

ON THE MEANING OF LOGICAL COMPLETENESS

MICHELE BASALDELLA^a AND KAZUSHIGE TERUI^b

^{a,b} Research Institute for Mathematical Sciences, Kyoto University, Kitashirakawa Oiwakecho, Sakyo-ku, Kyoto 606-8502, Japan.

e-mail address: {mbasalde,terui}@kurims.kyoto-u.ac.jp

ABSTRACT. Gödel’s completeness theorem is concerned with provability, while Girard’s theorem in ludics (as well as full completeness theorems in game semantics) are concerned with proofs. Our purpose is to look for a connection between these two disciplines. Following a previous work [3], we consider an extension of the original ludics with contraction and universal nondeterminism, which play dual roles, in order to capture a polarized fragment of linear logic and thus a constructive variant of classical propositional logic.

We then prove a completeness theorem for proofs in this extended setting: for any behaviour (formula) \mathbf{A} and any design (proof attempt) P , either P is a proof of \mathbf{A} or there is a model M of \mathbf{A}^\perp which defeats P . Compared with proofs of full completeness in game semantics, ours exhibits a striking similarity with proofs of Gödel’s completeness, in that it explicitly constructs a countermodel essentially using König’s lemma, proceeds by induction on formulas, and implies an analogue of Löwenheim-Skolem theorem.

INTRODUCTION

Gödel’s completeness theorem (for first-order classical logic) is one of the most important theorems in logic. It is concerned with a duality (in a naive sense) between proofs and models: For every proposition \mathbf{A} ,

$$\text{either } \exists P(P \vdash \mathbf{A}) \quad \text{or} \quad \exists M(M \models \neg \mathbf{A}).$$

Here P ranges over the set of proofs, M over the class of models, and $P \vdash \mathbf{A}$ reads “ P is a proof of \mathbf{A} .” One can imagine a debate on a general proposition \mathbf{A} , where Player tries to justify \mathbf{A} by giving a proof and Opponent tries to refute it by giving a countermodel. The completeness theorem states that exactly one of them wins. Actually, the theorem gives us far more insights than stated.

Finite proofs vs infinite models: A very crucial point is that proofs are always finite, while models can be of arbitrary cardinality. Completeness thus implies compactness and Löwenheim-Skolem theorems, leading to constructions of various nonstandard models.

1998 ACM Subject Classification: F.3.2, F.4.1.

Key words and phrases: Ludics, Linear Logic, Completeness.

^a Supported by JSPS Postdoctoral Fellowship Program for Foreign Researchers grant 2008803.

^{a,b} This work was supported by JSPS KAKENHI 21700041.

Nondeterministic principles: *Any* proof of Gödel’s completeness theorem relies on a strong nondeterministic principle such as König’s or Zorn’s lemma, in contrast to the trivial completeness theorem with respect to the class of boolean algebras.

Matching of two inductions: *Provability* is defined by induction on proofs, while *truth* by induction on formulas. The two inductions are somehow ascribed to the essence of syntax and semantics, respectively, and the completeness theorem states that they do match.

Unlike the real debate, however, there is no interaction between proofs and models in Gödel’s theorem. A more interactive account of completeness is given by Girard’s *ludics* ([19, 21]; see [16, 8] for good expositions). Ludics is a variant of game semantics, which has the following prominent features.

Monism: Proofs and models are not distinguished by their ontological status, but by their structural properties. The common objects are called *designs*.

Existentialism: *Behaviours* (semantic types) are built from designs, in contrast to the ordinary game semantics (*e.g.*, Hyland-Ong [22]) where one begins with the definition of arenas (types) and then proceeds to strategies (proofs).

Normalization as interaction: Designs (hence proofs and models) interact together via normalization. It induces an *orthogonality relation* between designs in such a way that $P \perp M$ holds if the normalization of P applied to M converges. A behaviour \mathbf{A} is defined to be a set of designs which is equivalent to its biorthogonal ($\mathbf{A} = \mathbf{A}^{\perp\perp}$).

In this setting, Girard shows a completeness theorem for proofs [21], which roughly claims that any “winning” design in a behaviour is a proof of it. In view of the interactive definition of behaviour, it can be rephrased as follows: For every (logical) behaviour \mathbf{A} and every (proof-like) design P ,

$$\text{either } P \vdash \mathbf{A} \text{ or } \exists M (M \models \mathbf{A}^\perp \text{ and } M \text{ defeats } P).$$

Here, “ $M \models \mathbf{A}^\perp$ ” means $M \in \mathbf{A}^\perp$, and “ M defeats P ” means $P \not\perp M$. Hence the right disjunct is equivalent to $P \notin \mathbf{A}^{\perp\perp} = \mathbf{A}$. Namely, $P \in \mathbf{A}$ if and only if $P \vdash \mathbf{A}$, that is a typical full completeness statement. Notice that $M \models \mathbf{A}^\perp$ no more entails absolute unprovability of \mathbf{A} (it is rather relativized to each P), and there is a real interaction between proofs and models.

Actually, Girard’s original ludics is so limited that it corresponds to a polarized fragment of multiplicative additive linear logic, which is too weak to be a stand-alone logical system. As a consequence, one does not really observe an opposition between finite proofs and infinite models, since one can always assume that the countermodel M is finite (related to the finite model property for **MALL** [23]). Indeed, proving the above completeness is easy once *internal completeness* (a form of completeness which does not refer to any proof system [21]) for each logical connective has been established.

In this paper, we employ a term syntax for designs introduced in [30], and extend Girard’s ludics with duplication (contraction) and its dual: universal nondeterminism (see [3] and references therein). Although our term approach disregards some interesting locativity-related phenomena (*e.g.*, normalization as merging of orders and different sorts of tensors [21]), our calculus is easier to manipulate and closer to the tradition of λ , $\lambda\mu$, $\bar{\lambda}\mu\tilde{\mu}$, π -calculi and other more recent syntaxes for focalized classical logic (*e.g.*, [11]). Our resulting framework is as strong as a polarized fragment of linear logic with exponentials ([8]; see also [25]), which is in turn as strong as a constructive version of classical propositional logic.

We then prove the completeness theorem above in this extended setting. Here, universal nondeterminism is needed on the model side to well interact with duplicative designs on the proof side. This is comparable to the need of “noninnocent” (and sometimes even nondeterministic) Opponents to have full completeness with respect to deterministic, but nonlinear Player’s strategies. Unlike before, we cannot anymore assume the finiteness of models, since they are not sufficient to refute infinite proof attempts. As a result, our proof is nontrivial, even after the internal completeness theorem has been proved. Indeed, our proof exhibits a striking similarity with Schütte’s proof of Gödel’s completeness theorem [29]. Given a (proof-like) design P which is not a proof of \mathbf{A} , we explicitly construct a countermodel M in \mathbf{A}^\perp which defeats P , essentially using König’s lemma. Soundness is proved by induction on proofs, while completeness is by induction on types. Thus our theorem gives matching of two inductions. Finally, it implies an analogue of Löwenheim-Skolem theorem, (and also the finite model property for the linear fragment), which well illustrates the opposition between finite proofs and infinite models with arbitrary cardinality.

In game semantics, one finds a number of similar full completeness results. However, the connection with Gödel’s completeness seems less conspicuous than ours. Typically, innocent strategies in Hyland-Ong games most naturally correspond to *Böhm trees*, which can be *infinite* (cf. [9]). Thus, in contrast to our result, one has to impose finiteness/compactness on strategies in an external way, in order to have a correspondence with *finite* λ -terms. Although this is also the case in [3], we show that such a finiteness assumption is not needed in ludics: infinitary proof attempts are always defeated by infinitary models.

The paper is organized as follows. In Section 1 we describe the syntax of (untyped) designs; in Section 2 we move to a typed setting and introduce behaviours (semantic types). In Section 3 we introduce our proof system and prove completeness for proofs. Finally, Section 4 concludes the paper.

1. DESIGNS

1.1. Syntax. In this paper, we employ a process calculus notation for designs, inspired by the close relationship between ludics and linear π -calculus [17]. Precisely, we extend the syntax introduced by the second author [30] adding a (universal) nondeterministic choice operator \bigwedge .

Although [30] mainly deals with linear designs, its syntax is designed to deal with nonlinear ones without any difficulty. However, in order to obtain *completeness*, we also need to incorporate the dual of nonlinearity, that is *universal nondeterminism* [3]. It is reminiscent of differential linear logic [14], which has nondeterministic sum as the dual of contraction; the duality is essential for the separation property [27] (see also [12] for separation of Böhm trees). A similar situation also arises in Hyland-Ong game semantics [22], where *nonlinear* strategies for Player may contain a play in which Opponent behaves *noninnocently*; Opponent’s noninnocence is again essential for full completeness.

Designs are built over a given *signature* $\mathcal{A} = (A, \text{ar})$, where A is a set of *names* a, b, c, \dots and $\text{ar} : A \rightarrow \mathbb{N}$ is a function which assigns to each name a its *arity* $\text{ar}(a)$. Let \mathcal{V} be a countable set of variables $\mathcal{V} = \{x, y, z, \dots\}$.

Over a fixed signature \mathcal{A} , a *positive action* is \bar{a} with $a \in A$, and a *negative action* is $a(x_1, \dots, x_n)$ where variables x_1, \dots, x_n are distinct and $\text{ar}(a) = n$. We often abbreviate a sequence of variables x_1, \dots, x_n by \vec{x} . In the sequel, we always assume that an expression

of the form $a(\vec{x})$ stands for a negative action, *i.e.*, $\text{ar}(a) = n$ and \vec{x} is a sequence consisting of n distinct variables. If a is a nullary name we simply write a for the negative action on name a .

