Causality in Linear Logic

Full completeness and injectivity (unit-free multiplicative-additive fragment)

Simon Castellan and Nobuko Yoshida

Imperial College London, UK

Abstract. Commuting conversions of Linear Logic induce a notion of dependency between rules inside a proof derivation: a rule depends on a previous rule when they cannot be permuted using the conversions. We propose a new interpretation of proofs of Linear Logic as *causal invariants* which captures *exactly* this dependency. We represent causal invariants using game semantics based on general event structures, carving out, inside the model of [6], a submodel of causal invariants. This submodel supports an interpretation of unit-free Multiplicative Additive Linear Logic with MIX (MALL^{-}) which is (1) *fully complete*: every element of the model is the denotation of a proof and (2) *injective*: equality in the model characterises exactly commuting conversions of MALL^{-}. This improves over the standard fully complete game semantics model of MALL^{-}.

Keywords: Event Structures · Linear Logic · Proof Nets · Game Semantics.

1 Introduction

Proofs up to commuting conversions. In the sequent calculus of Linear Logic, the order between rules need not always matter: allowed reorderings are expressed by *commuting conversions*. These conversions are necessary for confluence of cut-elimination by mitigating the sequentiality of the sequent calculus. The real proof object is often seen as an equivalence class of proofs modulo commuting conversions. The problem of providing a canonical representation of proofs up to those commuting conversions is as old as Linear Logic itself, and proves to be a challenging problem. The traditional solution interprets a proof by a graphical representation called *proof net* and dates back to Girard [15]. Girard's solution is only satisfactory in the multiplicative-exponential fragment of Linear Logic. For additives, a well-known solution is due to Hughes and van Glabbeck [20], where proofs are reduced to their set of axiom linkings. However, the correctness criterion relies on the difficult *toggling* condition.

Proof nets tend to be based on specific representations such as graphs or sets of linkings. Denotational semantics has not managed to provide a semantic counterpart to proof nets, which would be a model where every element is the interpretation of a proof (*full completeness*) and whose equational theory coincides with commuting conversions (*injectivity*). We believe this is because denotational semantics views conversions as extensional principles, hence models proofs with extensional objects (relations, functions) too far from the syntax.

Conversions essentially state that the order between rules applied to different premises does not matter, as evidenced in the two equivalent proofs of the sequent $\vdash X^{\perp} \oplus X^{\perp}, X \oplus X$

(a)
$$\begin{tikzcd}
u : X^\perp \oplus X^\perp, & v : X \oplus X \\
u[\texttt{inl}] \arrow[d] & v[\texttt{inl}] \arrow[dd] \\
u : X^\perp \arrow[dr] & \\
& v : X
\end{tikzcd}$$

(b)
$$\begin{tikzcd}
u : X^\perp & \& \quad Y^\perp, & v : X & \oplus \quad Y \\
u(\texttt{inl}) \arrow[r, phantom, "\sim\!\sim"] \arrow[dd] \arrow[drr] & u(\texttt{inr}) \arrow[dd] \arrow[drr] & & \\
& & v[\texttt{inl}] \arrow[dd] & v[\texttt{inr}] \arrow[dd] \\
u : X^\perp \arrow[drr] & u : Y^\perp \arrow[drr] & & \\
& & v : X & v : Y
\end{tikzcd}$$

Fig. 1: Examples of causal invariants

depicted on the right. These two proofs are equal in extensional models of Linear Logic because *they have the same extensional behaviour*. Unfortunately,

$$\dfrac{\dfrac{\overline{\vdash X^\perp, X}\;Ax}{\vdash X^\perp, X \oplus X}\;\oplus_1}{\vdash X^\perp \oplus X^\perp, X \oplus X}\;\oplus_1 \qquad \dfrac{\dfrac{\overline{\vdash X^\perp, X}\;Ax}{\vdash X^\perp \oplus X^\perp, X}\;\oplus_1}{\vdash X^\perp \oplus X^\perp, X \oplus X}\;\oplus_1$$

characterising the image of the interpretation proved to be a difficult task in extensional models. The first fully complete models used game semantics, and are due to Abramsky and Melliès (MALL) [1] and Melliès (Full LL) [22]. However, their models use an *extensional* quotient on strategies to satisfy the conversions, blurring the concrete nature of strategies.

The true concurrency of conversions. Recent work [5] highlights an interpretation of Linear Logic as communicating processes. Rules become actions whose polarity (input or output) is tied to the polarity of the connective (negative or positive), and cut-elimination becomes communication. In this interpretation, each assumption in the context is assigned a channel on which the proof communicates. Interestingly, commuting conversions can be read as asynchronous permutations. For instance, the conversion mentioned above becomes the equation in the syntax of Wadler [25]:

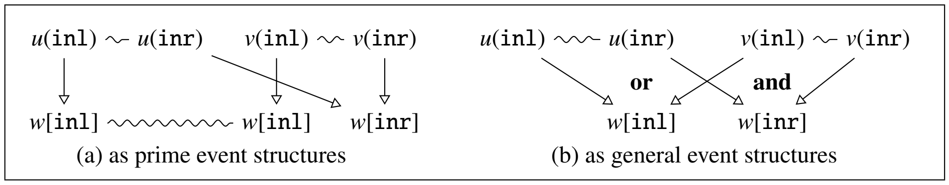
$$(1) \quad u[\texttt{inl}].\, v[\texttt{inl}].\, [u \leftrightarrow v] \equiv v[\texttt{inl}].\, u[\texttt{inl}].\, [u \leftrightarrow v] \;\triangleright\; u : X^\perp \oplus X^\perp, v : X \oplus X,$$

where $u[\texttt{inl}]$ corresponds to a $\oplus_1$-introduction rule on (the assumption corresponding to) u, and $[u \leftrightarrow v]$ is the counterpart to an axiom between the hypothesis corresponding to u and v. It becomes then natural to consider that the canonical object representing these two proofs should be a concurrent process issuing the two outputs in parallel. A notion of causality emerges from this interpretation, where a rule *depends on* a previous rule below in the tree when these two rules cannot be permuted using the commuting conversions. This leads us to causal models to make this dependency explicit. For instance, the two processes in (1) can be represented as the partial order depicted in Figure 1a, where dependency between rules is marked with $\rightarrowtriangle$.

In presence of &, a derivation stands for several execution (slices), given by different premises of a &-rule (whose process equivalent is $u.\texttt{case}\,(P, Q)$ and represents pattern matching on an incoming message). The identity on $X \oplus Y$, corresponding to the proof

$$u.\texttt{case}\,(v[\texttt{inl}].\, [u \leftrightarrow v], \;\; v[\texttt{inr}].\, [u \leftrightarrow v]) \;\triangleright\; u : X^\perp \,\&\, Y^\perp, v : X \oplus Y,$$

is interpreted by the *event structure* depicted in Figure 1b. Event structures [26] combine a partial order, representing causality, with a conflict relation representing when two events cannot belong to the same execution (here, same slice). Conflict here is indicating with $\sim\!\sim$ and separates the slices. The &-introduction becomes two conflicting events.



(a) as prime event structures (b) as general event structures

Fig. 2: Representations of or

Conjunctive and disjunctive causalities. Consider the process on the context $u : (X \oplus X)^\perp, v : (Y \oplus Y)^\perp, w : (X \otimes Y) \oplus (X \otimes Y)$ implementing disjunction:

$$\texttt{or} = u.\texttt{case}\begin{pmatrix} v.\texttt{case}\,(w[\texttt{inl}].\,P, w[\texttt{inl}].\,P), \\ v.\texttt{case}\,(w[\texttt{inl}].\,P, w[\texttt{inr}].\,P) \end{pmatrix} \text{ where } \; P = w[x].\,([u \leftrightarrow w] \mid [v \leftrightarrow x]).$$

Cuts of or against a proof starting with $u[\texttt{inl}]$ or $v[\texttt{inl}]$ answer on w after reduction:

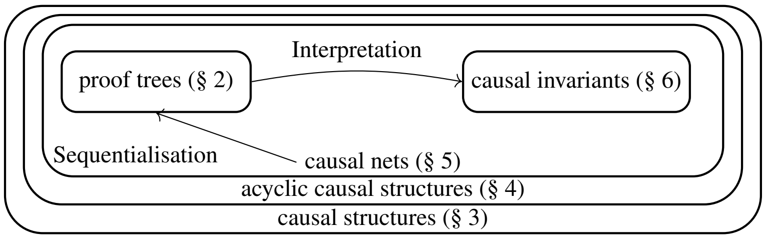
$$(\nu u)(\texttt{or} \mid u[\texttt{inl}]) \to^* w[\texttt{inl}].v.\texttt{case}\,(P, P) \quad (\nu v)(\texttt{or} \mid v[\texttt{inl}]) \to^* w[\texttt{inl}].u.\texttt{case}\,(P, P)$$

where $(\nu u)(P \mid Q)$ is the process counterpart to logical cuts. This operational behaviour is related to *parallel or*, evaluating its arguments in parallel and returning true as soon as one returns true. Due to this intentional behaviour, the interpretation of or in prime event structures is nondeterministic (Figure 2a), as causality in event structures is *conjunctive* (an event may only occur after all its predecessors have occurred). By moving to *general* event structures, however, we can make the disjunctive causality explicit and recover determinism (Figure 2b).

Contributions and outline. Drawing inspiration from the interpretation of proofs in terms of processes, we build a fully complete and injective model of unit-free Multiplicative Additive Linear Logic with MIX (MALL$^-$), interpreting proofs as general event structures living in a submodel of the model introduced by [6]. Moreover, our model captures the dependency between rules, which makes sequentialisation a local operation, unlike in proof nets, and has a more uniform acyclicity condition than [20].

We first recall the syntax of MALL$^-$ and its reading in terms of processes in § 2. Then, in § 3, we present a slight variation on the model of [6], where we call the (pre)strategies *causal structures*, by analogy with proof structures. Each proof tree can be seen as a (sequential) causal structure. However, the space of causal structures is too broad and there are many causal structures which do not correspond to any proofs. A major obstacle to sequentialisation is the presence of *deadlocks*. In § 4, we introduce a condition on causal structures, ensuring deadlock-free composition, inspired by the interaction between $\parr$ and $\otimes$ in Linear Logic. Acyclic causal structures are still allowed to only explore partially the game, contrary to proofs which must explore it exhaustively, hence in § 5, we introduce further conditions on causal structures, ensuring a *strong* sequentialisation theorem (Theorem 2): we call them *causal nets*. In § 6, we define causal invariants as maximal causal nets. Every causal net embeds in a *unique* causal invariant; and a particular proof P embeds inside a unique causal invariant which forms its denotation $[\![P]\!]$. Moreover, two proofs embed in the same causal invariant if and only if they are convertible (Theorem 4). Finally, we show how to equip causal invariants with the structure of $*$-autonomous category with products and deduce that they form a fully complete model of MALL$^-$ (Theorem 6) for which the interpretation is injective.

The appendix provides proofs of the statements in the main sections.



2 MALL$^-$ and its commuting conversions

In this section, we introduce MALL$^-$ formulas and proofs as well as the standard commuting conversions and cut elimination for this logic. As mentionned in the introduction, we use a process-like presentation of proofs following [25]. This highlights the communicating aspect of proofs which is an essential intuition for the model; and it offers a concise visualisation of proofs and conversions.

Formulas. We define the formulas of MALL$^-$: $T, S ::= X \mid X^\perp \mid T \otimes S \mid T \parr S \mid T \oplus S \mid T \& S$, where X and $X^\perp$ are *atomic formulas* (or *ltterals*) belonging to a set $\mathbb{A}$. Formulas come with the standard notion of duality $(\cdot)^\perp$ given by the De Morgan rules: $\otimes$ is dual to $\parr$, and $\oplus$ to $\&$. An *environment* is a partial mapping of *names* to formulas, instead of a multiset of formulas – names disambiguate which assumption a rule acts on.

Proofs as processes. We see proofs of MALL$^-$ (with MIX) as typing derivations for a variant of the π-calculus [25]. The (untyped) syntax for the processes is as follows:

$$\begin{array}{llr} P, Q ::= & u(v).\, P \mid u[v].\, (P \mid Q) & \text{(multiplicatives)} \\ \mid & u.\mathtt{case}\,(P, Q) \mid u[\mathtt{inl}].\, P \mid u[\mathtt{inr}].\, P & \text{(additives)} \\ \mid & [u \leftrightarrow v] \mid (\nu u)(P \mid Q) \mid \ (P \mid Q) & \text{(logical and mix)} \end{array}$$

$u(v).P$ denotes an input of v on channel u (used in $\parr$-introduction) while $u[v].(P \mid Q)$ denotes output of a fresh channel v along channel u (used in $\otimes$-introduction); The term $[u \leftrightarrow v]$ is a *link*, forwarding messages received on u to v, corresponds to axioms, and conversely; and $(\nu u)(P \mid Q)$ represents a restriction of u in P and Q and corresponds to cuts; $u.\mathtt{case}\,(P, Q)$ is an input branching representing $\&$-introductions, which interacts with selection, either $u[\mathtt{inl}].\, R$ or $u[\mathtt{inr}].\, R$; in $(\nu u)(P \mid Q)$, u is bound in both P and Q, in $u(v).\, P$, v is bound in P, and in $u[v].\, (P \mid Q)$, v is only bound in Q.

