. A pullback would a triple (P, π_1, π_2) with $\pi_1 : P \to A$ and $\pi_2 : P \to B$ satisfying the commutation $f \circ \pi_1 = g \circ \pi_2$. A naive answer to that problem would be to consider the pullback of f and g in sets $A \times_C B = \{(a, b) \in A \times B \mid fa = gb\}$. Since π_1 and π_2 need to preserve consistency, we could set:

$$\mathrm{Con}_{A \times_C B} = \{X \in A \times_C B \mid \pi_1 X \in \mathrm{Con}_A \ \& \pi_2 X \in \mathrm{Con}_B\}$$

As for causalities, since π_1 and π_2 need to reflect the order we could merge the causalities of A and B as follows. Define $(a, b) \to (a', b')$ as $\{(a, b), (a', b')\} \in \mathrm{Con}_{A \times_C B}$ and $a <_A a'$ or $b <_B b'$, and write $\leq_{A \times_C B}$ for its transitive and reflexive closure. Note that we

need to be careful to impose that events related by $\to$ are consistent as it is not automatic: we could have $a \leq_A a'$ and $\{b, b'\} \notin \mathrm{Con}_B$.

This fails to be an order – it might not be antisymmetric as $\to$ could have loops as in the following picture:



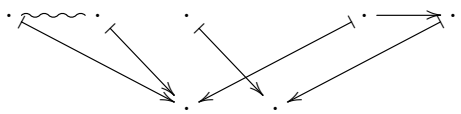
We need to restrict $A \times_C B$ to the subset of elements that are not involved is loops. An element $e \in A \times_C B$ is said to be *well-founded* if there is no infinite descending sequence $e = e_1 \leftarrow e_2 \leftarrow e_3 \leftarrow \ldots$. Call P the subset of well-founded elements of $A \times_C B$.

***Lemma* 3. —** *The restriction $\leq_P$ of $\leq_{A \times_C B}$ to P is a partial order on P such that for every $p \in P$, $[p]$ is finite.*

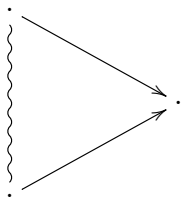
Proof. It is clear that $\leq_P$ is reflexive and transitive. Assume we have $p \leq_P p'$ and $p' \leq_P p$. This means that $p \to^* p' \to^*$. If one of those two sequences is non-empty, then we could create an infinite sequence $p \to^+ p \to^+ \to \ldots$ thus contradicting the well-foundedness of p.

Assume now $p \in P$. We know that every desceding path starting from p is finite – we can thus reason by induction on the maximal length of such a sequence. Call X the set of immediate predecessor of p for $\to$. We know that for each $a \in X$, $[a]$ must be finite by induction. Thus we have only to check that there is a finite number of them. But for each $a \in X$ we have either $\pi_1 a \leq \pi_1 p$ or $\pi_2 a \leq \pi_2 p$, thus. Thus either $\pi_1 a \in [\pi_1 p]$ or $\pi_2 a \in [\pi_2 p]$ both finite sets. Since π_1 and π_2 must be injective on $[p]$, we must have that X is finite. □

We let $\mathrm{Con}_P = \mathrm{Con}_{A \times_C B} \cap \mathscr{P}(P)$. However, P fails to be an event structure in the general case. Consider the following situation:



Computing P yields the following "event structure".



It is not an event structure because the down-closure of the maximal event is not a configuration. It turns out that there is no easy way to fix that problem in order to have an event structure. Causality and consistency needs to be inductively defined. However if A and B have no inconsistent sets, then this the pullback:

***Lemme* 1.** — *If* $\mathrm{Con}_A = \mathscr{P}(A)$ *and* $\mathrm{Con}_B = \mathscr{P}(B)$ *then* P *is an event structure and* (P, π_1, π_2) *is a pullback.*

Proof. We have checked already that prime configurations are finite and since we don't have inconsistency this the only thing to check. Clearly $f \circ \pi_1 = g \circ \pi_2$. Now assume we have an event structure X with two maps $\varphi : X \to A$ and $\psi : X \to B$. Because $A \times_C B$ is a pullback in set we get a unique function $h : X \to A \times_C B$ that makes the triangles commute. The only need left to check is that h's range is actually P. First we show that $h(a) \to h(a')$ implies $a \leq a'$ if a and a' are consistent. Indeed assume for instance $\pi_1(h(a)) \leq \pi_1(h(a'))$. This means that $\varphi a \leq \varphi a'$ and then $a \leq a'$ because φ is a map of event structures.