Definition 1.1 (Designs). For a fixed signature \mathcal{A} , the class of **positive designs** P, Q, \dots , that of **predesigns** S, T, \dots , and that of **negative designs** N, M, \dots are *coinductively* defined as follows:

$$\begin{array}{lll} P & ::= & \Omega \quad (\textit{divergence}), \\ & | & \bigwedge \{S_i : i \in I\} \quad (\textit{conjunction}), \\ S & ::= & N_0 | \bar{a} \langle N_1, \dots, N_n \rangle \quad (\textit{predesign}), \\ N & ::= & x \quad (\textit{variable}), \\ & | & \sum a(\vec{x}).P_a \quad (\textit{abstraction}), \end{array}$$

where:

- $\text{ar}(a) = n$;
- $\vec{x} = x_1, \dots, x_n$ and the formal sum $\sum a(\vec{x}).P_a$ has $|A|$ -many components $\{a(\vec{x}).P_a\}_{a \in A}$;
- $\bigwedge \{S_i : i \in I\}$ is built from a set $\{S_i : i \in I\}$ of predesigns with I an arbitrary index set.

We denote arbitrary designs by D, E, \dots . The set of **designs**, consisting of all positive, negative and predesigns, is denoted by \mathcal{D} . Any subterm E of D is equivalently called a *subdesign* of D .

Notice that designs are *coinductively* defined objects. In particular, infinitary designs are included in our syntax, just as in the original ludics [21]. It is strictly necessary, since we want to express both proof attempts and countermodels as designs, both of which tend to be infinite.

Informally, designs may be regarded as infinitary λ -terms with *named* applications, *named* and *superimposed* abstractions and a *universal nondeterministic* choice operator \bigwedge .

More specifically, a predesign $N_0 | \bar{a} \langle N_1, \dots, N_n \rangle$ can be thought of as iterated application $N_0 N_1 \cdots N_n = (\cdots ((N_0 N_1) N_2) \cdots N_n)$ of an n -ary name $a \in A$. In the sequel, we may abbreviate $N_0 | \bar{a} \langle N_1, \dots, N_n \rangle$ by $N_0 | \bar{a} \langle \vec{N} \rangle$. If a is a nullary name we simply write $N_0 | \bar{a}$.

On the other hand, a negative design of the form $a(\vec{x}).P_a$ can be thought of as iterated abstraction $\lambda \vec{x}.P_a = \lambda x_1.(\lambda x_2.(\cdots (\lambda x_n.P_a) \cdots))$ of an n -ary name $a \in A$. A family $\{a(\vec{x}).P_a\}_{a \in A}$ of abstractions indexed by A is then superimposed to form a negative design $\sum a(\vec{x}).P_a$. Since $\sum a(\vec{x}).P_a$ is built from a family indexed by A , there cannot be any overlapping of name in the sum. Each $a(\vec{x}).P_a$ is called an (additive) **component**.

A predesign is called a **cut** if it is of the form $(\sum a(\vec{x}).P_a) | \bar{b} \langle N_1, \dots, N_n \rangle$. Otherwise, it is of the form $x | \bar{a} \langle N_1, \dots, N_n \rangle$ and called a **head normal form**.

As we shall see in detail in Subsection 1.2, cuts have substantial computational significance in our setting: in fact a cut $(\sum a(\vec{x}).P_a) | \bar{b} \langle \vec{N} \rangle$ can be *reduced* to another design $P_b[\vec{N}/\vec{y}]$. Namely, when the application is of name b , one picks up the component $b(\vec{y}).P_b$ from the family $\{a(\vec{x}).P_a\}_{a \in A}$. Notice that the arities of \vec{y} and \vec{N} always agree. Then, one applies a simultaneous “ β -reduction” $(\lambda \vec{y}.P_b) \vec{N} \longrightarrow P_b[\vec{N}/\vec{y}]$.

The *head variable* x in an head normal form $x | \bar{a} \langle N_1, \dots, N_n \rangle$ plays the same role as a pointer in a strategy does in Hyland-Ong games and an address (or locus) in Girard’s ludics. On the other hand, a variable x occurring in a bracket as in $N_0 | \bar{a} \langle N_1, \dots, N_{i-1}, x, N_{i+1}, \dots, N_n \rangle$ does not correspond to a pointer nor address. Rather, it corresponds to an identity axiom (initial sequent) in sequent calculus, and for this reason is called an **identity**. If a negative design N simply consists of a variable x , then N is itself an identity.

The positive design Ω denotes *divergence* (or *partiality*) of the computation, in the sense we will make more precise in the next subsection. We also use Ω to encode partial sums. Given a set $\alpha = \{a(\vec{x}), b(\vec{y}), c(\vec{z}), \dots\}$ of negative actions with distinct names $\{a, b, c, \dots\} \subseteq A$, we write $\sum_{\alpha} a(\vec{x}).P_a$ to denote the negative design $\sum a(\vec{x}).R_a$, where $R_a = P_a$ if $a(\vec{x}) \in \alpha$, and $R_a = \Omega$ otherwise. We also use an informal notation $a(\vec{x}).P_a + b(\vec{y}).P_b + c(\vec{z}).P_c + \dots$ to denote $\sum_{\alpha} a(\vec{x}).P_a$.

So far, the syntax we are describing is essentially the same as the one introduced in [30]. A novelty of this paper is the nondeterministic conjunction operator \bigwedge , which allows us to build a positive design $\bigwedge\{S_i : i \in I\}$ from a set $\{S_i : i \in I\}$ of predesigns with I an arbitrary index set. Each S_i is called a **conjunct** of P . We write \boxtimes (**daimon**) for the empty conjunction $\bigwedge \emptyset$. This design plays an essential role in ludics, since it is used to define the concept of orthogonality. Although \boxtimes is usually given as a primitive (see *e.g.*, [21, 8, 30]), we have found it convenient and natural to identify (or rather encode) \boxtimes with the empty conjunction. As we shall see, its computational meaning exactly corresponds to the usual one: \boxtimes marks the *termination* of a computation. Put in another way, our nondeterministic conjunction can be seen as a generalization of the termination mark.

A design D may contain free and bound variables. An occurrence of subterm $a(\vec{x}).P_a$ *binds* the free variables \vec{x} in P_a . Variables which are not under the scope of the binder $a(\vec{x})$ are *free*. We denote by $\text{fv}(D)$ the set of free variables occurring in D . As in λ -calculus, we would like to identify two designs which are α -equivalent *i.e.*, up to renaming of bound variables. But it is more subtle than usual, since we also would like to identify, *e.g.*, $\bigwedge\{S, T\}$ with $\bigwedge\{S\}$ whenever S and T are α -equivalent. To enforce these requirements simultaneously and hereditarily, we define an equivalence relation by coinduction.

By *renaming* we mean a function $\rho : \mathcal{V} \rightarrow \mathcal{V}$. We write *id* for the identity renaming, and $\rho[z/x]$ for the renaming that agrees with ρ except that $\rho[z/x](x) = z$. The set of renamings is denoted by \mathcal{RN} .

Definition 1.2 (Design equivalence). A binary relation $\mathcal{R} \subseteq (\mathcal{D} \times \mathcal{RN})^2$ is called a **design equivalence** if for any D, E, ρ, τ such that $(D, \rho) \mathcal{R} (E, \tau)$, one of the following holds:

- (1) $D = \Omega = E$;
- (2) $D = \bigwedge\{S_i : i \in I\}$, $E = \bigwedge\{T_j : j \in J\}$ and we have:
 - (i) for any $i \in I$ there is $j \in J$ such that $(S_i, \rho) \mathcal{R} (T_j, \tau)$,
 - (ii) for any $j \in J$ there is $i \in I$ such that $(S_i, \rho) \mathcal{R} (T_j, \tau)$;
- (3) $D = N_0|\bar{a}\langle N_1, \dots, N_n \rangle$, $E = M_0|\bar{a}\langle M_1, \dots, M_n \rangle$ and $(N_k, \rho) \mathcal{R} (M_k, \tau)$ for every $0 \leq k \leq n$;
- (4) $D = x$, $E = y$ and $\rho(x) = \tau(y)$;
- (5) $D = \sum a(\vec{x}_a).P_a$, $E = \sum a(\vec{y}_a).Q_a$ and $(P_a, \rho[\vec{z}_a/\vec{x}_a]) \mathcal{R} (Q_a, \rho[\vec{z}_a/\vec{y}_a])$ for every $a \in A$ and some vector \vec{z}_a of fresh variables.

We say that two designs D and E are *equivalent* if there is a design equivalence \mathcal{R} such that $(D, id) \mathcal{R} (E, id)$. See [30] for further details.

Henceforth we *always* identify two designs D and E , and write $D = E$ by abuse of notation, if they are equivalent in the above sense. The following lemma is a straightforward extension of Lemma 2.6 of [30]. It makes it easier to prove equivalence of two designs (just as the “bisimulation up-to” technique in concurrency theory makes it easier to prove bisimilarity of two processes).