We now define MALL$^-$ proofs as typing derivations for processes. The inference rules, recalled in Figure 3, are from [25]. The links (axioms) are restricted to literals – for composite types, one can use the usual η-expansion laws. There is a straightforward bijection between standard (η-expanded) proofs of MALL$^-$ and typing derivations.

$$\frac{P \rhd u : T, v : S, \Gamma}{u(v).\, P \rhd u : T \parr S, \Gamma} \qquad \frac{P \rhd u : T, \Gamma \quad Q \rhd v : S, \Delta}{u[v].\, (P|Q) \rhd u : T \otimes S, \Gamma, \Delta} \qquad \frac{}{[u \leftrightarrow v] \rhd u : X^{\perp}, v : X}$$

$$\frac{P \rhd \Gamma, u : T \quad Q \rhd \Delta, u : T^{\perp}}{(\nu u)(P \mid Q) \rhd \Gamma, \Delta} \qquad \frac{P \rhd \Gamma, u : T \qquad Q \rhd \Gamma, u : S}{u.\mathtt{case}\,(P, Q) \rhd \Gamma, u : T \& S} \qquad \frac{P \rhd \Gamma, u : T}{u[\mathtt{inl}].\, P \rhd \Gamma, u : T \oplus S}$$

$$\frac{P \rhd \Gamma, u : S}{u[\mathtt{inr}].\, P \rhd \Gamma, u : T \oplus S} \qquad \frac{P \rhd \Gamma \qquad Q \rhd \Delta}{P \mid Q \rhd \Gamma, \Delta}$$

$$\frac{}{\vdash u[\mathtt{inl}].\, [] : \Gamma, u : T \Rightarrow \Gamma, u : T \oplus S} \qquad \frac{Q \rhd \Delta, v : S}{\vdash u[v].\, ([] \mid Q) : \Gamma, u : T \Rightarrow u : T \otimes S, \Gamma, \Delta}$$

$$\frac{}{\vdash u[\mathtt{inr}].\, [] : \Gamma, u : S \Rightarrow \Gamma, u : T \oplus S} \qquad \frac{P \rhd \Gamma, u : T}{\vdash u[v].\, (P \mid []) : \Delta, v : S \Rightarrow u : T \otimes S, \Gamma, \Delta}$$

$$\frac{}{\vdash u.\mathtt{case}\,([]_1, []_2) : (\Gamma, u : T) \times (\Gamma, u : S) \Rightarrow \Gamma, u : T \,\&\, S}$$

$$\frac{}{\vdash u(v).\, [] : \Gamma, u : T, v : S \Rightarrow \Gamma, u : T \parr S} \qquad \frac{P \rhd \Delta}{\vdash ([] \mid P) : \Gamma \Rightarrow \Gamma, \Delta} \qquad \frac{P \rhd \Gamma}{\vdash (P \mid []) : \Delta \Rightarrow \Gamma, \Delta}$$

Fig. 3: Typing rules for MALL^{-} (above) and contexts (below)

Commutation rules and cut elimination We now explain the valid commutations rules in our calculus. We consider contexts $C\,[[]_1, \ldots, []_n]$ with several holes to accomodate & which has two branches. Contexts are defined in Figure 3, and are assigned a type $\Gamma_1 \times \ldots \times \Gamma_n \Rightarrow \Delta$. It intuitively means that if we plug proofs of Γ_i in the holes, we get back a proof of Δ. We use the notation $C[P_i]_i$ for $C[P_1, \ldots, P_n]$ when (P_i) is a family of processes. Commuting conversion is the smallest congruence $\equiv$ satisfying all well-typed instances of the rule $C[D[P_{i,j}]_j]_i \equiv D[C[P_{i,j}]_i]_j$ for C and D two contexts. For instance $a[\mathtt{inl}].\, b.\mathtt{case}\,(P, Q) \equiv b.\mathtt{case}\,(a[\mathtt{inl}].\, P, a[\mathtt{inl}].\, Q)$. Figure 4 gives reduction rules $P \to Q$. The first four rules are the *principal* cut rules and describe the interaction of two dual terms, while the last one allows cuts to move inside contexts.

3 Concurrent games based on general event structures

This section introduces a slight variation on the model of [6]. In § 3.1, we define *games* as prime event structures with polarities, which are used to interpret formulas. We then introduce general event structures in § 3.2, which are used to define causal structures.

$$(\nu u)([u \leftrightarrow v] \mid P) \to P[v/u] \qquad (\nu u)(u[x].\, (P \mid Q) \mid u(x).\, R) \to (\nu u)(P \mid (\nu x)(Q \mid R))$$

$$(\nu u)(u[\mathtt{inl}].\, R \mid u.\mathtt{case}\,(P, Q)) \to (\nu u)(R \mid P) \qquad (\nu u)(u[\mathtt{inr}].\, R \mid u.\mathtt{case}\,(P, Q)) \to (\nu u)(R \mid Q)$$

$$(\nu u)(C[P_i]_i \mid Q) \to C[(\nu u)(P_i \mid Q)]_i \quad (u \notin C)$$

Fig. 4: Cut elimination in MALL^{-}

3.1 Games as prime event structures with polarities

Definition of games. Prime event structures [26] (simply event structures in the rest of the paper) are a causal model of nondeterministic and concurrent computation. We use here prime event structures *with binary conflict*. An **event structure** is a triple $(E, \leq_E, \#_E)$ where $(E, \leq_E)$ is a partial order and $\#_E$ is an irreflexive symmetric relation (representing **conflict**) satisfying: (1) if $e \in E$, then $[e] := \{e' \in E \mid e' \leq_E e\}$ is finite; and (2) if $e \#_E e'$ and $e \leq_E e''$ then $e'' \#_E e'$. We often omit the E subscripts when clear from the context.

A **configuration** of E is a downclosed subset of E which does not contain two conflicting events. We write $\mathscr{C}(E)$ for the set of *finite* configurations of E. For any $e \in E$, $[e]$ is a configuration, and so is $[e) := [e] \setminus \{e\}$. We write $e \rightarrowtriangle e'$ for the immediate causal relation of E defined as $e < e'$ with no event between. Similarly, a conflict $e\#e'$ is **minimal**, denoted $e \sim\!\sim e'$, when the $[e] \cup [e')$ and $[e) \cup [e']$ are configurations. When drawing event structures, only $\rightarrowtriangle$ and $\sim\!\sim$ are represented. We write $\max(E)$ for the set of maximal events of E for $\leq_E$. An event e is maximal in x when it has no successor for $\leq_E$ in x. We write $\max_E x$ for the maximal events of a configuration $x \in \mathscr{C}(E)$.

An event structure E is **confusion-free** when (1) for all $e \sim\!\sim_E e'$ then $[e) = [e')$ and (2) if $e \sim\!\sim_E e'$ and $e' \sim\!\sim_E e''$ then $e = e''$ or $e \sim\!\sim_E e''$. As a result, the relation "$e \sim\!\sim e'$ or $e = e'$" is an equivalence relation whose equivalent classes $\mathfrak{a}$ are called **cells**.

Definition 1. *A **game** is a confusion-free event structure A along with an assignment $pol : A \to \{-, +\}$ such that cells contain events of the same polarity, and a function atom: $\max(A) \to \mathbb{A}$ mapping every maximal event of A to an atom. Events with polarity $-$ (resp. $+$) are **negative** (resp. **positive**).*

Events of a game are usually called *moves*. The restriction imposes branching to be polarised (*i.e.* belonging to a player). A game is **rooted** when two minimal events are in conflict. Single types are interpreted by rooted games, while contexts are interpreted by arbitrary games. When introducing moves of a game, we will indicate their polarity in exponent, *e.g.* "let $a^+ \in A$" stands for assuming a positive move of A.

Interpretation of formulas. To interpret formulas, we make use of standard constructions on prime event structures. The event structure $a \cdot E$ is E prefixed with a, *i.e.* $E \cup \{a\}$ where *all* events of E depends on a. The parallel composition of E and E' represents parallel executions of E and E' without interference:

Definition 2. *The parallel composition of event structures A_0 and A_1 is the event structure $A_0 \parallel A_1 = (\{0\} \times A_0 \cup \{1\} \times A_1, \leq_{A_0\parallel A_1}, \#_{A_0\parallel A_1})$ with $(i, a) \leq_{A_0\parallel A_1} (j, a')$ iff $i = j$ and $a \leq_{A_i} a'$; and $(i, a) \#_{A_0\parallel A_1} (j, a')$ when $i = j$ and $a \#_{A_j} a'$.*

The sum of event structure $E + F$ is the nondeterministic analogue of parallel composition.

Definition 3. *The sum $A_0 + A_1$ of the two event structures A_0 and A_1 has the same partial order as $A_0 \parallel A_1$, and conflict relation $(i, a) \#_{A_0+A_1} (j, a')$ iff $i \neq j$ or $i = j$ and $a \#_{A_j} a'$.*

Prefixing, parallel composition and sum of event structures extend to games. The dual of a game A, obtained by reversing the polarity labelling, is written $A^\perp$. Given $x \in \mathscr{C}(A)$, we define A/x ("A after x") as the subgame of A comprising the events $a \in A \setminus x$ not in conflict with events in x.

Interpretation of formulas. The interpretation of the atom X is the game with a single positive event simply written X with $atom(X) = X$, and the interpretation of $X^\perp$ is $[\![X]\!]^\perp$, written simply $X^\perp$ in diagrams. For composite formulas, we let (where `send`, `inl` and `inr` are simply labels):

$$[\![S \otimes T]\!] = \mathtt{send}^+ \cdot ([\![S]\!] \parallel [\![T]\!]) \qquad [\![S \parr T]\!] = \mathtt{send}^- \cdot ([\![S]\!] \parallel [\![T]\!])$$
$$[\![S \oplus T]\!] = (\mathtt{inl}^+ \cdot [\![S]\!]) + (\mathtt{inr}^+ \cdot [\![T]\!]) \qquad [\![S \mathbin{\&} T]\!] = (\mathtt{inl}^- \cdot [\![S]\!]) + (\mathtt{inr}^- \cdot [\![T]\!])$$

Parallel composition is used to interpret contexts: $[\![u_1 : T_1, \ldots, u_n : T_n]\!] = [\![T_1]\!] \parallel \ldots \parallel [\![T_n]\!]$. The interpretation commutes with duality: $[\![T]\!]^\perp = [\![T^\perp]\!]$.

In diagrams, we write moves of a context following the syntactic convention: for instance $u[\mathtt{inl}]$ denotes the minimal `inl` move of the u component. For tensors and pars, we use the notation $u[v]$ and $u(v)$ to make explicit the variables we use in the rest of the diagram, instead of $\mathtt{send}^+$ and $\mathtt{send}^-$ respectively. For atoms, we use $u : X$ and $u : X^\perp$.

3.2 Causal structures as deterministic general event structures

As we discussed in § 1, prime event structures cannot express disjunctive causalities deterministically, hence fail to account for the determinism of LL. Our notion of causal structure is based on *general event structures*, which allow more complex causal patterns. We use a slight variation on the definition of deterministic general event structures given by [6], to ensure that composition is well-defined without further assumptions.

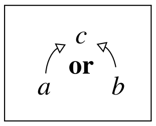
Instead of using the more concrete representation of general event structures in terms of a set of events and an enabling relation, we use the following formulation in terms of set of configurations, more adequate for mathematical reasoning. Being only sets of configurations, they can be reasoned on with very simple set-theoretic arguments.