Assume $a \in X$ is such that $h(a)$ is not well-founded, ie. $h(a) = p_1 \leftarrow p_2 \leftarrow \ldots$. Because $h[a]$ is a configuration, there exists $p_i \in [a]$ such that $h(p_i) = a_i$. All the a_i are consistent, thus by the previous remark, $a \geq a_1 \geq \ldots$ which is absurd because X is an event structure. □

Computing the pullbacks of general event structures will be more elegantly done by doing in a broader space, that of *stable families* that include the previous example.

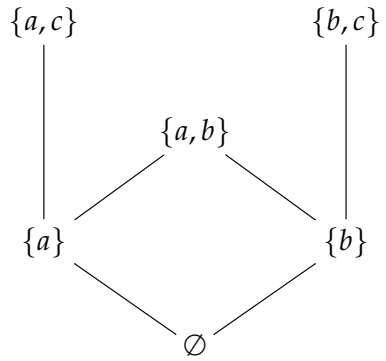
2.2 Stable families

Stable families are a generalization of event structures allowing for causal disjunctives. They generalize the configuration domains of an event structure.

***Definition* 4.** — *A stable family is a pair* $(A, \mathcal{F}_A)$ *where* A *is a set, and* $\mathcal{F}_A \subseteq \mathscr{P}_f(A)$ *(called the set of configurations of A) such that:*

- Completeness*: Forall* $Z \subseteq \mathcal{F}_A$*, if* Z *is bounded then* $\bigcup Z \in \mathcal{F}_A$
- Stability*: Forall* $Z \subseteq \mathcal{F}_A$*, if* Z *is non-empty and bounded, then* $\bigcap Z \in \mathcal{F}_A$
- Coincidence-freeness*: Forall* $x \in \mathcal{F}$*, and* $e, e' \in x$ *distinct, there exists* $y \in \mathcal{F}_A$ *included in* x *such that* y *contains exactly one of the events* e *and* e'*.*
- $A = \bigcup \mathcal{F}_A$

For any event structure A, the pair $(A, \mathscr{C}(A))$ is a stable family, but we obviously have more expressivity power. Consider for instance the following stable family on the set $A = \{a, b, c\}$:



This stable family represents a causal disjunction: c can happen after a or b. There is no way to make A into an event structures such that this is its configuration domain.

Maps of stable families. Let $(A, \mathcal{F}_A)$ and $(B, \mathcal{F}_B)$ be stable families. A map from $(A, \mathcal{F}_A$ to $(B, \mathcal{F}_B)$ is a function $f : A \to B$ such that if $x \in \mathcal{F}_A$ then $fx \in \mathcal{F}_B$ and f restricted to an element of $\mathcal{F}_A$ is injective.

This definition extends the one on event structures and we have a category $\mathcal{SF}$ of stable families.

***Lemma* 4.** — *$A \mapsto (A, \mathscr{C}(A))$ is a full and faithful functor $\mathcal{ES} \to \mathcal{SF}$.*

2.3 Pullback in stable families

In this more general category, we describe how to build pullbacks. Let $f : (A, \mathcal{F}_A) \to (C, \mathcal{F}_C)$ and $g : (B, \mathcal{F}_B) \to (C, \mathcal{F}_C)$ be maps of stable families.

We write $A \times_C B$ for the pullback in set of f and g. On this set we define the stable family $\mathcal{F}$ consisting in those subsets $X \subseteq A \times_C B$ such that:

- $\pi_1 X \in \mathcal{F}_A$ and $\pi_2 X \in \mathcal{F}_B$
- π_1 and π_2 are injective on X
- Forall $e, e' \in X$ distinct, there exists a $Y \subseteq X$ containg one of e and e' and such that $\pi_1 Y \in \mathcal{F}_A$ and $\pi_2 Y \in \mathcal{F}_B$.

The first two requirements ensures that the projections are maps of stable families and the last one that $\mathcal{F}$ is coincidence free. Moreover, coincidence-freeness is what kills deadlock, since for the previous example of a deadlock, $\{(a, a), (b, b)\}$ will not be in $\mathcal{F}$ as it is not coincidence-free (neither $\{(a, a)\}$ nor $\{(b, b)\}$ are in $\mathcal{F}_X$).

***Lemma* 5.** — *$\mathcal{F}$ is a stable family.*

***Proposition* 1.** — *$(\mathcal{F}, \pi_1, \pi_2)$ is a pullback of f and g.*

Proof. By definition of $\mathcal{F}$, π_1 and π_2 are maps of stable families and we have $f \circ \pi_1 = g \circ \pi_2$. Assume we have $\varphi : (Z, \mathcal{F}_Z) \to (A, \mathcal{F}_A)$ and $\psi : (Z, \mathcal{F}_Z) \to (B, \mathcal{F}_B)$ such that $f \circ \varphi = g \circ \psi$.