Lemma 1.3. *Let \mathcal{R} be a binary relation on designs such that if $D \mathcal{R} E$ then one of the following holds:*

- (1) $D = \Omega = E$;
- (2) $D = \bigwedge\{S_i : i \in I\}$, $E = \bigwedge\{T_j : j \in J\}$, and we have:
 - (i) for any $i \in I$ there is $j \in J$ such that $S_i \mathcal{R} T_j$,
 - (ii) for any $j \in J$ there is $i \in I$ such that $S_i \mathcal{R} T_j$;
- (3) $D = N_0|\bar{a}\langle N_1, \dots, N_n \rangle$, $E = M_0|\bar{a}\langle M_1, \dots, M_n \rangle$ and $N_k \mathcal{R} M_k$ for every $0 \leq k \leq n$;
- (4) $D = x = E$;
- (5) $D = \sum a(\vec{x}).P_a$, $E = \sum a(\vec{x}).Q_a$ and $P_a \mathcal{R} Q_a$ for every $a \in A$.

If $D \mathcal{R} E$, then D and E are equivalent. \square

As a notational convention, a unary conjunction $\bigwedge\{S\}$ is simply written as S . This allows us to treat a predesign as a positive design. We also write:

- $S \in P$ if P is a conjunction and S is a conjunct of P ;
- $P \leq Q$ if either $P = \Omega$, or both P and Q are conjunctions and for all $S \in Q$, $S \in P$.

Thus $P \leq Q$ indicates that P has more conjuncts than Q unless $P = \Omega$. We also extend the conjunction operator to positive designs and abstractions as follows.

Definition 1.4 (\wedge operation).

- (1) As for positive designs, we set

$$\bigwedge\{S_i : i \in I\} \wedge \bigwedge\{S_j : j \in J\} := \bigwedge\{S_k : k \in I \cup J\}, \quad \Omega \wedge P := \Omega.$$

- (2) As for abstractions, *i.e.*, negative designs of the form $\sum a(\vec{x}).P_a$, observe that since we are working up to renaming of bound variables, it is no loss of generality to assume that in any pair $\sum a(\vec{x}).P_a$, $\sum a(\vec{y}).Q_a$, one has $\vec{x} = \vec{y}$ for every $a \in A$. We set:

$$\sum a(\vec{x}).P_a \wedge \sum a(\vec{x}).Q_a := \sum a(\vec{x}).(P_a \wedge Q_a).$$

Observe the following:

- The set of positive designs forms a semilattice with respect to \leq and \wedge .
- $\Omega \leq P \leq \mathbf{X}$ for any positive design P .

The previous definition can be naturally generalized to arbitrary sets as follows:

Definition 1.5 (\bigwedge operation).

- (1) Given a set \mathbf{X} of positive designs, we define the positive design $\bigwedge \mathbf{X}$ as follows:

- If $\mathbf{X} = \emptyset$, we set $\bigwedge \mathbf{X} := \mathbf{X}$.
- If $\Omega \in \mathbf{X}$, we set $\bigwedge \mathbf{X} := \Omega$.
- Otherwise, \mathbf{X} is a nonempty set of conjunctions and we set:

$$\bigwedge \mathbf{X} := \bigwedge\{S : S \in P \text{ for some } P \in \mathbf{X}\}.$$

- (2) Given a set \mathbf{X} of abstractions, we define the abstraction $\bigwedge \mathbf{X}$ as:

$$\bigwedge \mathbf{X} := \sum a(\vec{x}).\bigwedge\{P_a : \sum a(\vec{x}).P_a \in \mathbf{X}\}.$$

In particular, if $\mathbf{X} = \emptyset$ then $\bigwedge \mathbf{X} = \sum a(\vec{x}).\mathbf{X}$.

Notice that $\bigwedge\{D\} = D$, as long as D ranges over positive designs or abstractions.

A design D is said:

- **total**, if $D \neq \Omega$;
- **closed**, if D has no occurrence of free variable;
- **linear** (or **affine**, more precisely), if for any subdesign of the form $N_0|\bar{a}\langle N_1, \dots, N_n \rangle$, the sets $\text{fv}(N_0), \dots, \text{fv}(N_n)$ are pairwise disjoint;

- **deterministic**, if in any occurrence of subdesign $\bigwedge\{S_i : i \in I\}$, I is either empty (*i.e.*, we have \boxtimes) or a singleton (*i.e.*, we have a predesign).
- **cut-free**, if it does not contain a cut as a subdesign;
- **identity-free**, if it does not contain an identity as subdesign.

We remark that the notion of design introduced in [30] exactly corresponds in our terminology to that of deterministic design. Furthermore, considering the specific signature \mathcal{G} given below, we can also express in our setting Girard's original notion of design:

Example 1.6 (Girard's syntax). Let us consider the signature $\mathcal{G} = (\mathcal{P}_f(\mathbb{N}), | \ |)$ where:

- $\mathcal{P}_f(\mathbb{N})$ consists of finite subsets of \mathbb{N} ;
- $| \ | : \mathcal{P}_f(\mathbb{N}) \rightarrow \mathbb{N}$ is the function that maps a finite subset $I \subseteq_f \mathbb{N}$ to its cardinality $|I| \in \mathbb{N}$.

Girard's designs correspond to total, linear, deterministic, cut-free and identity-free designs which have a finite number of free variables over the signature \mathcal{G} . See [30] for more details.

1.2. Normalization. Ludics is an interactive theory. This means that designs, which subsume both proofs and models, interact together via normalization, and types (behaviours) are defined by the induced orthogonality relation (Section 2). Several ways to normalize designs have been considered in the literature: abstract machines [7, 15, 10, 3], abstract merging of orders [21, 18], and terms reduction [10, 30]. Here we extend the last solution [30]. As in untyped λ -calculus, normalization is not necessarily terminating, but in our setting a new difficulty arises through the presence of the operator \bigwedge .

We define the normal forms in two steps, first giving a *nondeterministic* reduction rule which finds head normal forms whenever possible, and then expanding it corecursively. As usual, let $D[\vec{N}/\vec{x}]$ denote the design obtained by the simultaneous and capture-free substitution of negative designs $\vec{N} = N_1, \dots, N_n$ for $\vec{x} = x_1, \dots, x_n$ in D .

Definition 1.7 (Reduction relation \rightarrow). Given positive designs P, Q , we write $P \rightarrow Q$ if $(\sum a(\vec{x}).P_a) | \bar{b}\langle \vec{N} \rangle \in P$ and $Q = P_b[\vec{N}/\vec{x}]$. We denote by \rightarrow^+ the transitive closure, and by \rightarrow^* the reflexive transitive closure of \rightarrow .

Given two binary relations $\mathcal{R}_1, \mathcal{R}_2$ on designs, we write $\mathcal{R}_1 \mathcal{R}_2$ to denote the relation given by their composition *i.e.*,

$$D \mathcal{R}_1 \mathcal{R}_2 F \iff \text{there exists a design } E \text{ such that } D \mathcal{R}_1 E \text{ and } E \mathcal{R}_2 F.$$

For instance, we write $P \rightarrow^* \ni S$ if there exists Q such that $P \rightarrow^* Q$ and $Q \ni S$.

Examples 1.8. We now give examples of reductions and some remarks.

- (1) $a(x).\boxtimes | \bar{a}\langle K \rangle \rightarrow \boxtimes$.
- (2) $b(x).\boxtimes | \bar{a}\langle K \rangle \rightarrow \Omega$ (recall that $b(x).\boxtimes$ stands for $b(x).\boxtimes + a(y).\Omega + c(\vec{z}).\Omega + \dots$ by our conventions on partial sums).
- (3) Let $P = a(x).\boxtimes | \bar{a}\langle K \rangle \wedge b(x).\boxtimes | \bar{a}\langle K \rangle$. We have $P \rightarrow \boxtimes$ and $P \rightarrow \Omega$.
- (4) For $N = a(x).x|\bar{a}\langle x \rangle$, let us consider $P = N | \bar{a}\langle N \rangle$. We then have an infinite reduction sequence

$$P \rightarrow P \rightarrow P \rightarrow \dots$$

since $P = N | \bar{a}\langle N \rangle = (a(x).x|\bar{a}\langle x \rangle) | \bar{a}\langle N \rangle$ and the latter design reduces to $x|\bar{a}\langle x \rangle[N/x] = N | \bar{a}\langle N \rangle = P$.

(5) Let $P = a(y).(y|\bar{b}\langle w \rangle \wedge z|\bar{c}\langle M \rangle) | \bar{a}\langle b(t).Q \rangle$. We have the following reduction:

$$P \multimap (y|\bar{b}\langle w \rangle \wedge z|\bar{c}\langle M \rangle)[b(t).Q/y] = b(t).Q | \bar{b}\langle w \rangle \wedge z|\bar{c}\langle M[b(t).Q/y] \rangle.$$

We therefore have $P \multimap \exists b(t).Q | \bar{b}\langle w \rangle$ and $P \multimap \exists z|\bar{c}\langle M[b(t).Q/y] \rangle$. Since $b(t).Q | \bar{b}\langle w \rangle$ is a cut, we have:

$$b(t).Q | \bar{b}\langle w \rangle \wedge z|\bar{c}\langle M[b(t).Q/y] \rangle \multimap Q[w/t].$$

- (6) The special designs \boxtimes and Ω do not reduce to anything (as we will see, they are normal forms).
- (7) By its definition, our reduction is not “closed under context” *i.e.*, if $P \multimap Q$ and P (resp. Q) occurs as a subdesign of D (resp. E), nothing ensures that $D \multimap E$. For instance a negative design (or an head normal form) having an occurrence of cut as subdesign does not reduce to anything. To expand the reduction “under context” we will use Definition 1.9.