Definition 4. *A* ***causal structure*** *(abbreviated as causal struct) on a game A is a subset $\sigma \subseteq \mathscr{C}(A)$ containing $\emptyset$ and satisfying the following conditions:*

Coincidence-freeness *If $e, e' \in x \in \sigma$ then there exists $y \in \sigma$ with $y \subseteq x$ and $y \cap \{e, e'\}$ is a singleton.*

Determinism *for $x, y \in \sigma$ such that $x \cup y$ does not contain any minimal negative conflict, then $x \cup y \in \sigma$.*

Configurations of prime event structures satisfy a further axiom, *stability*, which ensures the absence of disjunctive causalities. When σ is a causal struct on A, we write $\sigma : A$. We draw as regular event structures, using $\rightarrowtriangle$ and $\rightsquigarrow$. To indicate disjunctive causalities, we annotate joins with **or**. This convention is not powerful enough to draw *all* causal structs, but enough for the examples in this paper. As an example, on $A = a \parallel b \parallel c$ the diagram on the right denotes the following causal struct $\sigma = \{x \in \mathscr{C}(A) \mid c \in x \Rightarrow x \cap \{a, b\} \neq \emptyset\}$.



A **minimal event** of $\sigma : A$ is an event $a \in A$ with $\{a\} \in \sigma$. An event $a \in x \in \sigma$ is **maximal** in x when $x \setminus \{a\} \in \sigma$. A **prime configuration** of $a \in A$ is a configuration $x \in \sigma$ such that a is its unique maximal event. Because of disjunctive causalities, an event $a \in A$ can have several distinct prime configurations in σ (unlike in event structures). In the previous example, since c can be caused by either a or b, it has two prime configurations: $\{a, c\}$ and $\{b, c\}$. We write $\max \sigma$ for the set of **maximal configurations** of σ, ie. those configurations that cannot be further extended.

Even though causality is less clear in general event structures than in prime event structures, we give here a notion of immediate causal dependence that will be central to define acyclic causal structs. Given a causal struct $\sigma : A$ and $x \in \sigma$, we define a relation $\rightarrowtriangle_{x,\sigma}$ on x as follows: $a \rightarrowtriangle_{x,\sigma} a'$ when there exists a prime configuration y of a' such that $x \cup y \in \sigma$, and that a is maximal in $y \setminus \{a'\}$. This notion is compatible with the drawing above: we have $a \rightarrowtriangle_{\emptyset} c$ and $b \rightarrowtriangle_{\emptyset} c$ as c has two prime configurations: $\{a, c\}$ and $\{b, c\}$. Causality needs to be contextual, since different slices can implement different causal patterns. Parallel composition and prefixing structures extend to causal structs:

$$\sigma \parallel \tau = \{x \parallel y \in \mathscr{C}(A \parallel B) \mid (x, y) \in \sigma \times \tau\} \qquad a \cdot \sigma \quad = \{x \in \mathscr{C}(a \cdot A) \mid x \cap A \in \sigma\}.$$

Categorical setting. Causal structs can be composed using the definitions of [6]. Consider $\sigma : A^\perp \parallel B$ and $\tau : B^\perp \parallel C$. A **synchronised configuration** is a configuration $x \in \mathscr{C}(A \parallel B \parallel C)$ such that $x \cap (A \parallel B) \in \sigma$ and $x \cap (B \parallel C) \in \tau$. A synchronised configuration x is **reachable** when there exists a sequence (**covering chain**) of synchronised configurations $x_0 = \emptyset \subseteq x_1 \subseteq \ldots \subseteq x_n = x$ such that $x_{i+1} \setminus x_i$ is a singleton. The reachable configurations are used to define the interaction $\tau \circledast \sigma$, and then after hiding, the composition $\tau \odot \sigma$:

$$\tau \circledast \sigma = \{x \text{ is a reachable synchronised configuration}\} \qquad \tau \odot \sigma = \{x \cap (A \parallel C) \mid x \in \tau \circledast \sigma\}.$$

Unlike in [6], our determinism is strong enough for $\tau \odot \sigma$ to be a causal struct.

Lemma 1. *If $\sigma : A^\perp \parallel B$ and $\tau : B^\perp \parallel C$ are causal structs then $\tau \odot \sigma$ is a causal struct.*

Composition of causal structs will be used to interpret cuts between proofs of Linear Logic. In concurrent game semantics, composition has a natural identity, asynchronous copycat [23], playing on the game $A^\perp \parallel A$, forwarding negative moves on one side to the positive occurrence on the other side. Following [6], we define $\text{æ}_A = \{x \parallel y \in \mathscr{C}(A^\perp \parallel A) \mid y \supseteq^-_A x \cap y \subseteq^+_A x\}$ where $x \subseteq^p y$ means $x \subseteq y$ and $pol(y \setminus x) \subseteq \{p\}$.

However, in general copycat is not an identity on all causal structs, only $\sigma \subseteq \text{æ}_A \odot \sigma$ holds. Indeed, copycat represents an asynchronous buffer, and causal structs which expects messages to be transmitted synchronously may be affected by composition with copycat. We call causal structs that satisfy the equality **asynchronous**. From [6], we know that asynchronous causal structs form a compact-closed category.

The syntactic tree. The syntactic tree of a derivation $P \triangleright \Delta$ can be read as a causal struct $Tr(P)$ on $[\![\Delta]\!]$, which will be the basis for our interpretation. It is defined by induction:

$$Tr(u(v).\,P) = u(v) \cdot Tr(P) \qquad Tr(u[v].\,(P \mid Q)) = u[v] \cdot (Tr(P) \parallel Tr(Q))$$
$$Tr(a.\mathtt{case}\,(P, Q)) = (a(\mathtt{inl}) \cdot Tr(P)) \cup (a(\mathtt{inr}) \cdot Tr(Q))$$
$$Tr(a[\mathtt{inl}].\,P) = a[\mathtt{inl}] \cdot Tr(P) \qquad Tr(a[\mathtt{inr}].\,P) = a[\mathtt{inr}] \cdot Tr(P)$$
$$Tr([a \leftrightarrow b]) = \text{æ}_{[\![X]\!]} \text{ where } \Delta = a : X^\perp, b : X \qquad Tr(P \mid Q) = Tr(P) \parallel Tr(Q)$$
$$Tr((\nu a)(P \mid Q)) = Tr(P) \odot Tr(Q)$$

We use the convention in the diagram, for instance $u[v]$ means the initial `send` move of the u component. An example of this construction is given in Figure 5a. Note that it is not asynchronous.

4 Acyclicity of causal structures

The space of causal structs is unfortunately too broad to provide a notion of causal nets, due in particular to the presence of deadlocks during composition. As a first step towards defining causal nets, we introduce in this section a condition on causal structs inspired by the tensor rule in Linear Logic. In § 4.1, we propose a notion of communication between actions, based on causality. In § 4.2, we introduce a notion of acyclicity which is shown to be stable under composition and ensure deadlock-free composition.

4.1 Communication in causal structures

The tensor rule of Linear Logic says that after a tensor $u[v]$, the proof splits into two independent subproofs, one handling u and the other v. This syntactic condition is there to ensure that there are no communications between u and v. More precisely, we want to prevent any dependence between subsequent actions on u and an action v. Indeed such a causal dependence could create a deadlock when facing a par rule $u(v)$, which is allowed to put arbitrary dependence between such subsequent actions.

Communication in MLL. Let us start by the case of MLL, which corresponds to the case where games do not have conflicts. Consider the following three causal structs:

$$\begin{tikzcd}
u : X^\perp \arrow[rrrr, bend left] & & w : Y^\perp \arrow[rrrr, bend left] & & v : X & & z : Y \\
& u[w] \arrow[ul] \arrow[ur] & & & & v(z) \arrow[ul] \arrow[ur] \arrow[llll] &
\end{tikzcd}$$

σ_1

$$\begin{tikzcd}
u : X^\perp \arrow[rrrr, bend left] & & w : Y^\perp \arrow[rrrr, bend left] & & v : X & & z : Y \\
& u[w] \arrow[ul] \arrow[ur] & & & & v(z) \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

σ_2

$$\begin{tikzcd}
u : X^\perp \arrow[rrrr, bend left] & & w : Y^\perp \arrow[rrrr, bend left] & & v : X & & z : Y \\
& u[w] \arrow[ul] \arrow[ur] & & & & v[z] \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

σ_3

The causal structs σ_1 and σ_2 play on the game $[\![u : X^\perp \otimes Y^\perp, v : X \parr Y]\!]$, while σ_3 plays on the game $[\![u : X^\perp \otimes Y^\perp, v : X \otimes Y]\!]$. The causal structs σ_2 and σ_3 are very close to proof nets, and it is easy to see that σ_2 represents a correct proof net while σ_3 does not. In particular, there exists a proof P such that $Tr(P) \subseteq \sigma_2$ but there are no such proof Q for σ_3. Clearly, σ_3 should not be acyclic. But should σ_2? After all it is sequentialisable. But, in all sequentialisations of σ_2, the par rule $v(z)$ is applied *before* the tensor $u[w]$, and this dependency is not reflected by σ_2. Since our goal is exactly to compute these implicit dependencies, we will only consider σ_1 to be acyclic, by using a stronger sequentialisation criterion:

Definition 5. *A causal struct $\sigma : [\![\Gamma]\!]$ is **strongly sequentialisable** when for all $x \in \sigma$, there exists $P \triangleright \Gamma$ with $x \in Tr(P)$ and $Tr(P) \subseteq \sigma$.*

To understand the difference between σ_1 and σ_2, we need to look at causal chains. In both σ_1 and σ_2, we can go from $u : X^\perp$ to $w : Y^\perp$ by following immediate causal links $\rightarrowtriangle$ in any direction, but observe that in σ_1 they must all cross an event below $u[w]$ (namely $v(z)$ or $u[w]$). This prompts us to define a notion of communication *outside a configuration x*:

Definition 6. *Given $\sigma : A$ and $x \in \sigma$ we say that $a, a' \in A \setminus x$ communicate* outside x *(written $a \leftrightsquigarrow_{x,\sigma} a'$) when there exists a chain $a \leftrightarrow_{x,\sigma} a_0 \leftrightarrow_{\sigma} \cdots \leftrightarrow_{x,\sigma} a_n \leftrightarrow_{\sigma} a'$ where all the $a_i \in A \setminus x$, and $\leftrightarrow_{x,\sigma}$ denotes the symmetric closure of $\rightarrowtriangle_{x,\sigma}$.*

Communication in MALL. In presence of additives, immediate causality is not the only vector of communication. Consider the following causal struct σ_4, playing on the context $u : (A \,\&\, A) \otimes (A \,\&\, A), v : (A \oplus A) \,\&\, (A \oplus A)$ where A is irrelevant:

$$\begin{tikzcd}
u(\mathtt{inl}) \arrow[r, phantom, "\sim"] \arrow[rrrrrrr, bend left] & u(\mathtt{inr}) \arrow[rrrrr, bend left] & w(\mathtt{inr}) \arrow[r, phantom, "\sim"] \arrow[rrr, bend left] & w(\mathtt{inr}) \arrow[r] & v[\mathtt{inl}] & v[\mathtt{inr}] & v[\mathtt{inl}] & v[\mathtt{inr}] \\
& u[w] \arrow[ul] \arrow[u] \arrow[ur] \arrow[urr] & & & & v(\mathtt{inl}) \arrow[r, phantom, "\sim"] \arrow[ul] \arrow[u] & v(\mathtt{inr}) \arrow[u] \arrow[ur] &
\end{tikzcd}$$

This pattern is not strongly sequentialisable: the tensor $u[w]$ must always go after the &-introduction on v, since we need this information to know how whether v should go with u or w when splitting the context. Yet, it is not possible to find a communication path from one side to the other by following purely causal links without crossing $u[w]$. There is however a path that uses both immediate causality and *minimal conflict*. This means that we should identify events in minimal conflict, since they represent the same (&-introduction rule). Concretely, this means lifting the previous definition at the level of cells. Given an causal struct $\sigma : A$ and $x \in \sigma$, along with two cells $\mathfrak{a}, \mathfrak{a}'$ of A/x, we define the relation $\mathfrak{a} \leftrightarrow_{x,\sigma} \mathfrak{a}'$ when there exists $a \in \mathfrak{a}$ and $a' \in \mathfrak{a}'$ such that $a \leftrightarrow_{x,\sigma} a'$; and $\mathfrak{a} \leftrightsquigarrow_{x,\sigma} \mathfrak{a}'$ when there exists $\mathfrak{a} \leftrightarrow_{x,\sigma} \mathfrak{a}_0 \leftrightarrow_{x,\sigma} \cdots \leftrightarrow_{\sigma} \mathfrak{a}_n \leftrightarrow_{\sigma} \mathfrak{a}'$ where all the $\mathfrak{a}_i$ do not intersect x. For instance, the two cells which are successors of the tensor $u[w]$ in σ_4 communicate outside the configuration $\{u[w]\}$ by going through the cell $\{v(\mathtt{inl}), v(\mathtt{inr})\}$.