By the definition of pullbacks, there is a unique map (in **Set**) $h : X \to A \times_C B$ which makes the diagram commute. Thus the only need to check is that it is actually a map of stable families. On a $x \in \mathcal{F}_X$, h has to be injective because $\pi_1 \circ h = \varphi$ and φ is injective on x by definition of maps of stable families. Moreover if $x \in \mathcal{F}_X$, $hx \in \mathcal{F}$ (all properties are easy to show). □

We have seen how stable families help us solve the first problem, what about the second one? Duplication in stable families is not done concretely: we have the same events as for the pullback in sets. But if we look at the stable family we get for the second example (with the duplication) we get a stable family that does not correspond to an event structure. To actually get an event structure, we will need to perform the duplication, and this is what we describe now.

2.4 An adjunction

Finally, how we unfold the causal disjonctions from stable families to event structures by means of an adjunction.

The prime construction. Let $(A, \mathcal{F}_A)$ be a stable family. First we show how to recover the orders on the elements of $\mathcal{F}_A$. Let $x \in \mathcal{F}_A$. We define the order $\leq_x$ on x by:

$$a \leq_x b \text{ iff } \forall y \in \mathcal{F}_A \text{ such that } y \subseteq x, b \in y \Rightarrow a \in y$$

Which means: for all subconfigurations of x, if b appears in it, a must have appeared before. Given $x \in \mathcal{F}_A$ and $a \in x$, we define the prime configuration of a inside x as the down-closure of $\{a\}$ for $\leq_x$:

$$[a]_x = \{b \in x \mid b \leq_x a\}$$

Note that in a stable family the prime configuration associated to an event depends on the ambient history. In the example above, we have $[c]_{\{a,c\}} = \{a, c\}$ and $[c]_{\{b,c\}} = \{b, c\}$.

From the stable family $(A, \mathcal{F}_A)$ we are now ready to define an event structure written $\Pr(A)$ and defined as follows:

- Events: All the prime configuration of the form $[a]_x$ for $a \in x \in \mathcal{F}_A$.
- Causal order: Inclusion of prime configuration
- Consistency: a set of prime configurations X is consistent in $\Pr(A)$ whenever $\bigcup X \in \mathcal{F}_A$

It is easy to check it defines an event structure. Moreover, we have the following property:

***Lemma* 6.** — *Let A be an event structure. There is a (natural) isomorphism $Pr(A, \mathscr{C}(A)) \cong A$.*

Proof. The isomorphisms consists in the map $a \mapsto [a]_{[a]} : A \to \Pr(A, \mathscr{C}(A))$ and $[a]_x \mapsto a : \Pr(A, \mathscr{C}(A)) \to A$. □

It is easy to turn $\Pr(\cdot)$ into a functor $\mathcal{SF} \to \mathcal{ES}$ by letting $\Pr(f)([a]_x) = [fa]_{fx}$ for $f : (A, \mathcal{F}_A) \to (B, \mathcal{F}_B)$.

***Proposition* 2.** — *$Pr(\cdot)$ is right adjoint to $\mathscr{C}(\cdot)$. (And thus $\mathcal{ES}$ is coreflective in $\mathcal{SF}$)*

Proof. We have already described the unit $\mathrm{Id} \to \Pr(\mathscr{C}(\cdot))$. We now describe the counit $\mathscr{C}(\Pr(A)) \to A$ for a stable family $(A, \mathcal{F}_A)$. The maps on events is given by $[a]_x \to a$ and it is easy to check that this is a map of stable family, natural in A and satisfies the required diagrams. □

Moreover, even then $\mathscr{C}(\Pr(\cdot))$ is not naturally isomorphic to Id (because of the duplication – thus the set of events is not the same), we still have this result:

***Lemma* 7.** — *For every stable family $(A, \mathcal{F}_A)$, there is an order-isomorphism*

$$(\mathcal{F}_A, \subseteq) \cong (\mathscr{C}(Pr(A)), \subseteq).$$

Proof. If $x \in \mathcal{F}_A$, then $\{[a]_x \mid a \in x\}$ is a configuration of $\Pr(A)$. Conversely if $x \in \mathscr{C}(\Pr(A))$, then by definition of consistency, $\bigcup x \in \mathcal{F}$. □

2.5 Pullback in event structures

From the adjunction and the existence of pullbacks in $\mathcal{SF}$ we can conclude for pullbacks in $\mathcal{ES}$.