Notice that any *closed* positive design P has one of the following forms: \boxtimes , Ω and $\bigwedge \{S_i : i \in I\}$, where S_i are cuts. The conjunction then reduces to another closed positive design. Hence any sequence of reductions starting from P either terminates with \boxtimes or Ω or it diverges. By *stipulating* that the normal form of P in case of divergence is Ω , we obtain a dichotomy between \boxtimes and Ω : the normal form of a closed positive design is either \boxtimes or Ω .

This leads us to the following definition of normal form:

Definition 1.9 (Normal form). The **normal form function** $\llbracket \cdot \rrbracket : \mathcal{D} \rightarrow \mathcal{D}$ is defined by corecursion as follows:

$$\begin{aligned} \llbracket P \rrbracket &= \Omega && \text{if there is an infinite reduction} \\ &&& \text{sequence or a reduction sequence} \\ &&& \text{ending with } \Omega \text{ starting from } P; \\ \llbracket \sum a(\vec{x}).P_a \rrbracket &= \bigwedge \{x|\bar{a}\langle \llbracket \vec{N} \rrbracket \rangle : P \multimap^* \exists x|\bar{a}\langle \vec{N} \rangle\} && \text{otherwise;} \\ \llbracket x \rrbracket &= x. \end{aligned}$$

We observe that when P is a closed positive design, we have $\llbracket P \rrbracket = \boxtimes$ precisely when *all* reduction sequences from P are finite and terminate with \boxtimes ; thus our nondeterminism is *universal* rather than existential. This, however, does not mean that the set $\{Q : P \multimap^* Q\}$ is finite; even when it is infinite, it may happen that $\llbracket P \rrbracket = \boxtimes$.

The following facts are easily observed:

Lemma 1.10.

- (1) If $P \neq \Omega$, $P \leq Q$ and $Q \multimap R$, then $P \multimap R$.
- (2) $P \multimap Q$ implies $\llbracket P \rrbracket \leq \llbracket Q \rrbracket$. Furthermore, if P is a predesign, then $P \multimap Q$ implies $\llbracket P \rrbracket = \llbracket Q \rrbracket$.
- (3) $\llbracket \bigwedge \mathbf{X} \rrbracket = \bigwedge \{\llbracket P \rrbracket : P \in \mathbf{X}\}$, for any set \mathbf{X} of positive designs. □

Notice that the first statement means that the composed relation $\leq \multimap$ is equivalent to \multimap as far as total designs are concerned.

Example 1.11 (Acceptance of finite trees). In [30], it is illustrated how words and deterministic finite automata are represented by (deterministic) designs in ludics. We may extend the idea to trees and finite tree automata in presence of nondeterminism. Rather

than describing it in full detail, we will only give an example which illustrates the power of nondeterminism to express (topdown) finite tree automata.

We consider the set of *finite trees* labelled with a, b which are at most binary branching. It is defined by the following grammar:

$$t ::= \epsilon \mid a(t_1, t_2) \mid b(t_1, t_2).$$

Here, $a(t_1, t_2)$ represents a tree with the root labelled by a and with two subtrees t_1, t_2 . In particular, $a(\epsilon, \epsilon)$ represents a leaf labelled by a . We simply write a in this case.

Suppose that the signature \mathcal{A} contains a unary name \uparrow , binary names a, b and a nullary name ϵ . We write \downarrow for the positive action $\bar{\uparrow}$. We abbreviate $\uparrow(x).x|\bar{a}\langle\vec{N}\rangle$ by $\uparrow\bar{a}\langle\vec{N}\rangle$, so that we have

$$(\uparrow\bar{a}\langle\vec{N}\rangle) \mid \downarrow\langle a(\vec{x}).P \rangle \multimap (a(\vec{x}).P) \mid \bar{a}\langle\vec{N}\rangle \multimap P[\vec{N}/\vec{x}].$$

Each tree is then represented by a deterministic linear negative design as follows:

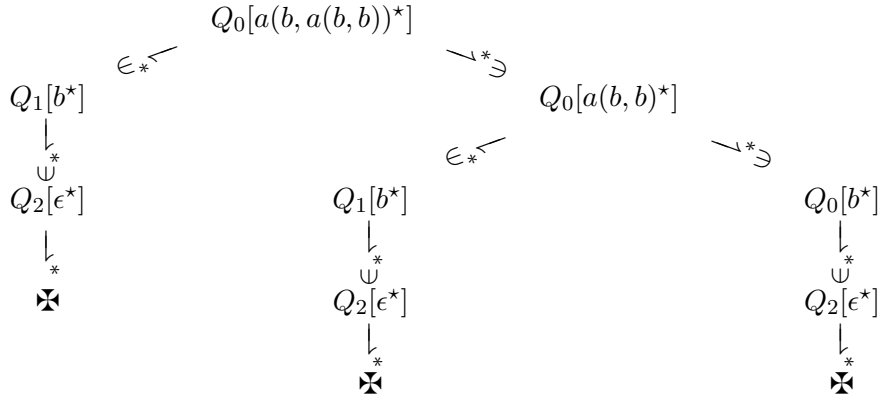
$$\begin{aligned} \epsilon^* &:= \uparrow\bar{\epsilon}, \\ a(t_1, t_2)^* &:= \uparrow\bar{a}\langle t_1^*, t_2^* \rangle, \\ b(t_1, t_2)^* &:= \uparrow\bar{b}\langle t_1^*, t_2^* \rangle. \end{aligned}$$

Now consider the positive design $Q = Q_0[x_0]$ defined by the following equations:

$$\begin{aligned} Q_0[x] &:= x|\downarrow\langle a(x, y).Q_1[x] \wedge Q_0[y] + b(x, y).Q_2[x] \wedge Q_2[y] \rangle, \\ Q_1[x] &:= x|\downarrow\langle b(x, y).Q_2[x] \wedge Q_2[y] \rangle, \\ Q_2[x] &:= x|\downarrow\langle \epsilon.\boxtimes \rangle. \end{aligned}$$

This design Q works as an automata accepting all trees of the form $a(b, a(b, \dots a(b, b) \dots))$.

Indeed, given $a(b, a(b, b))$, it works nondeterministically as follows:



Hence we conclude $\llbracket Q_0[a(b, a(b, b))^*] \rrbracket = \boxtimes$, *i.e.*, Q “accepts” the tree $a(b, a(b, b))$.

1.3. Associativity. In this subsection, we prove one of the fundamental properties of designs which we will need later:

Theorem 1.12 (Associativity). *Let D be a design and N_1, \dots, N_n be negative designs. We have:*

$$\llbracket D[N_1/y_1, \dots, N_n/y_n] \rrbracket = \llbracket \llbracket D \rrbracket [\llbracket N_1 \rrbracket /y_1, \dots, \llbracket N_n \rrbracket /y_n] \rrbracket.$$

□

Associativity corresponds to a weak form of the Church-Rosser property: the normal form is the same even if we do not follow the head reduction strategy. In this paper we are not concerned with the full Church-Rosser property, and leave it as an open question.

The proof consists of several stages and it can be skipped at first reading.

To prove associativity, first notice that a simultaneous substitution $D[N_1/y_1, \dots, N_n/y_n]$ can be turned into a sequential one of the form $D'[N_1/z_1] \cdots [N_n/z_n]$ by renaming y_1, \dots, y_n by fresh variables z_1, \dots, z_n as follows:

$$D[N_1/y_1, \dots, N_n/y_n] = D[z_1/y_1, \dots, z_n/y_n][N_1/z_1] \cdots [N_n/z_n].$$

Moreover, we have:

$$\llbracket D \rrbracket [\llbracket N_1 \rrbracket / y_1, \dots, \llbracket N_n \rrbracket / y_n] = \llbracket D[z_1/y_1, \dots, z_n/y_n] \rrbracket [\llbracket N_1 \rrbracket / z_1] \cdots [\llbracket N_n \rrbracket / z_n].$$

This allows us to work with sequential substitutions rather than simultaneous ones.

We define a binary relation \gg on designs by:

- $D \gg E$ if $D = D_0[N_1/y_1] \cdots [N_n/y_n]$ and $E = \llbracket D_0 \rrbracket [\llbracket N_1 \rrbracket / y_1] \cdots [\llbracket N_n \rrbracket / y_n]$ for some D_0, N_1, \dots, N_n such that $y_i \notin \text{fv}(N_j)$ for $1 \leq i \leq j \leq n$.

Lemma 1.13. *Suppose that $P = P_0[N_1/y_1] \cdots [N_n/y_n]$ and $Q = \llbracket P_0 \rrbracket [\llbracket N_1 \rrbracket / y_1] \cdots [\llbracket N_n \rrbracket / y_n]$ so that $P \gg Q$. When $P \rightarrow P'$, two cases can be distinguished.*

- (1) *If $P_0 \rightarrow P_1$ and $P' = P_1[N_1/y_1] \cdots [N_n/y_n]$, then there exists Q' such that $Q \leq Q'$ and $P' \gg Q'$.*
- (2) *Otherwise, there exists Q' such that $Q \rightarrow Q'$ and $P' \gg Q'$.*

Proof.