4.2 Definition of acyclicity on causal structures

Since games are trees, two events a, a' are either incomparable or have a meet $a \wedge a'$. If $a \wedge a'$ is defined and positive, we say that a and a' **have positive meet**, and means that that they are on two distinct branches of a tensor. If $a \wedge a'$ is undefined, or defined and negative, we say that $a \wedge a'$ has a **negative meet**. When the meet is undefined, it means that a and a' are events of different components of the context. We consider the meet to be negative in this case, since components of a context are related by an implicit par.

These definitions are easily extended to cells. The meet $\mathfrak{a} \wedge \mathfrak{a}'$ of two cells $\mathfrak{a}$ and $\mathfrak{a}'$ of A is the meet $a \wedge a'$ for $a \in \mathfrak{a}$ and $a' \in \mathfrak{a}'$: by confusion-freeness, it does not matter which ones are chosen. Similarly, we say that $\mathfrak{a}$ and $\mathfrak{a}'$ have positive meet if $\mathfrak{a} \wedge \mathfrak{a}'$ is defined and positive; and have negative meet otherwise. These definitions formalise the idea of "the two sides of a tensor", and allow us to define acyclicity.

Definition 7. *A causal struct $\sigma : A$ is* ***acyclic*** *when for all $x \in \sigma$, for any cells $\mathfrak{a}, \mathfrak{a}'$ not intersecting x and with positive meet, if $\mathfrak{a} \leftrightsquigarrow_{x,\sigma} \mathfrak{a}'$ then $\mathfrak{a} \wedge \mathfrak{a}' \notin x$.*

This captures the desired intuition: if $\mathfrak{a}$ and $\mathfrak{a}'$ are on two sides of a tensor a (ie. have positive meet), and there is a communication path outside x relating them, then a must also be outside x (and implicitly, the communication path must be going through a).

Reasoning on the interaction of acyclic strategies proved to be challenging. We prove that acyclic strategies compose, and their interaction are deadlock-free, when composition is on a rooted game B. This crucial assumption arises from the fact that in linear logic, cuts are on *formulas*. It entails that for any $b, b' \in B$, $b \wedge b'$ is defined, hence must be positive either from the point of view of σ or of τ.

Theorem 1. *For acyclic causal structs $\sigma : A^\perp \parallel B$ and $\tau : B^\perp \parallel C$, (1) their interaction is* deadlock-free*: $\tau \circledast \sigma = (\sigma \parallel C) \cap (A \parallel \tau)$; and (2) the causal struct $\tau \odot \sigma$ is acyclic.*

As a result, acyclic and asynchronous causal structs form a category. We believe this intermediate category is interesting in its own right since it generalises the deadlock-freeness argument of Linear Logic without having to assume other constraints coming from Linear Logic, such as linearity. In the next section, we study further restriction on acyclic causal structs which guarantee strong sequentialisability.

5 Causal nets and sequentialisation

We now ready to introduce causal nets. In § 5.1, we give their definition by restricting acyclic causal structs and in § 5.2 we prove that causal nets are strongly sequentialisable.

5.1 Causal nets: totality and well-linking causal structs

To ensure that our causal structs are strongly sequentialisable, acyclicity is not enough. First, we need to require causal structs to respect the linearity discipline of Linear Logic:

Definition 8. *A causal struct $\sigma : A$ is **total** when* (1) *for $x \in \sigma$, if x is maximal in σ, then it is maximal in $\mathscr{C}(A)$; and* (2) *for $x \in \sigma$ and $a^- \in A \setminus x$ such that $x \cup \{a\} \in \sigma$, then whenever $a \sim_A a'$, we also have $x \cup \{a'\} \in \sigma$ as well.*

The first condition forces a causal struct to play until there are no moves to play, and the second forces an causal struct to be receptive to all Opponent choices, not a subset.

Our last condition constrains axiom links. A **linking** of a game A is a pair (x, ℓ) of a $x \in \max \mathscr{C}(A)$, and a bijection $\ell : (\max_A x)^- \simeq (\max_A x)^+$ preserving the *atom* labelling.

Definition 9. *A total causal struct $\sigma : A$ is **well-linking** when for each $x \in \max(\sigma)$, there exists a linking ℓ_x of x, such that if y is a prime configuration of $\ell_x(e)$ in x, then* $\max(y \setminus \{\ell_x(e)\}) = \{e\}$.

This ensures that every positive atom has a unique predecessor which is a negative atom.

Definition 10. *A **causal net** is an acyclic, total and well-linking causal struct.*

A causal net $\sigma : A$ induces a set of linkings A, $\mathsf{link}(\sigma) := \{\ell_x \mid x \in \max \sigma\}$. The mapping $\mathsf{link}(\cdot)$ maps causal nets to the proof nets of [20].

5.2 Strong sequentialisation of causal nets

Our proof of sequentialisation relies on an induction on causal nets. To this end, we provide an inductive deconstruction of parallel proofs. Consider $\sigma : A$ a causal net and a minimal event $a \in \sigma$ not an atom. We write A/a for $A/\{a\}$. Observe that if $A = [\![\Delta]\!]$, it is easy to see that there exists a context Δ/a such that $[\![\Delta/a]\!] \cong A/a$. Given a causal struct $\sigma : A$, we define the causal struct $\sigma/a = \{x \in \mathscr{C}(A/a) \mid x \cup \{a\} \in \sigma\} : A/a$.

Lemma 2. *σ/a is a causal net on A/a.*

When a is positive, we can further decompose σ/a in disjoint parts thanks to acyclicity. Write $\mathfrak{a}_1, \dots, \mathfrak{a}_n$ for the minimal cells of A/a and consider for $n \geq k > 0$, $A_k = \{a' \in A/a \mid \mathrm{cell}(a') \leftrightsquigarrow_{\{a\},\sigma} \mathfrak{a}_k\}$. A_k contains the events of A/a which σ connects to the k-th successor of a. We also define the set $A_0 = A/a \setminus \bigcup_{1 \leq k \leq n} A_k$, of events not connected to any successor of a (this can happen with MIX). It inherits a game structure from A.

Each subset inherits a game structure from A/a. By acyclicity of σ, the A_k are pairwise disjoint, so $A/a \cong A_0 \parallel \dots \parallel A_n$. For $0 \leq k \leq n$, define $\sigma_k = \mathscr{C}(A_k) \cap \sigma/a$.

Lemma 3. *σ_k is a causal net on A_k and we have $\sigma/a = \sigma_0 \parallel \dots \parallel \sigma_n$.*

This formalises the intuition that after a tensor, an acyclic causal net must be a parallel composition of proofs (following the syntactic shape of the tensor rule of Linear Logic). From this result, we show by induction that any causal net is strongly sequentialisable.

Theorem 2. *If $\sigma : A$ is a causal net, then σ is strongly sequentialisable.*

We believe sequentialisation without MIX requires causal nets to be *connected*: two cells with negative meets always communicate outside any configuration they are absent from. We leave this lead for future work.

6 Causal invariants and completeness

Causal nets are naturally ordered by inclusion. When $\sigma \subseteq \tau$, we can regard τ as a less sequential implementation of σ. Two causal nets which are upper bounded by a causal net should represent the same proof, but with varying degrees of sequentiality. Causal nets which are maximal for inclusion (among causal nets) are hence most parallel implementations of a certain behaviour and capture our intuition of causal invariants.

Definition 11. *A **causal invariant** is a causal net $\sigma : A$ maximal for inclusion.*

6.1 Causal invariants as maximal causal nets

We start by characterising when two causal nets are upper-bounded for inclusion:

Proposition 1. *Given two causal nets $\sigma, \tau : A$, the following are equivalent:*

1. *there exists a causal net $\upsilon : A$ such that $\sigma \subseteq \upsilon$ and $\tau \subseteq \upsilon$,*
2. *the set $\sigma \vee \tau = \{x \cup y \mid x \in \sigma, y \in \tau, x \cup y \in \mathscr{C}(A)\}$ is a causal net on A,*
3. *$\mathsf{link}(\sigma) = \mathsf{link}(\tau)$.*

In this case we write $\sigma \uparrow \tau$ and $\sigma \vee \tau$ is the least upper bound of σ and τ for $\subseteq$.

It is a direct consequence of Proposition 1 that any causal net σ is included in a unique causal invariant $\sigma^\uparrow : A$, defined as: $\sigma^\uparrow = \bigvee_{\sigma \subseteq \tau} \tau$, where τ ranges over causal nets.

Lemma 4. *For $\sigma, \tau : A$ causal nets, $\sigma \uparrow \tau$ iff $\sigma^\uparrow = \tau^\uparrow$. Moreover, if σ and τ are causal invariants, $\sigma \uparrow \tau$ if and only if $\sigma = \tau$.*

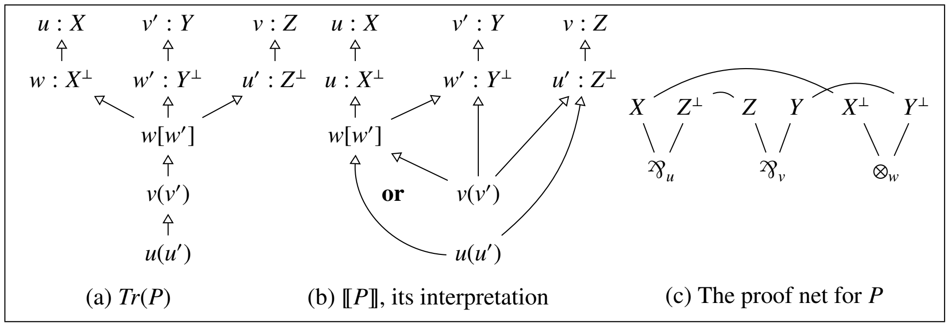


Fig. 5: Interpreting $P = u(u').\, v(v').\, w[w'].\, ([u \leftrightarrow w] \mid ([w' \leftrightarrow v'] \mid [u' \leftrightarrow v]))$ in the context $u : X ⅋ Z^\perp, v : Z ⅋ Y, w : X^\perp \otimes Y^\perp$

The interpretation of a proof $P \triangleright \Delta$ is simply defined as $[\![P]\!] = Tr(P)^\uparrow$. Figure 5c illustrates the construction on a proof of MLL+mix. The interpretation features a disjunctive causality, as the tensor can be introduced as soon as *one* of the two pars has been.

Defining $\mathsf{link}(P) = \mathsf{link}(Tr(P))$, we have from Lemma 4: $\mathsf{link}(P) = \mathsf{link}(Q)$ if and only if $[\![P]\!] = [\![Q]\!]$. This implies that our model has the same equational theory than the proof nets of [20]. Such proof nets are already complete:

Theorem 3 ([20]). *For P, Q two proofs of Γ, we have $P \equiv Q$ iff $\mathsf{link}(P) = \mathsf{link}(Q)$.*

As a corollary, we get:

Theorem 4. *For cut-free proofs P, Q we have $P \equiv Q$ iff $[\![P]\!] = [\![Q]\!]$.*

We also provide an inductive proof not using the result of [20] in Appendix. A consequence of this result, along with *strong* sequentialisation is: $[\![P]\!] = \bigcup_{Q \equiv P} Tr(Q)$. This equality justifies our terminology of "causal completeness", as for instance it implies that the minimal events of $[\![P]\!]$ correspond exactly the possible rules in P that can be pushed to the front using the commuting conversions.

6.2 The category of causal invariants

So far we have focused on the static. Can we integrate the dynamic aspect of proofs as well? In this section, we show that causal invariants organise themselves in a category. First, we show that causal nets are stable under composition:

Lemma 5. *If $\sigma : A^\perp \parallel B$ and $\tau : B^\perp \parallel C$ are causal nets, then so is $\tau \odot \sigma$.*

Note that totality requires acyclicity (and deadlock-freedom) to be stable under composition. However, causal invariants are not stable under composition: $\tau \odot \sigma$ might not be maximal, even if τ and σ are. Indeed, during the interaction, some branches of τ will not be explored by σ and vice-versa which can lead to new allowed reorderings. However, we can always embed $\tau \odot \sigma$ into $(\tau \odot \sigma)^\uparrow$:

Lemma 6. *Rooted games and causal invariants form a category **CInv**, where the composition of $\sigma : A^\perp \parallel B$ and $\tau : B^\perp \parallel C$ is $(\tau \odot \sigma)^\uparrow$ and the identity on A is $\text{æ}_A^\uparrow$.*

Note that the empty game is an object of **CInv**, as we need a monoidal unit.