***Lemma* 8.** — *Let $f : A \to C$ and $g : B \to C$ be maps of event structures. The pullback of f and g exists.*

Proof. We have $\mathscr{C}(f) : \mathscr{C}(A) \to \mathscr{C}(C)$ and $\mathscr{C}(g) : \mathscr{C}(B) \to \mathscr{C}(C)$ in $\mathcal{SF}$. It has a pullback (P, π_1, π_2). Because $\Pr(\cdot)$ is a right adjoint, it preserves limits and in particular pullbacks, thus $(\Pr(P), \Pr(\pi_1), \Pr(\pi_2))$ is pullback of $\Pr(\mathscr{C}(f))$ and $\Pr(\mathscr{C}(g))$. By the natural isomorphism $A \cong \Pr(\mathscr{C}(A))$ we get that $\Pr(P)$ is also a pullback of f and g in $\mathcal{ES}$. □

From this abstract definition, we can get a concrete grip on the configurations of a pullback through the notion of *secured bijection*.

***Definition* 5.** — *Let $f : A \to B$ be a bijection between two orders A and B. f is secured when the transitive closure of $\to$ defined as follows is an order. $a \to a'$ when $a \leq a'$ or $f(a) \leq f(a')$.*

A bijection is secured if it there is no deadlock between the orders A and B.

***Proposition* 3.** — *Let $f : A \to C$ and $g : B \to C$ be maps of event structures and let (P, π_1, π_2) be their pullback. The map $z \in \mathscr{C}(P) \mapsto (\pi_1 z, \pi_2 z)$ defines an order-isomorphism between $\mathscr{C}(P)$ and the following set*

$$\{(x, y) \in \mathscr{C}(A) \times \mathscr{C}(B) \mid \sigma x = \tau y \ \& \ \textit{the bijection } x \cong \sigma x = \tau y \cong y \textit{ is secured}\}$$

ordered by pairwise inclusion.

The secured hypothesis takes care of removing the causal loops by ensuring that the union of the orders stays an order. By Lemma 2, this characterizes up to isomorphism the event structures of the pullback. We can also have a grip of the causality in the pullback:

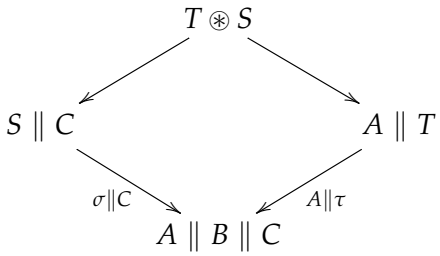
***Lemma* 9.** — *Let (P, π_1, π_2) be a pullback of $f : A \to C$ and $g : B \to C$. Then if $e \rightarrowtriangle_P e'$ then $\pi_1 e \rightarrowtriangle \pi_1 e'$ or $\pi_2 e \rightarrowtriangle \pi_2 e'$.*

This lemma states that immediate causality $\rightarrowtriangle$ in P contains the union of causalities in A or B. The converse is not true though because immediate causalities in A for instance have to be related by $\leq_P$ but the causality need not be immediate anymore.

3 A bicategory of concurrent games

3.1 Composition

With pullbacks we can define now the interaction of a pre-strategy $\sigma : S \to A^\perp \parallel B$ and $\tau : T \to B^\perp \parallel C$ as the following pullback:



To get the actual composition from this interaction, we restrict $T \circledast S$ to those events that are sent to A or C (called *visible events*):

$$T \odot S = \{p \in T \circledast S \mid (\sigma \parallel C)(\pi_1 p) \in A \parallel C\}$$

and the order and the consistent set is inherited from $T \circledast S$. We have a map of event structures $\tau \odot \sigma : T \odot S \to A^\perp \parallel C$. Even though composition requires the pullbacks whose construction is very indirect, we have seen that in the case without inconsistency how to easily compute the nteraction.

3.2 Examples of composition

...TODO... In this section, we let the B game to be

Sequential strategies. As an example of composition, let us see if copycat, our candidate behaves well with respect to composition.

Recall our negation strategy that plays on $B^\perp \parallel B$:

$$\ominus^{\mathtt{tt}} \longrightarrow \oplus^{\mathtt{tt}}$$

$$\ominus^{\mathtt{ff}} \longrightarrow \oplus^{\mathtt{ff}}$$

$$B^\perp \qquad\qquad B$$

We want to compute $neg \odot \mathbb{c}_A$ which involves only strategies whose event structures are elementary, so it is enough to make the pullback on set, and merge the causalities, which gives us:

3.3 Innocence and receptivity

In an event structure with polarities, we write for $x \subseteq^+ y$ for $x \subseteq y$ and $y \setminus x$ consists in only positive events. Likewise we define $x \subseteq^- y$ for negative extensions.