- (1) By Lemma 1.10 (2), we have $\llbracket P_0 \rrbracket \leq \llbracket P_1 \rrbracket$ that implies $Q \leq \llbracket P_1 \rrbracket [\llbracket N_1 \rrbracket / y_1] \cdots [\llbracket N_n \rrbracket / y_n]$. Hence by letting $Q' = \llbracket P_1 \rrbracket [\llbracket N_1 \rrbracket / y_1] \cdots [\llbracket N_n \rrbracket / y_n]$, we have $Q \leq Q'$ and $P' \gg Q'$.
- (2) If (1) is not the case, a cut must be created by substitution of some N_j for a head variable of P_0 . Hence P_0 must contain a head normal form $y_j | \bar{a} \langle \vec{M} \rangle$ as conjunct for some $1 \leq j \leq n$ and $N_j = \sum a(\vec{x}).R_a$, so that P contains a cut

$$y_j | \bar{a} \langle \vec{M} \rangle [N_1/y_1] \cdots [N_n/y_n] = N_j | \bar{a} \langle \vec{M} \rangle [N_1/y_1] \cdots [N_n/y_n]$$

(the equality due to $y_i \notin \text{fv}(N_j)$ for $1 \leq i \leq j$) and $P \rightarrow R_a[\vec{M}/\vec{x}][N_1/y_1] \cdots [N_n/y_n] = P'$. In this case, $\llbracket P_0 \rrbracket$ contains $y_j | \bar{a} \langle \llbracket \vec{M} \rrbracket \rangle$ so that Q contains

$$y_j | \bar{a} \langle \llbracket \vec{M} \rrbracket \rangle [\llbracket N_1 \rrbracket / y_1] \cdots [\llbracket N_n \rrbracket / y_n] = \llbracket N_j \rrbracket | \bar{a} \langle \llbracket \vec{M} \rrbracket \rangle [\llbracket N_1 \rrbracket / y_1] \cdots [\llbracket N_n \rrbracket / y_n].$$

Since $\llbracket N_j \rrbracket = \sum a(\vec{x}).\llbracket R_a \rrbracket$, we have $Q \rightarrow \llbracket R_a \rrbracket [\llbracket \vec{M} \rrbracket / \vec{x}][\llbracket N_1 \rrbracket / y_1] \cdots [\llbracket N_n \rrbracket / y_n]$. Let Q' be the latter design. Since the simultaneous substitutions $[\vec{M}/\vec{x}]$ and $[\llbracket \vec{M} \rrbracket / \vec{x}]$ can be made sequential, we have $P' \gg Q'$. \square

Lemma 1.14. *If $P \gg Q$ and $Q \rightarrow Q'$, then there exists some P' such that $P' \gg Q'$ and $P \rightarrow^+ P'$.*

Proof. Suppose that $P = P_0[N_1/y_1] \cdots [N_n/y_n]$, $Q = \llbracket P_0 \rrbracket [\llbracket N_1 \rrbracket / y_1] \cdots [\llbracket N_n \rrbracket / y_n]$ and $Q \rightarrow Q'$. Then $\llbracket P_0 \rrbracket$ must contain $y_j | \bar{a} \langle \llbracket \vec{M} \rrbracket \rangle$ for some \vec{M} and $1 \leq j \leq n$. Thus, $P_0 \rightarrow^* \ni y_j | \bar{a} \langle \vec{M} \rangle$. Suppose also $N_j = \sum a(\vec{x}).R_a$ so that $\llbracket N_j \rrbracket = \sum a(\vec{x}).\llbracket R_a \rrbracket$.

Now the situation is as follows: Q contains

$$y_j | \bar{a} \langle \llbracket \vec{M} \rrbracket \rangle [\llbracket N_1 \rrbracket / y_1] \cdots [\llbracket N_n \rrbracket / y_n] = \llbracket N_j \rrbracket | \bar{a} \langle \llbracket \vec{M} \rrbracket \rangle [\llbracket N_1 \rrbracket / y_1] \cdots [\llbracket N_n \rrbracket / y_n],$$

so we have

$$Q \rightarrow [R_a][[\vec{M}]/\vec{x}][[N_1]/y_1] \cdots [[N_n]/y_n] = Q'$$

On the other hand,

$$\begin{aligned} P &\rightarrow^* \ni y_j |\bar{a}\langle \vec{M} \rangle [N_1/y_1] \cdots [N_n/y_n] \\ &= N_j |\bar{a}\langle \vec{M} \rangle [N_1/y_1] \cdots [N_n/y_n] \\ &\rightarrow R_a [\vec{M}/\vec{x}_a] [N_1/y_1] \cdots [N_n/y_n] = P', \end{aligned}$$

which implies $P \rightarrow^+ P'$. Since the simultaneous substitutions $[\vec{M}/\vec{x}]$ and $[[\vec{M}]/\vec{x}]$ can be made sequential, we have $P' \gg Q'$. \square

Lemma 1.15. *Suppose that $P \gg Q$. Then $\llbracket P \rrbracket = \Omega$ if and only if $\llbracket Q \rrbracket = \Omega$.*

Proof.

- For the ‘if’ direction, we distinguish two cases.
 - If there is an infinite reduction sequence from Q , then there is also an infinite sequence from P by Lemma 1.14.
 - If $Q \rightarrow^* \Omega$, then there is P' such that $P \rightarrow^* P'$ and $P' \gg \Omega$. Namely, P' can be written as $P_0[N_1/y_1] \cdots [N_n/y_n]$ and $\Omega = \llbracket P_0 \rrbracket [[N_1]/y_1] \cdots [[N_n]/y_n]$. The latter means that $\llbracket P_0 \rrbracket = \Omega$, which implies $\llbracket P' \rrbracket = \Omega$. From this and $P \rightarrow^* P'$, we conclude $\llbracket P \rrbracket = \Omega$.
- For the ‘only-if’ direction, if $P \rightarrow^* \Omega$, we easily obtain $\llbracket Q \rrbracket = \Omega$. Otherwise, there is an infinite reduction sequence $P = P^0 \rightarrow P^1 \rightarrow P^2 \rightarrow \cdots$. Suppose that $P = P^0 = P_0[N_1/y_1] \cdots [N_n/y_n]$ and $Q = \llbracket P_0 \rrbracket [[N_1]/y_1] \cdots [[N_n]/y_n]$. Our purpose is to build either a finite reduction sequence $Q \rightarrow^* \Omega$ or an infinite reduction sequence $Q = Q^0 \rightarrow Q^1 \rightarrow Q^2 \rightarrow \cdots$. Two cases arise:
 - The reductions take place inside P_0 and independently of N_1, \dots, N_n . Namely, there is an infinite reduction sequence $P_0 \rightarrow P_1 \rightarrow P_2 \rightarrow \cdots$ such that $P^i = P_i[N_1/y_1] \cdots [N_n/y_n]$ for every $i \geq 0$. Then $\llbracket P_0 \rrbracket = \Omega$, which implies $Q = \Omega$. So we have $\llbracket Q \rrbracket = \Omega$.
 - Otherwise, there is at most a finite sequence $P_0 \rightarrow P_1 \rightarrow \cdots \rightarrow P_m$ such that $P^i = P_i[N_1/y_1] \cdots [N_n/y_n]$ for $0 \leq i \leq m$ and P_m contains a head normal form that is responsible for the reduction $P^m \rightarrow P^{m+1}$. By repeatedly applying Lemma 1.13 (1), we obtain Q' such that $Q \leq Q'$ and $P^m \gg Q'$. Since $P^m \rightarrow P^{m+1}$, there exists Q^1 such that $Q' \rightarrow Q^1$ and $P^{m+1} \gg Q^1$ by Lemma 1.13 (2). Hence by Lemma 1.10 (1), we obtain $Q = Q^0 \rightarrow Q^1$.

In the former case, we are already done. In the latter case, we still have an infinite reduction sequence $P^{m+1} \rightarrow P^{m+2} \rightarrow \cdots$ and $P^{m+1} \gg Q^1$. Hence we may repeat the same argument to prolong the reduction sequence $Q^0 \rightarrow Q^1$. Hence we eventually obtain $\llbracket Q \rrbracket = \Omega$. \square

Lemma 1.16. *Suppose that $P \gg Q$.*

- *If $P \rightarrow^* \ni x|\bar{a}\langle M_1, \dots, M_m \rangle$, then there exist L_1, \dots, L_m such that $Q \rightarrow^* \ni x|\bar{a}\langle L_1, \dots, L_m \rangle$ and $M_1 \gg L_1, \dots, M_m \gg L_m$.*
- *Conversely, if $Q \rightarrow^* \ni x|\bar{a}\langle L_1, \dots, L_m \rangle$, then there exist M_1, \dots, M_m such that $P \rightarrow^* \ni x|\bar{a}\langle M_1, \dots, M_m \rangle$ and $M_1 \gg L_1, \dots, M_m \gg L_m$.*

Proof. Suppose that $P \rightarrow^* P' \ni x|\bar{a}\langle M_1, \dots, M_m \rangle$. By Lemmas 1.13 and 1.10 (1) (which states that the composed relation $\leq \rightarrow$ is identical with \rightarrow), there is Q' such that $Q \rightarrow^* \leq Q'$ and $P' \gg Q'$. Since $P' \ni x|\bar{a}\langle \vec{M} \rangle$, we may write

$$x|\bar{a}\langle \vec{M} \rangle = x|\bar{a}\langle \vec{K} \rangle [N_1/y_1] \cdots [N_n/y_n] = x|\bar{a}\langle \vec{K} \rangle [N_1/y_1] \cdots [N_n/y_n]$$

for some $\vec{K} = K_1, \dots, K_m$, where $x \notin \{y_1, \dots, y_n\}$, and Q' contains

$$\llbracket x|\bar{a}\langle\vec{K}\rangle\rrbracket\llbracket\llbracket N_1\rrbracket/y_1\rrbracket\cdots\llbracket\llbracket N_n\rrbracket/y_n\rrbracket = x|\bar{a}\langle\llbracket\vec{K}\rrbracket\rrbracket\llbracket\llbracket N_1\rrbracket/y_1\rrbracket\cdots\llbracket\llbracket N_n\rrbracket/y_n\rrbracket\langle\rangle.$$

Hence by letting $L_i = \llbracket K_i\rrbracket\llbracket\llbracket N_1\rrbracket/y_1\rrbracket\cdots\llbracket\llbracket N_n\rrbracket/y_n\rrbracket$ we obtain $M_i \gg L_i$ for every $1 \leq i \leq m$. Since $Q \rightarrow^* \leq Q' \ni x|\bar{a}\langle\vec{L}\rangle$, namely $Q \rightarrow^* \ni x|\bar{a}\langle\vec{L}\rangle$, the claim holds.