Monoidal-closed structure. Given two games A and B we define $A \otimes B$ as $\texttt{send}^+ \cdot (A \parallel B)$, and 1 as the empty game. There is an obvious isomorphism $A \otimes 1 \cong A$ and $A \otimes (B \otimes C) \cong (A \otimes B) \otimes C$ in **CInv**. We now show how to compute directly the functorial action of $\otimes$, without resorting to $^\uparrow$. Consider $\sigma \in \mathbf{CInv}(A, B)$ and $\tau \in \mathbf{CInv}(C, D)$. Given $x \in \mathscr{C}((A \otimes C)^\perp \parallel (B \otimes D))$, we define $x\langle\sigma\rangle = x \cap (A^\perp \parallel B)$ and $x\langle\tau\rangle = x \cap (C^\perp \parallel D)$. If $x\langle\sigma\rangle \in \sigma$ and $x\langle\tau\rangle \in \tau$, we say that x is connected when there exists cells $\mathfrak{a}, \mathfrak{b}, \mathfrak{c}$ and $\mathfrak{d}$ of A, B, C and D respectively such that $\mathfrak{a} \leftrightsquigarrow_{x\langle\sigma\rangle,\sigma} \mathfrak{c}$ and $\mathfrak{b} \leftrightsquigarrow_{x\langle\tau\rangle,\tau} \mathfrak{d}$. We define:

$$\sigma \otimes \tau = \left\{ \begin{array}{l} x \in \mathscr{C}((A \otimes C)^\perp \parallel (B \otimes D)) \text{ such that :} \\ \quad (1)\ x\langle\sigma\rangle \in \sigma \text{ and } x\langle\tau\rangle \in \tau \\ \quad (2) \text{ if } x \text{ is connected and contains } \texttt{send}^+, \text{ then } \texttt{send}^- \in x \end{array} \right\}$$

In (2), $\texttt{send}^-$ refers to the minimal move of $(A \otimes C)^\perp$ and $\texttt{send}^+$ to the one of $B \otimes D$. (2) ensures that $\sigma \otimes \tau$ is acyclic.

Lemma 7. *The tensor product defines a symmetric monoidal structure on* ***CInv***.

Define $A \parr B = (A^\perp \otimes B^\perp)^\perp$, $\perp = 1 = \emptyset$ and $A \multimap B = A^\perp \parr B$.

Lemma 8. *We have a bijection* $\parr_{B,C}$ *between causal invariants on* $A \parallel B \parallel C$ *and on* $A \parallel (B \parr C)$. *As a result, there is an adjunction* $A \otimes \_ \dashv A \multimap \_$.

Lemma 8 implies that $\mathbf{CInv}((A \multimap \perp) \multimap \perp) \simeq \mathbf{CInv}(A)$, and **CInv** is $*$-autonoumous.

Cartesian products. Given two games A, B in **CInv**, we define their product $A \mathbin{\&} B = \texttt{inl}^- \cdot A + \texttt{inr}^- \cdot B$. We show how to construct the pairing of two causal invariants concretely. Given $\sigma \in \mathbf{CInv}(A, B)$ and $\tau \in \mathbf{CInv}(A, C)$, we define the common behaviour of σ and τ on A to be those $x \in \mathscr{C}(A^\perp) \cap \sigma \cap \tau$ such that for all $\mathfrak{a}, \mathfrak{a}'$ outside of x with positive meet, $\mathfrak{a} \leftrightsquigarrow_{x,\sigma} \mathfrak{a}'$ iff $\mathfrak{a} \leftrightsquigarrow_{x,\tau} \mathfrak{a}'$ We write $\sigma \cap_A \tau$ for the set of common behaviours of σ and τ and define: $\langle\sigma, \tau\rangle = (L^- \cdot \sigma) \cup (R^- \cdot \tau) \cup \sigma \cap_A \tau$. The projections are defined using copycat: $\pi_1 = \{x \in \mathscr{C}((A \mathbin{\&} B)^\perp \parallel A) \mid x \cap (A^\perp \parallel A) \in \mathfrak{c}\!\!\mathfrak{c}_A^\uparrow\}$ (and similarly for π_2).

Theorem 5. ***CInv*** *has products. As it is also* $*$*-autonomous, it is a model of MALL.*

It is easy to see that the interpretation of MALL$^-$ in **CInv** following the structure is the same as $[\![\cdot]\!]$, however it is computed compositionally without resorting to the $^\uparrow$ operator. We deduce that our interpretation is invariant by cut-elimination: if $P \to Q$, then $[\![P]\!] = [\![Q]\!]$. Putting the pieces together, we get the final result.

Theorem 6. ***CInv*** *is an injective and fully complete model of MALL*$^-$.

7 Extensions and related work

The model provides a representation of proofs which retains only the necessary sequentiality. We study the phenomenon in Linear Logic, but commuting conversions of additives arise in other languages, eg. in functional languages with sums and products, where proof nets do not necessarily exist. Having an abstract representation of which reorderings are allowed could prove useful (reasoning on the possible commuting conversions in a language with sum types is notoriously difficult).

Extensions. Exponentials are difficult to add, as their conversions are not as canonical as those of MALL. Cyclic proofs [2] could be accomodated via recursive event structures.

Adding multiplicative units while keep determinism is difficult, as their commuting conversion is subtle (*e.g.* conversion for MLL is PSPACE-complete [16]), and exhibit apparent nondeterminism. For instance the following proofs are convertible in MLL:

$$a().\, b[] \mid c[] \equiv a().\,(b[] \mid c[]) \equiv b[] \mid a().\, c[] \triangleright a : \bot, b : 1, c : 1$$

where $a().\,P$ is the process counterpart to introduction of $\bot$ and $a[]$ of 1. Intuitively, $b[]$ and $c[]$ can be performed at the start, but as soon as one is performed, the other has to wait for the input on a. This cannot be modelled inside deterministic general event structures, as it is only deterministic against an environment that will emit on b. In contrast, proofs of MALL$^-$ remain deterministic even if their environment is not total.

We would also be interested in recast multifocusing [7] in our setting by defining a class of focussed causal nets, where there are no concurrency betwen positive and negative events, and show that sequentialisation always give a focused proof.

Related work. The first fully complete model of MALL$^-$ is based on closure operators [1], later extended to full Linear Logic [22]. True concurrency is used to define innocence, on which the full completeness result rests. However their model does not take advantage of concurrency to account for permutations, as strategies are sequential. This investigation has been extended to concurrent strategies by Mimram and Melliès [23,24]. De Carvalho showed that the relational model is injective for MELL [9]. In another direction, [4] provides a fully complete model for MALL without game semantics, by using a glueing construction on the model of hypercoherences. [19] explores proof nets a weaker theory of commuting conversions for MALL.

The idea of having intermediate representations between proof nets and proofs has been studied by Faggian and coauthors using l-nets [13,12,8,14,11], leading to a similar analysis to ours: they define a space of causal nets as partial orders and compare different versions of proofs with varying degree of parallelism. Our work recasts this idea using event structures and adds the notion of causal completeness: keeping jumps that cannot be undone by a permutation, which leads naturally to step outside partial orders, as well as full completeness: which causal nets can be strongly sequentialised?

The notion of dependency between logical rules has also been studied in [3] in the case of MLL. From a proof net R, they build a partial order $D_{\parr,\otimes}(R)$ which we believe is very related to $[\![P]\!]$ where P is a sequentialisation of R. Indeed, in the case of MLL *without MIX* a partial order is enough to capture the dependency between rules. The work [10] shows that permutation rules of Linear Logic, understood as asynchronous optimisations on processes, are included in the observational equivalence. [17] studies mutual embedding between polarised proof nets [21] and the control π-calculus [18].

Acknowledgements. We would like to thank Willem Heijltjes, Domenico Ruoppolo, and Olivier Laurent for helpful discussions, and the anonymous referees for their insightful comments. This work has been partially sponsored by: EPSRC EP/K034413/1, EP/K011715/1, EP/L00058X/1, EP/N027833/1, and EP/N028201/1.

References

1. Samson Abramsky and Paul-André Melliès. Concurrent games and full completeness. In *14th Annual IEEE Symposium on Logic in Computer Science, Trento, Italy, July 2-5, 1999*, pages 431–442, 1999. URL: `http://dx.doi.org/10.1109/LICS.1999.782638`, `doi:10.1109/LICS.1999.782638`.
2. David Baelde, Amina Doumane, and Alexis Saurin. Infinitary Proof Theory: the Multiplicative Additive Case. In *CSL*, volume 62 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 42:1–42:17. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016.
3. Marc Bagnol, Amina Doumane, and Alexis Saurin. On the dependencies of logical rules. In *Foundations of Software Science and Computation Structures - 18th International Conference, FoSSaCS 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015. Proceedings*, pages 436–450, 2015. URL: `https://doi.org/10.1007/978-3-662-46678-0_28`, `doi:10.1007/978-3-662-46678-0\_28`.
4. Richard Blute, Masahiro Hamano, and Philip J. Scott. Softness of hypercoherences and MALL full completeness. *Ann. Pure Appl. Logic*, 131(1-3):1–63, 2005. URL: `https://doi.org/10.1016/j.apal.2004.05.002`, `doi:10.1016/j.apal.2004.05.002`.
5. Luís Caires and Frank Pfenning. Session types as intuitionistic linear propositions. In *CONCUR 2010*, pages 222–236, 2010.
6. Simon Castellan, Pierre Clairambault, and Glynn Winskel. Observably deterministic concurrent strategies and intensional full abstraction for parallel-or. In *2nd International Conference on Formal Structures for Computation and Deduction, FSCD 2017, September 3-9, 2017, Oxford, UK*, pages 12:1–12:16, 2017. URL: `https://doi.org/10.4230/LIPIcs.FSCD.2017.12`, `doi:10.4230/LIPIcs.FSCD.2017.12`.
7. Kaustuv Chaudhuri, Dale Miller, and Alexis Saurin. Canonical sequent proofs via multi-focusing. In *Fifth IFIP International Conference On Theoretical Computer Science - TCS 2008, IFIP 20th World Computer Congress, TC 1, Foundations of Computer Science, September 7-10, 2008, Milano, Italy*, pages 383–396, 2008. URL: `https://doi.org/10.1007/978-0-387-09680-3_26`, `doi:10.1007/978-0-387-09680-3\_26`.
8. Pierre-Louis Curien and Claudia Faggian. L-nets, strategies and proof-nets. In C.-H. Luke Ong, editor, *Computer Science Logic, 19th International Workshop, CSL 2005, 14th Annual Conference of the EACSL, Oxford, UK, August 22-25, 2005, Proceedings*, volume 3634 of *Lecture Notes in Computer Science*, pages 167–183. Springer, 2005. URL: `http://dx.doi.org/10.1007/11538363_13`, `doi:10.1007/11538363_13`.
9. Daniel de Carvalho. The relational model is injective for multiplicative exponential linear logic. In *25th EACSL Annual Conference on Computer Science Logic, CSL 2016, August 29 - September 1, 2016, Marseille, France*, pages 41:1–41:19, 2016. URL: `https://doi.org/10.4230/LIPIcs.CSL.2016.41`, `doi:10.4230/LIPIcs.CSL.2016.41`.
10. Henry DeYoung, Luís Caires, Frank Pfenning, and Bernardo Toninho. Cut reduction in linear logic as asynchronous session-typed communication. In *CSL*, pages 228–242, 2012.
11. Paolo Di Giamberardino. Jump from Parallel to Sequential Proofs: Additives. Technical report, 2011. URL: `https://hal.archives-ouvertes.fr/hal-00616386`.
12. Claudia Faggian and François Maurel. Ludics nets, a game model of concurrent interaction. In *20th IEEE Symposium on Logic in Computer Science (LICS 2005), 26-29 June 2005, Chicago, IL, USA, Proceedings*, pages 376–385. IEEE Computer Society, 2005. URL: `http://dx.doi.org/10.1109/LICS.2005.25`, `doi:10.1109/LICS.2005.25`.
13. Claudia Faggian and Mauro Piccolo. A graph abstract machine describing event structure composition. *Electr. Notes Theor. Comput. Sci.*, 175(4):21–36, 2007. URL: `https://doi.org/10.1016/j.entcs.2007.04.014`, `doi:10.1016/j.entcs.2007.04.014`.