When we try to compute composition with copycat, sometimes it fails. We need to look for the class of strategies that commutes with copycat, ie. strategies $\sigma : S \to A$ such that $\mathbb{c}_A \odot \sigma \cong \sigma$. This first Lemma tells us that this statement is equivalent to composing copycat both on the input and the output of strategy:

***Lemma* 10.** — *Let $\sigma : S \to A^\perp \parallel B$ be a pre-strategy. We have the following isomorphism:*

$$\mathbb{c}_B \odot \sigma \odot \mathbb{c}_A \cong \sigma$$

***Definition* 6.** — *Let $\sigma : S \to A$ be a pre-strategy.*

- *σ is courteous when for all causalities $s \rightarrowtriangle s'$ such that either s is positive or s' negative, $\sigma s \rightarrowtriangle_A \sigma s'$*
- *σ is receptive when, for every configuration $x \in \mathscr{C}(S)$ such that $\sigma x \subseteq^- y$ with $y \in \mathscr{C}(A)$, then there exists a unique $x \subseteq x' \in \mathscr{C}(S)$ such that $\sigma x' = y$.*

A strategy will be a courteous and receptive pre-strategy.

Recall that given a monotone function f from an order $(A, \leq_A)$ to an order $(B, \leq_B)$ is a *discrete fibration* when for every $a \in A$ and $b \in B$ such that $b \leq fa$ there exists a unique $a' \leq a \in A$ such that $fa' = b$. Thus receptivity exactly means that $\sigma : (\mathscr{C}(S), \supseteq^-) \to (\mathscr{C}(A), \supseteq^-)$ is a discrete fibration. Can we express innocence in the same way?

***Lemma* 11.** — *Let $\sigma : S \to A$ be a receptive pre-strategy. Then σ is innocent if and only if $\sigma : (\mathscr{C}(S), \subseteq^+) \to (\mathscr{C}(A), \subseteq^+)$ is a discrete fibration.*

3.4 Scott order

This definition of strategies invites us to define a new order, called the *Scott order* defined on configurations of an event structure with polarity A. We define $x \sqsubseteq y$ as $x \supseteq^- \subseteq y$. This is equivalent to $x \supseteq^- x \cap y \subseteq^+ y$. The Scott order is indeed an order, and we have that:

***Lemma* 12.** — *Let $\sigma : S \to A$ be a pre-strategy. It is a strategy if and only if it is a discrete fibration wrt the Scott order on $\mathscr{C}(S)$ and $\mathscr{C}(A)$.*

Moreover, we also have:

***Lemma* 13.** — *Configurations of copycat are order-isomorphic to pairs $(x, y) \in \mathscr{C}(A) \times \mathscr{C}(A)$ with $y \sqsubseteq x$, ordered by pairwise inclusion.*

The Scott order is called that way because intuitively $x \sqsubseteq y$ means that y contains more information about the behaviour of player: he will play more events having received more information. If wee copycat as a relation between an input configuration of A and an output configuration of A, we see that copycat outputs less information that what received.

3.5 Strategies and copycat

Now we wish to prove that our condition are right in the sense that they capture exactly the pre-strategies behaving well wrt copycat which acts as a forwarder.

Thanks to Lemma 10, we only have to study the case where copycat is "applied" to a strategy on A. Let $\sigma : S \to A$ be a pre-strategy on A.

***Lemma* 14.** — *Configurations of $\mathfrak{CC}_A \circledast S$ are order-isomorphic to pairs $(x, y) \in \mathscr{C}(S) \times \mathscr{C}(A)$ such that $y \sqsubseteq \sigma x$.*

Thus interacting with copycat induces delay: what we see is y but what the strategy has performed is σx.

Configurations of the composition correspond to configurations of the interaction whose maximal events are all sent to the visible part. In our case, it means that:

***Lemma* 15.** — *Configurations of $\mathfrak{CC}_A \odot S$ are order-isomorphic to pairs $(x, y) \in \mathscr{C}(S) \times \mathscr{C}(A)$ such that $y \sqsubseteq \sigma x$, satisfying that all maximal events s of x are negative and are such that $\sigma s \in y$.*

Thus configurations of the composition correspond exactly to those configurations of the interaction where all maximal events played by S are propagated. From that lemma it is easy to conclude:

***Lemma* 16.** — *There is an isomorphism $\gamma_A \odot \sigma\sigma \cong \sigma$ if and only if σ is a strategy.*

4 Symmetry

TODO: write on symmetry

5 Concurrent Hyland-Ond games