Conversely, suppose that $Q \rightarrow^* Q' \ni x|\bar{a}\langle L_1, \dots, L_m \rangle$. By Lemma 1.14, there is P' such that $P \rightarrow^* P'$ and $P' \gg Q'$. The rest is similar to the above. \square

Lemma 1.17. *If $M \gg N$, then either $M = y = N$ for a variable y , or $M = \sum a(\vec{x}_a).P_a$, $N = \sum a(\vec{x}_a).Q_a$ and $P_a \gg Q_a$ for every $a \in A$.*

Proof. Immediate. \square

The following lemma completes the proof of Theorem 1.12.

Lemma 1.18. *If $D_0 \gg E_0$, then $\llbracket D_0 \rrbracket = \llbracket E_0 \rrbracket$.*

Proof. Define a binary relation \mathcal{R} on designs as follows:

- For positive (resp. negative) designs D, E , we have $D \mathcal{R} E$ if $D = \llbracket D_0 \rrbracket$, $E = \llbracket E_0 \rrbracket$, and $D_0 \gg E_0$ for some D_0 and E_0 .
- For predesigns S, T , we have $S \mathcal{R} T$ if $S = x|\bar{a}\langle\llbracket M_1 \rrbracket, \dots, \llbracket M_m \rrbracket\rangle$, $T = x|\bar{a}\langle\llbracket L_1 \rrbracket, \dots, \llbracket L_m \rrbracket\rangle$, and $M_i \gg L_i$ for every $1 \leq i \leq m$.

We now verify that this \mathcal{R} satisfies the conditions of Lemma 1.3.

First, let P, Q be positive designs such that $P \mathcal{R} Q$, i.e., $P = \llbracket P_0 \rrbracket$, $Q = \llbracket Q_0 \rrbracket$, and $P_0 \gg Q_0$ for some P_0 and Q_0 .

- If $\llbracket P_0 \rrbracket = \Omega$, then $\llbracket Q_0 \rrbracket = \Omega$ too by Lemma 1.15. Hence (1) holds.
- If $\llbracket P_0 \rrbracket$ is a conjunction, then $\llbracket Q_0 \rrbracket$ is not Ω by Lemma 1.15, so is a conjunction. If $\llbracket P_0 \rrbracket$ contains $x|\bar{a}\langle\llbracket M_1 \rrbracket, \dots, \llbracket M_m \rrbracket\rangle$, then $P_0 \rightarrow^* \ni x|\bar{a}\langle\vec{M}\rangle$. Since $P_0 \gg Q_0$, Lemma 1.16 yields $Q_0 \rightarrow^* \ni x|\bar{a}\langle\vec{L}\rangle$ for some $\vec{L} = L_1, \dots, L_m$. Namely, $\llbracket Q_0 \rrbracket$ contains $x|\bar{a}\langle\llbracket\vec{L}\rrbracket\rangle$. Moreover, we have $M_i \gg L_i$ for every $1 \leq i \leq m$. Similarly, one can show that if $\llbracket Q_0 \rrbracket$ contains $x|\bar{a}\langle\llbracket\vec{L}\rrbracket\rangle$, then $\llbracket P_0 \rrbracket$ contains $x|\bar{a}\langle\llbracket\vec{M}\rrbracket\rangle$ and $M_i \gg L_i$ for every $1 \leq i \leq m$. Hence (2) holds.

Let S, T be predesigns such that $S \mathcal{R} T$, i.e., $S = x|\bar{a}\langle\llbracket M_1 \rrbracket, \dots, \llbracket M_m \rrbracket\rangle$, $T = x|\bar{a}\langle\llbracket L_1 \rrbracket, \dots, \llbracket L_m \rrbracket\rangle$, and $M_i \gg L_i$ for every $1 \leq i \leq m$. It immediately follows that $\llbracket M_i \rrbracket \mathcal{R} \llbracket L_i \rrbracket$ for every $1 \leq i \leq m$. Also, $x \mathcal{R} x$. Hence (3) holds.

Finally, let N, M be negative designs such that $N = \llbracket N_0 \rrbracket$, $M = \llbracket M_0 \rrbracket$, and $N_0 \gg M_0$ for some N_0 and M_0 .

- If $N = x$, then $N_0 = M_0 = M = x$. Hence (4) holds.
- Otherwise, N must be of the form $\sum a(\vec{x}_a).\llbracket P_a \rrbracket$ and $N_0 = \sum a(\vec{x}_a).P_a$. Since $N_0 \gg M_0$, M_0 is of the form $\sum a(\vec{x}_a).Q_a$ and $P_a \gg Q_a$ for every $a \in A$ by Lemma 1.17. So $M = \llbracket M_0 \rrbracket = \sum a(\vec{x}_a).\llbracket Q_a \rrbracket$ and $\llbracket P_a \rrbracket \mathcal{R} \llbracket Q_a \rrbracket$. Hence (5) holds.

Therefore, if $D_0 \gg E_0$, we have $\llbracket D_0 \rrbracket = \llbracket E_0 \rrbracket$ by Lemma 1.3. \square

2. BEHAVIOURS

This section is concerned with the type structure of ludics. We describe orthogonality and behaviours in 2.1, logical connectives in 2.2 and finally explain (the failure of) internal completeness of logical connectives in 2.3.

2.1. Orthogonality. In the rest of this paper, we mainly restrict ourselves to a special subclass of designs: we only consider designs which are *total*, *cut-free*, and *identity-free*. Generalizing the terminology in [30], we call them *standard designs*. In other words:

Definition 2.1 (Standard design). A design D is said **standard** if it satisfies the following two conditions:

- (i) *Cut-freeness and identity-freeness*: D can be coinductively generated by the following restricted version of the grammar given in Definition 1.1:

$$\begin{aligned} P &::= \Omega \mid \bigwedge \{S_i : i \in I\}, \\ S &::= x|\bar{a}\langle N_1, \dots, N_n \rangle, \\ N &::= \sum a(\vec{x}).P_a. \end{aligned}$$

- (ii) *Totality*: $D \neq \Omega$.

The totality condition is due to the original work [21]. It has a pleasant consequence that behaviours (see below) are never empty. We also remark that the lack of identities can be somehow compensated by considering their infinitary η expansions, called *faxes* in [21]. In our setting, the infinitary η expansion of an identity x is expressed by the negative standard design $\eta(x)$ defined by the equation:

$$\eta(x) = \sum a(y_1, \dots, y_n).x|\bar{a}\langle \eta(y_1), \dots, \eta(y_n) \rangle.$$

We refer to [30] for more details.

We are now ready to define orthogonality and behaviours.

Definition 2.2 (Orthogonality). A positive design P is said **atomic** if it is standard and $\text{fv}(P) \subseteq \{x_0\}$ for a certain fixed variable x_0 .¹

A negative design N is said **atomic** if it is standard and $\text{fv}(N) = \emptyset$.

Two atomic designs P, N of opposite polarities are said **orthogonal** and written $P \perp N$ (or equivalently $N \perp P$) when $\llbracket P[N/x_0] \rrbracket = \mathbf{X}$.

If \mathbf{X} is a set of atomic designs of the same polarity, then its **orthogonal set**, denoted by \mathbf{X}^\perp , is defined by $\mathbf{X}^\perp := \{E : \forall D \in \mathbf{X}, D \perp E\}$.

The meaning of \bigwedge and the associated partial order \leq can be clarified in terms of orthogonality. For atomic designs D, E of the same polarity, define $D \leq E$ if and only if $\{D\}^\perp \subseteq \{E\}^\perp$. $D \leq E$ means that E has more chances of convergence than D when interacting with other atomic designs. The following is easy to observe.

Proposition 2.3.

- (1) \leq is a preorder.
- (2) $P \leq Q$ implies $P \preceq Q$ for any pair of atomic positive designs P, Q .
- (3) Let \mathbf{X} and \mathbf{Y} be sets of atomic designs of the same polarity. Then $\mathbf{X} \subseteq \mathbf{Y}$ implies $\bigwedge \mathbf{Y} \preceq \bigwedge \mathbf{X}$. □

¹ The variable x_0 here plays the same role as the empty address “ $\langle \rangle$ ” does in [21]: x_0 may be thought of as a fixed and predetermined “location.”

In particular, $\Omega \preceq P \preceq \mathfrak{X}$ for any atomic positive design P .² This justifies our identification of \mathfrak{X} with the empty conjunction $\bigwedge \emptyset$.

Remark 2.4. Designs in [21] satisfy the *separation property*: for any designs D, E of the same polarity, we have $D = E$ if and only if $\{D\}^\perp = \{E\}^\perp$. But when the constraint of linearity is removed, this property no more holds, as observed in [26] (see also [10]).