14. Paolo Di Giamberardino and Claudia Faggian. Jump from parallel to sequential proofs: Multiplicatives, 2006. URL: `https://doi.org/10.1007/11874683_21`, `doi:10.1007/11874683\_21`.
15. J.Y. Girard. Linear logic. *Theoretical computer science*, 50(1):1–101, 1987.
16. Willem Heijltjes and Robin Houston. No Proof Nets for MLL with Units: Proof Equivalence in MLL is PSPACE-complete. CSL-LICS'14, pages 50:1–50:10. ACM, 2014.
17. Kohei Honda and Olivier Laurent. An exact correspondence between a typed pi-calculus and polarised proof-nets. *Theor. Comput. Sci.*, 411(22-24):2223–2238, 2010. URL: `https://doi.org/10.1016/j.tcs.2010.01.028`, `doi:10.1016/j.tcs.2010.01.028`.
18. Kohei Honda, Nobuko Yoshida, and Martin Berger. Process Types as a Descriptive Tool for Interaction: Control and the Pi-Calculus. In *Joint 25th International Conference on Rewriting Techniques and Applications and 12th International Conference on Typed Lambda Calculi and Applications*, volume 8560 of *LNCS*, pages 1–20. Springer, 2014.
19. Dominic J. D. Hughes and Willem Heijltjes. Conflict nets: Efficient locally canonical MALL proof nets. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 437–446, 2016. URL: `http://doi.acm.org/10.1145/2933575.2934559`, `doi:10.1145/2933575.2934559`.
20. Dominic J. D. Hughes and Rob J. van Glabbeek. Proof nets for unit-free multiplicative-additive linear logic. *ACM Trans. Comput. Log.*, 6(4):784–842, 2005.
21. Olivier Laurent. Polarized proof-nets and $\lambda\mu$-calculus. *Theor. Comput. Sci.*, 290(1):161–188, January 2003. URL: `http://dx.doi.org/10.1016/S0304-3975(01)00297-3`, `doi:10.1016/S0304-3975(01)00297-3`.
22. Paul-André Melliès. Asynchronous games 4: A fully complete model of propositional linear logic. In *20th IEEE Symposium on Logic in Computer Science (LICS 2005), 26-29 June 2005, Chicago, IL, USA, Proceedings*, pages 386–395, 2005. URL: `http://dx.doi.org/10.1109/LICS.2005.6`, `doi:10.1109/LICS.2005.6`.
23. Paul-André Melliès and Samuel Mimram. Asynchronous games: Innocence without alternation. In *CONCUR 2007 - Concurrency Theory, 18th International Conference*, pages 395–411, 2007.
24. Samuel Mimram. *Sémantique des jeux asynchrones et réécriture 2-dimensionnelle. (Asynchronous Game Semantics and 2-dimensional Rewriting Systems)*. PhD thesis, Paris Diderot University, France, 2008. URL: `https://tel.archives-ouvertes.fr/tel-00338643`.
25. Philip Wadler. Propositions as sessions. *J. Funct. Program.*, 24(2-3):384–418, 2014.
26. Glynn Winskel. Event structures. In *Petri Nets: Central Models and Their Properties, Advances in Petri Nets 1986, Part II, Proceedings of an Advanced Course, Bad Honnef, 8.-19. September 1986*, pages 325–392, 1986. `doi:10.1007/3-540-17906-2_31`.

A Proofs of § 3 (Games)

Lemma 9. *Let $x, y \in \tau \circledast \sigma$ such that there are no negative minimal conflicts in $(x \cup y) \cap (A \parallel C)$. Then $x \cup y \in \mathscr{C}(A \parallel B \parallel C)$.*

Proof. Consider a minimal pair (x, y) where this does not hold. By minimality, there is a unique minimal conflict $b \in x$ and $b' \in y$ with $b \sim\!\sim b'$ in $A \parallel B \parallel C$. Since games are race-free, b and b' have the same polarity. By assumption on x and y, b and b' cannot be negative in $A^\perp \parallel C$, so this implies that they are either positive in $A^\perp \parallel B$ or in $B^\perp \parallel C$. Assume the former. Then $x\langle\sigma\rangle$ and $y\langle\sigma\rangle$ have no negative minimal conflict, hence $x\langle\sigma\rangle \cup y\langle\sigma\rangle$, absurd.

Lemma 1. *If $\sigma : A^\perp \parallel B$ and $\tau : B^\perp \parallel C$ are causal structs then $\tau \odot \sigma$ is a causal struct.*

Proof. Determinism. Consider $x, y \in \tau \odot \sigma$ with no negative minimal conflict. Write x' and y' for some witness in $\tau \circledast \sigma$. Then x' and y' satisfy the condition of Lemma 9, hence we have $x' \cup y' \in \mathscr{C}(A \parallel B \parallel C)$. By determinism of σ and τ we conclude that $x' \cup y' \in (\sigma \parallel C) \cap (A \parallel \tau)$. To conclude, we just need to show it is reachable. This is trivial because we know that x' and y' are both reachable so we merge their covering chains.

Coincidence-freeness. By construction, $\tau \circledast \sigma$ is coincidence-free, which implies that $\tau \odot \sigma$ is also coincidence-free.

Lemma 10. *If A is a game and a a minimal event of A, then A/a is a game. Moreover, if $A = [\![\Delta]\!]$, then there exists a context Δ/a such that $[\![\Delta/a]\!] = A/a$.*

Proof. That A/a is a game is an easy verification.

If A is a type, and $a \in \min([\![A]\!])$, we define a context Δ such that $[\![\Delta]\!] = [\![A]\!]/a$. By duality, we define it only on positive connectives: $(A \oplus B)/\mathtt{L}^+ = u : A$ and $(A \otimes B)/\mathtt{send}^+ = u : A, v : B$. This definition can be extended to contexts in a straightforward manner.

B Proofs of § 4 (Acyclic causal structs)

Lemma 11. *Consider $\sigma : A$ a causal struct and $x \in \sigma$. If $a \in x$, there exists a prime configuration $y_{a'} \subseteq x$ of $a' \in \max(x)$ such that $a \in y_{a'}$.*

Proof. Note that x must be non-empty. Write z for the union of all prime configurations within x of all elements of $\max(x)$. The lemma amounts to showing $z = x$. Assume that $z \neq x$. Then consider a covering chain from z to x and write a_0 the last element in it. Then by definition $z \setminus \{a_0\} \in \sigma$ hence $a_0 \in \max(x)$ and $a_0 \in z$ absurd.

Lemma 12. *Let y a prime configuration of $a \in A$ compatible with x and $a \notin x$. Then for any $a' \in y \setminus x$, $cell(a') \leftrightsquigarrow_{x,\sigma} cell(a)$.*

Proof. We proceed by induction on the size of y. If a' is maximal in $y \setminus \{a\}$, we are done. Otherwise, by Lemma 11, there exists $a_0 \in \max(y \setminus \{a\})$ and a prime configuration y_{a_0} of a_0 such that $a' \in y_{a_0}$. By induction, we know that $a' \leftrightsquigarrow_{x,\sigma} a_0$, and by definition $a_0 \rightarrowtriangle_{x,\sigma} a$, hence we conclude.

B.1 Interaction

Lemma 13. *For $x \in \tau \circledast \sigma$, there are no cycles of the form:* $\mathfrak{b}_0 \leftrightsquigarrow_{x\langle\sigma\rangle,\sigma} \mathfrak{b}_1 \leftrightsquigarrow_{x\langle\tau\rangle,\tau} \mathfrak{b}_2 \leftrightsquigarrow_{x\langle\sigma\rangle,\sigma} \dots \leftrightsquigarrow_{x\langle\sigma\rangle,\sigma} \mathfrak{b}_{n-1} \leftrightsquigarrow_{x\langle\tau\rangle,\tau} \mathfrak{b}_0$ *where* $\mathfrak{b}_i \wedge \mathfrak{b}_{i+1} \in x$ *(operations are modulo n).*

Proof. We proceed on the number of alternances σ/τ of the cycle.

- If there is only one alternance, we have $\mathfrak{b}_0 \leftrightsquigarrow_{x_\sigma,\sigma} \mathfrak{b}_1$ and $\mathfrak{b}_1 \leftrightsquigarrow_{x_\tau,\tau} \mathfrak{b}_0$. Then $\mathfrak{b}_0 \wedge \mathfrak{b}_1 \in x\langle\sigma\rangle \cap x\langle\tau\rangle$ it has to be both negative and positive because of acyclicity of σ and τ, which is absurd.
- Write m_i for $\mathfrak{b}_i \wedge \mathfrak{b}_{i+1}$. Since games are trees, two consecutives m_i must be comparable. Moreover, by acyclicity of σ and τ, m_{2i} must be negative while m_{2i+1} must be positive in B. As a result $m_i \neq m_{i+1}$ hence $m_i < m_{i+1}$ or $m_{i+1} < m_i$.
  Consider a covering chain of x: all the m_i occur. Consider the step y at which all the m_i occured except one, m_k. Assume eg that k is even so that $\mathfrak{b}_k \leftrightsquigarrow_{x\langle\sigma\rangle,\sigma} \mathfrak{b}_{k+1}$.
  Note that we must have $m_{k+1} < m_k$ since m_k is maximal among the m_i for $\leq_B$. We have:
  $$\mathfrak{b}_k \leftrightsquigarrow_{y\langle\tau\rangle,\tau} \mathrm{cell}(m_k) \leftrightsquigarrow_{y\langle\tau\rangle,\tau} \mathfrak{b}_{k+1} \leftrightsquigarrow_{y\langle\tau\rangle,\tau} b_{k+2},$$
  and we can consider the cycle $(\mathfrak{b}_0, \dots, \mathfrak{b}_k, \mathfrak{b}_{k+2}, \dots, \mathfrak{b}_n)$ in y and apply the induction hypothesis since $\mathfrak{b}_k \wedge \mathfrak{b}_{k+2} < \mathfrak{m}_k$ hence it belongs in y.

Lemma 14. *For $x \in \tau \circledast \sigma$, if* $\mathfrak{b} \leftrightsquigarrow_{x,\tau\circledast\sigma} \mathfrak{b}'$ *where* $\mathfrak{b}$ *and* $\mathfrak{b}'$ *are cells of B, then there exists a non trivial sequence (ie. $n \geq 1$)* $\mathfrak{b} \leftrightsquigarrow_{x\langle\tau\rangle,\tau} \mathfrak{b}_1 \leftrightsquigarrow_{x\langle\sigma\rangle,\sigma} \dots \leftrightsquigarrow_{x\langle\tau\rangle,\tau} \mathfrak{b}_n \leftrightsquigarrow_{x\langle\sigma\rangle,\sigma} \mathfrak{b}'$*, such that all the* $\mathfrak{b} \wedge \mathfrak{b}_1, \mathfrak{b}_1 \wedge \mathfrak{b}_2, \dots, \mathfrak{b}_n \wedge \mathfrak{b}'$ *are all in x.*

For the case $n = 0$, $\mathfrak{b} \wedge \mathfrak{b}'$ *is not guaranteed to be in x.*

Proof. We use $\leftrightsquigarrow_\sigma$ for $\leftrightsquigarrow_{x\langle\sigma\rangle,\sigma}$ and $\leftrightsquigarrow_\tau$ for $\leftrightsquigarrow_{x\langle\tau\rangle,\tau}$ to simplify notations. *Constructing a sequence.* We simply need to show the result when $\mathfrak{c} \rightarrowtriangle_{x,\tau\circledast\sigma} \mathfrak{c}'$. Consider a prime configuration y of $a' \in \mathfrak{c}'$, such that $\mathfrak{c}$ intersects y at a and $a \in \max(y \setminus \{a'\})$. Write $z_{\sigma\|C}$ and $z_{A\|\tau}$ for some prime configurations inside y of a' in $\sigma \parallel C$ and $A \parallel \tau$, respectively.

If $a \notin z_{\sigma\|C} \cup z_{A\|\tau}$, then we have $y \setminus \{a\} = y \setminus \{a, a'\} \cup z_{\sigma\|C} \in \sigma \parallel C$ by determinism of $\sigma \parallel C$ and similarly $y \setminus \{a\} \in A \parallel \tau$ which implies that $y \setminus \{a\} \in \tau \circledast \sigma$, contradicting the fact that a' is the only maximal element of y.

Assume eg. $a \in z_{\sigma\|C}$. If $a \in C$, then $a \leq_C a'$ and we deduce that $a \in z_{A\|\tau} \cap (B^\perp \parallel C)$, hence $a \leftrightsquigarrow_{x\langle\tau\rangle,\tau} a'$. Otherwise, $a \in z_{\sigma\|C} \cap (A^\perp \parallel B)$, then a' is also in $A^\perp \parallel C$ hence and $z_{\sigma\|C}$ is actually a prime configuration of a' for σ, hence $a \leftrightsquigarrow_{x\langle\sigma\rangle,\sigma} a'$ by Lemma 12.

Showing that the meets are in x. We have seen than one sequence exists. We now consider a sequence of minimal length and show that all the required meets are in x. If the sequence is empty, there is nothing to show. Otherwise:

- If $\mathfrak{b} \wedge \mathfrak{b}_1 \notin x$. Assume that $\mathfrak{b} \leftrightsquigarrow_\tau \mathfrak{b}_1$. Then:
  $$\mathfrak{b} \leftrightsquigarrow_\sigma \mathfrak{b} \wedge \mathfrak{b}_1 \leftrightsquigarrow_\sigma \mathfrak{b}_1 \leftrightsquigarrow_\sigma \mathfrak{b}_2$$
  As a result, we can remove $\mathfrak{b}_1$ from the sequence, and get a shorter one, contradicting the minimality hypothesis.

- Similarly if $\mathfrak{b}_n \wedge \mathfrak{b}' \notin x$
- If $\mathfrak{b}_i \wedge \mathfrak{b}_{i+1} \notin x$, we proceed similarly by removing b_i from the sequence to get a shorter sequence.

Theorem 1. *For acyclic causal structs $\sigma : A^\perp \parallel B$ and $\tau : B^\perp \parallel C$, (1) their interaction is* deadlock-free*: $\tau \circledast \sigma = (\sigma \parallel C) \cap (A \parallel \tau)$; and (2) the causal struct $\tau \odot \sigma$ is acyclic.*

Proof. Deadlock-freeness. Consider $y \in (\sigma \parallel C) \cap (A \parallel \tau)$ and x the maximal subconfiguration of y in $\tau \circledast \sigma$. If $x = y$ we are done. Otherwise, consider all the immediate extensions of x in $\sigma \parallel C$ and $A \parallel \tau$ towards y: they must all be in B since extensions in A or C cannot be blocked. Hence write X for the set of such extensions. For $b, b' \in X$ $b <_\sigma b'$ when b belongs to a prime configuration of b' for σ in y_σ, and define similarly $<_\tau$. Consider the relation $(<_\sigma \cup <_\tau)$:

- if it has a minimal event b, then $x \overset{b_k}{\longrightarrow\!\!\!\!\subset}$ both in σ and τ. Hence $y = x \cup \{b_k\} \in \tau \circledast \sigma$: absurd.
- if it does not have a minimal event, since it is finite there must be a loop. Since $<_\sigma$ is included in $\leftrightsquigarrow_{x\langle\sigma\rangle,\sigma}$ and $<_\tau$ in $\leftrightsquigarrow_{x\langle\tau\rangle,\tau}$, we contradict Lemma 13 as they are all immediate extensions of x in σ or τ, hence $b \wedge b' \in x$ for $b, b' \in X$.

This implies that the interaction is deadlock-free.

Composition is acyclic. Consider a configuration $x \in \tau \odot \sigma$, and a witness $y \in \tau \circledast \sigma$. Suppose that we have two cells (eg. in C) $\mathfrak{c}$ and $\mathfrak{c}'$ with positive meet such that $\mathfrak{c} \leftrightsquigarrow_{x,\tau\odot\sigma} \mathfrak{c}'$ – in particular $\mathfrak{c} \leftrightsquigarrow_{y,\tau\circledast\sigma} \mathfrak{c}'$. Suppose that $\mathfrak{c} \wedge \mathfrak{c}' \notin x$.

By Lemma 14, there are two cases:

- The communication path is entirely contained in τ, ie. we have $\mathfrak{c} \leftrightsquigarrow_{y\langle\tau\rangle,\tau} \mathfrak{c}'$. Then we conclude by acyclicity of τ.
- Otherwise, the path must be of the form

$$\mathfrak{c} \leftrightsquigarrow_{x\langle\tau\rangle,\tau} \mathfrak{b}_0 \leftrightsquigarrow_{x_\sigma,\sigma} \mathfrak{b}_1 \ldots \leftrightsquigarrow_{x\langle\sigma\rangle,\sigma} \mathfrak{b}_n \leftrightsquigarrow_{x\langle\tau\rangle,\tau} \mathfrak{c}'$$

  such that all the meets are in y, and at least one σ step. Since $\mathfrak{c} \wedge \mathfrak{c}' \notin x$, $\mathfrak{b}_0 \leftrightsquigarrow_{x\langle\tau\rangle,\tau} \mathfrak{c} \leftrightsquigarrow_{x_\tau,\tau} \mathfrak{c}' \leftrightsquigarrow_{x\langle\tau\rangle,\tau} \mathfrak{b}_n$, and we have a cycle contradicting Lemma 13.

C Proofs of § 5 (Causal nets)

Lemma 3. *σ_k is a causal net on A_k and we have $\sigma/a = \sigma_0 \parallel \ldots \parallel \sigma_n$.*

Proof. We know that σ/a is a causal net. Coincidence-freeness and determinism are inherited from σ/a, same for totality and well-linking since A_k is upward closed in A. Acyclicity is a simple observation: any chain in σ_k is also a chain in σ/a.

We clearly have $\sigma_0 \parallel \ldots \parallel \sigma_n \subseteq \sigma/a$. If $x \in \sigma/a$, we can form $x_k = x \cap A_k$. To conclude we need to show that $x_k \in \sigma/a$. If it is not the case, there exists a prime configuration y in x of $a_k \in x_k$ such that y intersects A_i at a_i for $i \neq k$. This implies that $a_k \leftrightsquigarrow_{\{a\},\sigma} a_i$. By definition of A_k, this implies that that $i = k$, absurd.

Lemma 15. *For $\sigma, \tau : A$ be causal nets,* $\max \sigma = \max \tau$ *and* $\ell_x^\sigma = \ell_x^\tau$ *is equivalent to* $\mathsf{link}(\sigma) = \mathsf{link}(\tau)$*.*

Proof. The direct implication is clear. We show the converse implication. Consider $x \in \max \sigma$. We know that $\ell_x^\sigma \in \mathsf{link}(\tau)$ so there exists $y \in \max \tau$ with $\ell_y^\tau = \ell_x^\tau$. This implies that x and y have the same atoms. Since x and y are maximal in the game, it means that all their maximal elements are atoms, hence $x = y$, and we conclude.

Theorem 2. *If $\sigma : A$ is a causal net, then σ is strongly sequentialisable.*

Proof. Consider $x \in \sigma$, and choose $a \in \min(\sigma) \cap x$, if possible not atom. If a is an atom, it means that x only contains atoms, and by well-linking $\sigma = Tr(P)$ where P is a parallel composition of axioms.

Otherwise, we proceed by induction on $\Delta = u_1 : T_1 \dots, u_n : T_n$, and case distinction on $a \in [\![T_k]\!]$. We consider by Lemma 10 the context Δ/a corresponding to $[\![\Delta]\!]/a$.

- If a is $\mathtt{inl}^+$ or $\mathtt{inr}^+$: By Lemma 2, σ/a is a causal net, so there exists a a proof $Q \triangleright \Delta/a$ embeded in σ/a containing $x \setminus \{a\}$. Then we let $P = u_k[\mathtt{inl}].\, Q$.
- If a is a $⅋^-$, then same reaoning as for L^+ using again Lemma 2.
- If a is $\mathtt{inl}^-$, then write a' for the corresponding $\mathtt{inr}^-$. As before, by induction we get $Q_l \triangleright \Delta/a$ and $Q_r \triangleright \Delta/a'$ and $P = u_k.\mathtt{case}\,(Q_l, Q_r)$ is the desired process.
- If a is $\otimes$, then we know that $T_k = S \otimes S'$. We write $\Delta/a = u_1 : T_1, \dots, u_{k-1} : T_{k-1}, u_k : S, v : S', u_{k+1} : T_{k+1}, \dots, u_n : T_n$. By Lemma 3, we know that $\sigma/a = \sigma_0 \parallel \sigma_1 \parallel \sigma_2$, and we apply the induction hypothesis to get Q_0, Q_1 and Q_2, and let $P = u_k[v].\, ((Q_0|Q_1)|Q_2)$ (It does not matter where Q_0 is placed since it does not communicate with u_k or v.)

D Proofs of § 6 (Causal invariants)

D.1 Causal invariants as maximal proofs

Lemma 16. *For two causal nets $\sigma, \tau : A$, if $\sigma \subseteq \tau$, then we have* $\max \sigma = \max \tau$*.*

Proof. We show that if $x \in \sigma$ such that $x \overset{a}{\longrightarrow\!\!\subset}$ in τ, then there exists an extension $x \subseteq y \in \sigma$ with $a \in y$, which entails the desired result.

We proceed by reverse inclusion on $\mathscr{C}(A)$ which is well-founded because A is finite. Assume $x \in \sigma$ and $x \overset{a}{\longrightarrow\!\!\subset}$ in τ. Since x is not maximal in $\mathscr{C}(A)$, we know that $x \overset{a'}{\longrightarrow\!\!\subset}$ in σ. If $a \sim a'$, by determinism of τ they are negative and $x \overset{a}{\longrightarrow\!\!\subset}$ in σ by totality of σ. Otherwise, we have $x \cup \{a'\} \overset{a}{\longrightarrow\!\!\subset}$ in τ by determinism, hence we apply the induction hypothesis.

Lemma 17. *If $\sigma \subseteq \tau$, we have* $\mathsf{link}(\sigma) = \mathsf{link}(\tau)$*.*

Proof. We already know by Lemma 16 that $\max(\sigma) = \max(\tau)$. We use the characterisation of Lemma 15, and show that for every $x \in \max \sigma = \max \tau$, we have $\ell_x^\sigma = \ell_x^\tau$.

Consider $e \in (\max_A x)^+$, and y a prime configuration of e for σ in x. Write a_σ such that $\ell_x^\sigma(a_\sigma) = e$ and similarly a_τ such that $\ell_x^\tau(a_\tau) = a$. Since $\sigma \subseteq \tau$, $y \in \tau$ and we write y' for a prime configuration of e for τ in y so that $y' \subseteq y \subseteq x$. We know that we have:

- $\max_\sigma(y \setminus \{e\}) = \{a_\sigma\}$
- $\max_\tau(y' \setminus \{e\}) = \{a_\tau\}$
- $\max_\tau(y \setminus \{e\}) \subseteq \max_\sigma(y \setminus \{e\})$ because $\sigma \subseteq \tau$
- $\max_\tau(y' \setminus \{e\}) \subseteq \max_\tau(y \setminus \{e\})$ because $y' \subseteq y$.

All these together imply that $a_\sigma = a_\tau$ as desired.

Lemma 18. *Let $\sigma : A$ acyclic and total and $\tau : A$ total with $\sigma \subseteq \tau$. Then, if $x \in \sigma$, $\mathfrak{a} \leftrightsquigarrow_{x,\tau} \mathfrak{a}'$ implies $\mathfrak{a} \leftrightsquigarrow_{x,\sigma} \mathfrak{a}'$.*

Proof. Assume that $e \rightarrowtriangle_{x,\tau} e'$ and let y be a prime configuration of e' in τ such that $e \in x$. Because $\max\sigma = \max\tau$ by Lemma 16, there exists $y \subseteq z \in \sigma$. Consider a minimal configuration y' in z containing y. By construction it is a prime configuration of e' containing e hence $e \leftrightsquigarrow_{x,\sigma} e'$ as desired. We deduce the result on the equivalence closure.

Proposition 1. *Given two causal nets $\sigma, \tau : A$, the following are equivalent:*

1. *there exists a causal net $\upsilon : A$ such that $\sigma \subseteq \upsilon$ and $\tau \subseteq \upsilon$,*
2. *the set $\sigma \vee \tau = \{x \cup y \mid x \in \sigma, y \in \tau, x \cup y \in \mathscr{C}(A)\}$ is a causal net on A,*
3. *$\mathsf{link}(\sigma) = \mathsf{link}(\tau)$.*

In this case we write $\sigma \uparrow \tau$ and $\sigma \vee \tau$ is the least upper bound of σ and τ for $\subseteq$.

Proof. We have (2) $\Rightarrow$ (1) and (1) $\Rightarrow$ (3) by Lemma 17. We focus on (3) $\Rightarrow$ (2) by showing that $\sigma \vee \tau$ is a causal net. By Lemma 17 we know that $\max\sigma = \max\tau$.

Confusion-freeness. Straightforward.

Determinism. Determinism follows easily from the following lemma: if $x \in \sigma$ and $y \in \tau$ are such that there are no minimal negative conflict in $x \cup y$, then $x \cup y \in \mathscr{C}(A)$. By using totality, we can extend x to a maximal x' which has no negative conflict with y. Similarly, we can extend y to a maximal configuration y' of τ which has no negative minimal conflict with x'. Since $\max\sigma = \max\tau$, we must have $x', y' \in \max\sigma$ without negative conflict, hence they must be equal hence $x \cup y \in \mathscr{C}(A)$.