In our setting, separation does not hold, even when D and E are deterministic (atomic) designs. For instance, consider the following two designs [26]:

$$\begin{aligned} P &:= x_0 | \downarrow \langle \uparrow(y). \mathfrak{X} \rangle, \\ Q &:= x_0 | \downarrow \langle \uparrow(y). P \rangle = x_0 | \downarrow \langle \uparrow(y). x_0 | \downarrow \langle \uparrow(y). \mathfrak{X} \rangle \rangle. \end{aligned}$$

It is easy to see that in our setting $P \perp N$ holds if and only if N has an additive component of the form $\uparrow(z). \bigwedge \{z | \downarrow \langle M_i \rangle : i \in I\}$ for arbitrary index set I and arbitrary standard negative designs M_i with $\text{fv}(M_i) \subseteq \{z\}$.

The same holds for Q , as can be observed from the following reduction sequence (for readability, we only consider the case in which N has a component of the form $\uparrow(z). z | \downarrow \langle M \rangle$, the general case easily follows):

$$\begin{aligned} Q[N/x_0] &= N | \downarrow \langle \uparrow(y). P[N/x_0] \rangle \\ &\rightarrow (\uparrow(y). P[N/x_0]) | \downarrow \langle M[\uparrow(y). P[N/x_0]/z] \rangle \\ &\rightarrow P[N/x_0] \rightarrow^* \mathfrak{X}. \end{aligned}$$

(If N does not have a component of the form discussed above, we have $Q[N/x_0] \rightarrow^* \Omega$.)

We therefore conclude $\{P\}^\perp = \{Q\}^\perp$, even though $P \neq Q$.

Although possible, we do not define orthogonality for nonatomic designs. Accordingly, we only consider *atomic* behaviours which consist of atomic designs.

Definition 2.5 (Behaviour). A **behaviour** \mathbf{X} is a set of atomic standard designs of the same polarity such that $\mathbf{X}^{\perp\perp} = \mathbf{X}$.

A behaviour is positive or negative according to the polarity of its designs. We denote positive behaviours by $\mathbf{P}, \mathbf{Q}, \mathbf{R}, \dots$ and negative behaviours by $\mathbf{N}, \mathbf{M}, \mathbf{K}, \dots$.

Orthogonality satisfies the following standard properties:

Proposition 2.6. *Let \mathbf{X}, \mathbf{Y} be sets of atomic designs of the same polarity. We have:*

- (1) $\mathbf{X} \subseteq \mathbf{X}^{\perp\perp}$.
- (2) $\mathbf{X} \subseteq \mathbf{Y} \implies \mathbf{Y}^\perp \subseteq \mathbf{X}^\perp$.
- (3) $\mathbf{X} \subseteq \mathbf{Y}^{\perp\perp} \implies \mathbf{X}^{\perp\perp} \subseteq \mathbf{Y}^{\perp\perp}$.
- (4) $\mathbf{X}^\perp = \mathbf{X}^{\perp\perp\perp}$. *In particular, any orthogonal set is a behaviour.*
- (5) $(\mathbf{X} \cup \mathbf{Y})^\perp = \mathbf{X}^\perp \cap \mathbf{Y}^\perp$. *In particular, the intersection of two behaviours is a behaviour. \square*

² Here we are tentatively considering the nontotal design Ω , which does not officially belong to the universe of atomic designs.

We also observe that $D \preceq E$ and $D \in \mathbf{X}$ implies $E \in \mathbf{X}$ when \mathbf{X} is a behaviour.

Among all positive (resp. negative) behaviours, there exist the least and the greatest behaviours with respect to set inclusion:

$$\begin{aligned} \mathbf{0}^+ &:= \{\mathfrak{X}\}^{\perp\perp} = \{\mathfrak{X}\}, & \top^- &:= (\mathbf{0}^+)^{\perp} = \{ \text{atomic negative designs} \}, \\ \mathbf{0}^- &:= \{\mathfrak{X}^-\}^{\perp\perp} = \{\mathfrak{X}^-\}, & \top^+ &:= (\mathbf{0}^-)^{\perp} = \{ \text{atomic positive designs} \}, \end{aligned}$$

where $\mathfrak{X}^- := \sum a(\bar{x}).\mathfrak{X}$ plays the role of the design called *negative daimon* in [21]. Notice that behaviours are always nonempty due to the totality condition: any positive (resp. negative) behaviour contains \mathfrak{X} (resp. \mathfrak{X}^-).

Now that we have given behaviours, we can define *contexts of behaviours* and then the *semantical entailment* \models in order to relate designs to contexts of behaviours. These constructs play the role of typing environments in type systems. They correspond to *sequents of behaviours*, in the terminology of [21].

Definition 2.7 (Contexts of behaviours and semantical entailment \models).

- (a) A **positive context** Γ is of the form $x_1 : \mathbf{P}_1, \dots, x_n : \mathbf{P}_n$, where x_1, \dots, x_n are distinct variables and $\mathbf{P}_1, \dots, \mathbf{P}_n$ are (atomic) positive behaviours. We denote by $\text{fv}(\Gamma)$ the set $\{x_1, \dots, x_n\}$.

A **negative context** Γ, \mathbf{N} is a positive context Γ enriched with an (atomic) negative behaviour \mathbf{N} , to which no variable is associated.

- (b) The **semantical entailment** is the binary relation \models between designs and contexts of behaviours of the same polarity defined as follows:

$P \models x_1 : \mathbf{P}_1, \dots, x_n : \mathbf{P}_n$ if and only if:

- P is standard;
- $\text{fv}(P) \subseteq \{x_1, \dots, x_n\}$;
- $\llbracket P[K_1/x_1, \dots, K_n/x_n] \rrbracket = \mathfrak{X}$ for any $K_1 \in \mathbf{P}_1^{\perp}, \dots, K_n \in \mathbf{P}_n^{\perp}$.

$N \models x_1 : \mathbf{P}_1, \dots, x_n : \mathbf{P}_n, \mathbf{N}$ if and only if:

- N is standard;
- $\text{fv}(N) \subseteq \{x_1, \dots, x_n\}$;
- $\llbracket Q[N[K_1/x_1, \dots, K_n/x_n]/x_0] \rrbracket = \mathfrak{X}$ for any $K_1 \in \mathbf{P}_1^{\perp}, \dots, K_n \in \mathbf{P}_n^{\perp}, Q \in \mathbf{N}^{\perp}$.

Clearly, $N \models \mathbf{N}$ if and only if $N \in \mathbf{N}$, and $P \models y : \mathbf{P}$ if and only if $P[x_0/y] \in \mathbf{P}$. Furthermore, associativity (Theorem 1.12) implies the following quite useful principle:

Lemma 2.8 (Closure principle).

- (1) $P \models \Gamma, z : \mathbf{P}$ if and only if $\llbracket P[M/z] \rrbracket \models \Gamma$ for any $M \in \mathbf{P}^{\perp}$;
- (2) $N \models \Gamma, \mathbf{N}$ if and only if $\llbracket Q[N/x_0] \rrbracket \models \Gamma$ for any $Q \in \mathbf{N}^{\perp}$;
- (3) $N \models \Gamma, z : \mathbf{P}, \mathbf{N}$ if and only if $\llbracket N[M/z] \rrbracket \models \Gamma, \mathbf{N}$ for any $M \in \mathbf{P}^{\perp}$.

Proof.

- (1) Let P be a standard design with $\text{fv}(P) \subseteq \{x_1, \dots, x_n, z\}$ and Γ a context $x_1 : \mathbf{P}_1, \dots, x_n : \mathbf{P}_n$.

First, we claim that $\llbracket P[M/z] \rrbracket$ is a standard design when $P \models \Gamma, z : \mathbf{P}$ and $M \in \mathbf{P}^{\perp}$. Indeed, it is obviously cut-free. It is also identity-free because so are P, M and neither substitution $P[M/z]$ nor normalization $\llbracket P[M/z] \rrbracket$ introduces identities. Totality will be shown below. We also note that $\text{fv}(\llbracket P[M/z] \rrbracket) \subseteq \{x_1, \dots, x_n\}$, since M is an atomic negative design that is always closed.

Next, we observe that $\llbracket P[\vec{K}/\vec{x}, M/z] \rrbracket = \llbracket \llbracket P[M/z] \rrbracket [\vec{K}/\vec{x}] \rrbracket$ for any list $\vec{K} = K_1, \dots, K_n$ of standard negative designs. Indeed, notice that $P[\vec{K}/\vec{x}, M/z] = P[M/z][\vec{K}/\vec{x}]$ since M is closed, and $\llbracket K_i \rrbracket = K_i$ since K_i is cut-free. Hence by associativity, we obtain:

$$\llbracket P[\vec{K}/\vec{x}, M/z] \rrbracket = \llbracket P[M/z][\vec{K}/\vec{x}] \rrbracket = \llbracket \llbracket P[M/z] \rrbracket \llbracket [\vec{K}/\vec{x}] \rrbracket \rrbracket = \llbracket \llbracket P[M/z] \rrbracket [\vec{K}/\vec{x}] \rrbracket.$$

In particular, $\llbracket P[\vec{K}/\vec{x}, M/z] \rrbracket = \mathfrak{X}$ implies the totality of $\llbracket P[M/z] \rrbracket$.