Totality. (1) is trivial. Assume $z \in \max(\sigma \vee \tau)$ with a maximal decomposition $x \cup y$. If $x \overset{a}{\longrightarrow\!\!\subset}$ in σ, then if a is positive, we have $x \cup y \overset{a}{\longrightarrow\!\!\subset}$, absurd. If a is negative, then either there are no conflict with y, and then $x \cup y$ extends as well, or there is a minimal conflict with $a' \in y$, and then $x \overset{a'}{\longrightarrow\!\!\subset}$ by totality, which means that x was not picked maximal. As a result, we must have $x \in \max\sigma$ and similarly $y \in \max\tau$. Since x and y are compatible in $\mathscr{C}(A)$, by totality we must have $x = y$, hence z is maximal in $\mathscr{C}(A)$.

Acyclicity. Let $x \cup y \in \sigma \vee \tau$. Consider $\mathfrak{a} \leftrightsquigarrow_{x \cup y, \sigma \vee \tau} \mathfrak{a}'$ with positive meet. First let us notice that $\mathfrak{a} \leftrightsquigarrow_{x, \sigma \vee \tau} \mathfrak{a}'$ implies that $\mathfrak{a} \leftrightsquigarrow_{x,\sigma} \mathfrak{a}'$ by Lemma 18. This entails that $\mathfrak{a} \wedge \mathfrak{a}' \notin x$. Similarly, $\mathfrak{a} \wedge \mathfrak{a}' \notin y$, and we can conclude.

Well-linking. Follow from $\max(\sigma) = \max(\tau) = \max(\sigma \vee \tau)$ and $\ell_x^\sigma = \ell_x^\tau$ for all $x \in \max(\sigma)$ as we can define $\ell_x^{\sigma \vee \tau} = \ell_x^\sigma$.

Theorem 4. *For cut-free proofs P, Q we have $P \equiv Q$ iff $[\![P]\!] = [\![Q]\!]$.*

Proof. ($\Rightarrow$) To show this, we show that if $P \equiv Q$, then $Tr(P) \uparrow Tr(Q)$ using the characterisation (3) of Proposition 1. It is easy to show that if $P \equiv Q$, then $\max(Tr(P)) = \max(Tr(Q))$ and the linking coincide, by induction on the proof that $P \equiv Q$.

($\Leftarrow$). Assume that $[\![P]\!] = [\![Q]\!]$. We proceed by induction on Δ, by case distinction on the unique minimal elements of $Tr(P)$ and $Tr(Q)$. Most cases are similar, we show the interesting cases:

- $\&/\&$: assume $P = u.\mathsf{case}\,(P_l, P_r)$ and $Q = v.\mathsf{case}\,(Q_l, Q_r)$. If $u = v$, we can directly conclude by induction. Otherwise, we know that $v(\mathtt{inl}^-) \in \min([\![P_l]\!]) \cap \min([\![P_r]\!])$, hence by strong sequentialisation of $[\![P_l]\!]$ and $[\![P_r]\!]$, we get that there exists $P_l^l, P_l^r, P_r^l, P_r^r$ such that:

$$[\![P_l]\!] = [\![v.\mathsf{case}\,(P_l^l, P_l^r)]\!] \qquad [\![P_r]\!] = [\![v.\mathsf{case}\,(P_r^l, P_r^r)]\!]$$

  which entails by induction

$$P_l \equiv v.\mathsf{case}\,(P_l^l, P_l^r) \qquad P_r \equiv v.\mathsf{case}\,(P_r^l, P_r^r).$$

  By using the $\&/\&$ permutation rule, we get:

$$P \equiv v.\mathsf{case}\,(u.\mathsf{case}\,(P_l^l, P_r^l), u.\mathsf{case}\,(P_l^r, P_r^r))$$

  and we can then conclude by induction since $[\![Q_l]\!] = [\![u.\mathsf{case}\,(P_l^l, P_r^l)]\!]$ and similarly for $[\![Q_r]\!]$.
- $\parr/\otimes$: assume $P = u(u').\,P_0$ and $Q = v[v'].\,(Q_0 \mid Q_1)$. We know that $[\![P_0]\!]$ must have a minimal element in $[\![T_v]\!]$, hence by induction we get that $P_0 \equiv v[v'].\,(P_0' \mid P_1')$. Then, since the tensor on v is minimal in $[\![P]\!]$, this means that the par on u communicates with at most one side of the tensor on v. This means that u, u' are either both in P_0' or both in P_1' – assume the former. Then we can conclude by induction as $P \equiv v[v'].\,(u(u').\,P_0' \mid P_1')$.
- Other cases are similar, using the acyclicity for the commutation with tensors.

D.2 The category of causal invariants

Lemma 5. *If $\sigma : A^\perp \parallel B$ and $\tau : B^\perp \parallel C$ are causal nets, then so is $\tau \odot \sigma$.*

Proof. We already know that $\tau \odot \sigma$ is acyclic.

Totality. The second property is trivial to check.

Let $x \in \tau \odot \sigma$ maximal in $\tau \odot \sigma$ and write $y \in \tau \circledast \sigma$ a maximal witness. It is clear that $y\langle\sigma\rangle$ cannot be extended by elements of A and $y\langle\tau\rangle$ by elements of C.

Consider $X = \{\mathfrak{b} \mid x\langle\sigma\rangle \overset{b}{-\!\!\subset} \vee x\langle\tau\rangle \overset{b}{-\!\!\subset}, b \in \mathfrak{b}\}$. We define $\mathfrak{b} <_\sigma \mathfrak{b}'$ when there exists a prime configuration in σ of $b' \in \mathfrak{b}'$ compatible with $x\langle\sigma\rangle$, which intersects $\mathfrak{b}$. We also define $<_\tau$. By Lemma 13, we know that $(<_\sigma \cup <_\tau)$ does not have any cycle. As a result, consider a cell $\mathfrak{b}$ minimal for both $<_\sigma$ and $<_\tau$. Because B is race-free, we can assume it is eg. positive. Then $x\langle\sigma\rangle \overset{b}{-\!\!\subset}$ and $x\langle\tau\rangle \overset{b'}{-\!\!\subset}$ with $b, b' \in \mathfrak{b}$. By condition (2) of totality, we know we can choose $b = b'$ since b' is negative for τ. We then contradict the maximality of y.

Well-linking. We show that $\tau \odot \sigma$ is well-linking. Let $x \in \tau \odot \sigma$ maximal and y a maximal witness. Consider ℓ_σ and ℓ_τ the linking for y_τ and y_σ. Consider $c^+ \in \max(x)$ and a prime configuration z_c in $\tau \circledast \sigma$. It is clear that $z_c \setminus \{\ell_\tau^{-1}(c)\}$ is a prime configuration of $\ell_\tau(c)$. If $\ell_\tau^{-1}(c) \in A^\perp \parallel C$, we let $\ell(\ell_\tau^{-1}(c)) = c$. Otherwise, it is in B, and we continue the process by removing ${\ell_\sigma}^{-1}(\ell_\tau^{-1}(c))$, and so on, until we stumble upon $e \in A^\perp \parallel C$ and we let $\ell(e) = c$. By construction, ℓ_x satisfies the required hypothesis.

Lemma 6. *Rooted games and causal invariants form a category* ***CInv****, where the composition of $\sigma : A^\perp \parallel B$ and $\tau : B^\perp \parallel C$ is $(\tau \odot \sigma)^\uparrow$ and the identity on A is $\text{\oe}_A^\uparrow$.*

Proof. Associativity. For composable σ, τ, υ we have:

$$(\upsilon \odot (\tau \odot \sigma)^\uparrow)^\uparrow = (\upsilon \odot \tau \odot \sigma)^\uparrow = ((\upsilon \odot \tau)^\uparrow \odot \sigma)^\uparrow$$

by Lemma 4 and associativity of composition of causal structs.

Identity. Given $\sigma : A$, we have $\sigma \subseteq \sigma \odot \text{\oe}_A$ hence $\sigma = \sigma^\uparrow = (\sigma \odot \text{\oe}_A)^\uparrow$ as desired.

Lemma 19. *$\sigma \otimes \tau$ is a maximal causal net.*

Proof. The main difficulty is maximality and acyclicity.
Acyclicity. First, let us notice that if $\mathfrak{c} \leftrightsquigarrow_{x,\sigma\otimes\tau} \mathfrak{c}'$ with positive meet, then either $\mathfrak{c} \leftrightsquigarrow_{x,\sigma} \mathfrak{c}'$, $\mathfrak{c} \leftrightsquigarrow_{x,\tau} \mathfrak{c}'$, or x does not contain $\mathtt{send}^-$ and $\mathfrak{c} \wedge \mathfrak{c}' = \mathtt{send}^+$. In the first two cases, we can directly apply the acyclicity of σ or τ. The third case is absurd, since x connects B and D as evidenced by the communication path yet $\mathtt{send}^- \notin x$ – absurd.

Maximality. Assume that there exists $\upsilon \in \mathbf{CInv}(A \otimes C, B \otimes D)$ with $\sigma \otimes \tau \subseteq \upsilon$. Define υ_1 to be the set of $x \cap A^\perp \parallel B$ for $x \in \upsilon$, and similarly υ_2. Clearly υ_1 and υ_2 are causal nets. Then we get $\sigma = \upsilon_1$ and $\tau = \upsilon_2$. From there it is immediate to see that $\upsilon = \sigma \otimes \tau$ as desired.

Lemma 7. *The tensor product defines a symmetric monoidal structure on* ***CInv****.*

Proof. Functoriality and coherence diagrams are straightforwad to check using Lemma 4 and Proposition 1.

Lemma 8. *We have a bijection $\parr_{B,C}$ between causal invariants on $A \parallel B \parallel C$ and on $A \parallel (B \parr C)$. As a result, there is an adjunction $A \otimes \_ \dashv A \multimap \_$.*

Proof. The existence of the bijection is straightforward: the bijection just adds or remove the negative move corresponding to $\parr$. Since it is negative, it preserves acyclicity.

For the adjunction, there is a natural bijection:

$$\begin{aligned} &\sigma : (A \otimes B)^\perp \parallel C = (A^\perp \parr B^\perp) \parallel C \\ &\parr^{-1}_{A^\perp,B^\perp}(\sigma) : A^\perp \parallel B^\perp \parallel C \\ &\parr_{B^\perp,C}(\parr^{-1}_{A^\perp,B^\perp}(\sigma)) : A^\perp \parallel (B^\perp \parr C) \end{aligned}$$

It is easy to see that this bijection induces the desired adjunction.

Lemma 20. *$\langle \sigma, \tau \rangle$ is a maximal causal net on $A^\perp \parallel (B \& C)$.*

Proof. Again, we only check maximality and acyclicity.

Acyclicity. Let $x \in \langle \sigma, \tau \rangle$. Assume that we have $\mathfrak{a} \leftrightsquigarrow_{x,\langle \sigma,\tau \rangle} \mathfrak{a}'$ with positive meet. If $x \in (\mathtt{inl}^- \cdot \sigma)$ or $x \in \mathtt{inr}^{\cdot} \tau$, then we conclude directly by acyclicity of σ and τ. Otherwise, if $x \in \sigma \cap_A \tau$, we know that $\leftrightsquigarrow_{x,\sigma} = \leftrightsquigarrow_{x,\tau}$ hence we also conclude by acyclicity of σ and τ.

Maximality. Consider $\langle \sigma, \tau \rangle \subseteq \upsilon$. From υ we can define a causal net $\upsilon_1 = (\upsilon \cap \mathscr{C}(A^\perp)) \cup \{x \setminus \{\mathtt{L}^-\} \in \upsilon \mid \mathtt{L}^- \in x\}$. This gives a maximal proof on $A^\perp \parallel B$ compatible with σ, hence $\upsilon_1 = \sigma$. Similarly, we can define υ_2 on $A^\perp \parallel C$ and show that $\upsilon_2 = \tau$. Now consider $x \in \upsilon$. If $\mathtt{inl}^- \in x$ or $\mathtt{inr}^- \in x$ we conclude directly by the previous point. Otherwise, we know that $x \in \sigma \cap \tau$. If $\mathfrak{a} \leftrightsquigarrow_{x,\sigma} \mathfrak{a}'$ with positive meet, then by acyclicity $\mathfrak{a} \wedge \mathfrak{a}' \notin x$, hence $\mathfrak{a} \leftrightsquigarrow_{x,\tau} \mathfrak{a} \wedge \mathfrak{a}' \leftrightsquigarrow_{x,\tau} \mathfrak{a}'$ as desired.

Theorem 5. ***CInv*** *has products. As it is also $*$-autonomous, it is a model of MALL.*

Proof. It amounts to checking that $\langle \sigma, \tau \rangle$ satisfies the universal property of products – which is straightforward.