We are now ready to prove the first claim. Writing $\vec{K} \in \Gamma^\perp$ for $K_1 \in \mathbf{P}_1^\perp, \dots, K_n \in \mathbf{P}_n^\perp$, we have:

$$\begin{aligned} P \models \Gamma, z : \mathbf{P} &\iff \llbracket P[\vec{K}/\vec{x}, M/z] \rrbracket = \mathfrak{X} \text{ for every } \vec{K} \in \Gamma^\perp \text{ and } M \in \mathbf{P}^\perp, \\ &\iff \llbracket \llbracket P[M/z] \rrbracket [\vec{K}/\vec{x}] \rrbracket = \mathfrak{X} \text{ for every } \vec{K} \in \Gamma^\perp \text{ and } M \in \mathbf{P}^\perp, \\ &\iff \llbracket P[M/z] \rrbracket \models \Gamma \text{ for every } M \in \mathbf{P}^\perp. \end{aligned}$$

(2) and (3) are proven in a similar way. We just mention that the crucial equalities

$$\begin{aligned} \llbracket Q[N[\vec{K}/\vec{x}]/x_0] \rrbracket &= \llbracket \llbracket Q[N/x_0] \rrbracket [\vec{K}/\vec{x}] \rrbracket, \\ \llbracket Q[N[\vec{K}/\vec{x}, M/z]/x_0] \rrbracket &= \llbracket Q[\llbracket N[M/z] \rrbracket [\vec{K}/\vec{x}]/x_0] \rrbracket, \end{aligned}$$

which are needed to show (2) and (3) respectively, can be straightforwardly derived from associativity. \square

2.2. Logical connectives. We next describe how to build behaviours by means of *logical connectives* in ludics.

Definition 2.9 (Logical connectives). An n -ary **logical connective** α is a pair $\alpha = (\vec{z}, \alpha_0)$ where:

- $\vec{z} = z_1, \dots, z_n$ is a sequence of distinct variables;
- $\alpha_0 = \{a_1(\vec{x}_1), \dots, a_k(\vec{x}_k)\}$ is a finite set of negative actions such that:
 - the names a_1, \dots, a_k are distinct;
 - $\{\vec{x}_i\} \subseteq \{z_1, \dots, z_n\}$ for each $1 \leq i \leq k$.

Two logical connectives are identified if one is obtained from another by renaming of variables.

We can intuitively explain the structure of logical connectives in terms of standard connectives of linear logic as follows.

The variables z_1, \dots, z_n play the role of *placeholders* for (immediate) subformulas, while α_0 determines the *logical structure* of α . An action $a(x_1, \dots, x_m) \in \alpha_0$ can be seen as a kind of m -ary “tensor product” $x_1 \otimes \dots \otimes x_m$ indexed by the name a . The whole set α_0 can be thought of as k -ary “additive sum” of its elements:

$$\cdots \oplus \underbrace{(x_1 \otimes \cdots \otimes x_m)}_a \oplus \cdots.$$

In Appendix A we give a more precise correspondence between logical connective in our sense and connectives of polarized linear logic [25].

Example 2.10. Consider the logical connective $\alpha = (x, y, z, t, \{a(x, y, t), b(t, x), c(y, x)\})$. By the previous discussion, we can intuitively think of it as

$$\underbrace{(x \otimes y \otimes t)}_a \oplus \underbrace{(t \otimes x)}_b \oplus \underbrace{(y \otimes x)}_c.$$

When α is applied to $\mathbf{N}, \mathbf{M}, \mathbf{K}, \mathbf{L}$, it gives the formula

$$\underbrace{(\mathbf{N} \otimes \mathbf{M} \otimes \mathbf{L})}_a \oplus \underbrace{(\mathbf{L} \otimes \mathbf{N})}_b \oplus \underbrace{(\mathbf{M} \otimes \mathbf{N})}_c.$$

We now define behaviours built by logical connectives.

Definition 2.11 (Behaviours defined by logical connectives). Given an m -ary name a , an n -ary logical connective $\alpha = (\vec{z}, \alpha_0)$ with $\vec{z} = z_1, \dots, z_n$ and behaviours $\mathbf{N}_1, \dots, \mathbf{N}_n, \mathbf{P}_1, \dots, \mathbf{P}_n$ we define:

- $\bar{a}\langle \mathbf{N}_1, \dots, \mathbf{N}_m \rangle := \{x_0 | \bar{a}\langle N_1, \dots, N_m \rangle : N_1 \in \mathbf{N}_1, \dots, N_m \in \mathbf{N}_m\}$,
- $\bar{\alpha}\langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle := \left(\bigcup_{a(\vec{x}) \in \alpha_0} \bar{a}\langle \mathbf{N}_{i_1}, \dots, \mathbf{N}_{i_m} \rangle \right)^{\perp\perp}$,
- $\alpha\langle \mathbf{P}_1, \dots, \mathbf{P}_n \rangle := \bar{\alpha}\langle \mathbf{P}_1^\perp, \dots, \mathbf{P}_n^\perp \rangle^\perp$,

where the indices $i_1, \dots, i_m \in \{1, \dots, n\}$ vary for each $a(\vec{x}) \in \alpha_0$ and are determined by the variables $\vec{x} = z_{i_1}, \dots, z_{i_m}$. We call the set

$$\bar{\alpha}_{\text{eth}}\langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle := \bigcup_{a(\vec{x}) \in \alpha_0} \bar{a}\langle \mathbf{N}_{i_1}, \dots, \mathbf{N}_{i_m} \rangle$$

the *ethics* of $\bar{\alpha}\langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle$.

Remark 2.12. An ethics is a set of atomic predesigns which are by construction linear in x_0 . It can be seen as a “generator” of a behaviour defined by logical connectives in the following sense. For positives, we have by definition $\bar{\alpha}\langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle = \bar{\alpha}_{\text{eth}}\langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle^{\perp\perp}$. For negatives, we have by Proposition 2.6 (3):

$$\begin{aligned} \alpha\langle \mathbf{P}_1, \dots, \mathbf{P}_n \rangle &= \bar{\alpha}\langle \mathbf{P}_1^\perp, \dots, \mathbf{P}_n^\perp \rangle^\perp \\ &= \bar{\alpha}_{\text{eth}}\langle \mathbf{P}_1^\perp, \dots, \mathbf{P}_n^\perp \rangle^{\perp\perp\perp} \\ &= \bar{\alpha}_{\text{eth}}\langle \mathbf{P}_1^\perp, \dots, \mathbf{P}_n^\perp \rangle^\perp. \end{aligned}$$

Example 2.13. Let α be the logical connective as given in Example 2.10 and $\mathbf{N}, \mathbf{M}, \mathbf{K}, \mathbf{L}$ negative behaviours. We have $\bar{\alpha}_{\text{eth}}\langle \mathbf{N}, \mathbf{M}, \mathbf{K}, \mathbf{L} \rangle = \bar{a}\langle \mathbf{N}, \mathbf{M}, \mathbf{L} \rangle \cup \bar{b}\langle \mathbf{L}, \mathbf{N} \rangle \cup \bar{c}\langle \mathbf{M}, \mathbf{N} \rangle$.

Example 2.14 (Linear logic connectives). Logical connectives $\wp, \&, \uparrow, \perp, \top$ can be defined if the signature \mathcal{A} contains a nullary name $*$, unary names \uparrow, π_1, π_2 and a binary name \wp . We also give notations to their duals for readability.

$$\begin{array}{lll} \wp & := & (x_1, x_2, \{\wp(x_1, x_2)\}), & \otimes & := & \overline{\wp}, & \bullet & := & \overline{\wp}, \\ \& & := & (x_1, x_2, \{\pi_1(x_1), \pi_2(x_2)\}), & \oplus & := & \overline{\&}, & \iota_i & := & \overline{\pi_i}, \\ \uparrow & := & (x, \{\uparrow(x)\}), & \downarrow & := & \overline{\uparrow}, & \downarrow & := & \overline{\uparrow}, \\ \perp & := & (\epsilon, \{*\}), & \mathbf{1} & := & \overline{\perp}, \\ \top & := & (\epsilon, \emptyset), & \mathbf{0} & := & \overline{\top}, \end{array}$$

where ϵ denotes the empty sequence. We do not have exponentials here, because we are working in a nonlinear setting so that they are already incorporated into the connectives.

With these logical connectives we can build behaviours corresponding to usual linear logic types (we use infix notations such as $\mathbf{N} \otimes \mathbf{M}$ rather than the prefix ones $\otimes(\mathbf{N}, \mathbf{M})$).

$$\begin{array}{ll}
\mathbf{N} \otimes \mathbf{M} & = \bullet \langle \mathbf{N}, \mathbf{M} \rangle^{\perp\perp}, & \mathbf{P} \wp \mathbf{Q} & = \bullet \langle \mathbf{P}^\perp, \mathbf{Q}^\perp \rangle^\perp, \\
\mathbf{N} \oplus \mathbf{M} & = (\iota_1 \langle \mathbf{N} \rangle \cup \iota_2 \langle \mathbf{M} \rangle)^{\perp\perp}, & \mathbf{P} \& \mathbf{Q} & = \iota_1 \langle \mathbf{P}^\perp \rangle^\perp \cap \iota_2 \langle \mathbf{Q}^\perp \rangle^\perp, \\
\downarrow \mathbf{N} & = \downarrow \langle \mathbf{N} \rangle^{\perp\perp}, & \uparrow \mathbf{P} & = \downarrow \langle \mathbf{P}^\perp \rangle^\perp, \\
\mathbf{1} & = \{x_0 | \bar{*}\}^{\perp\perp}, & \perp & = \{x_0 | \bar{*}\}^\perp, \\
\mathbf{0} & = \emptyset^{\perp\perp}, & \top & = \emptyset^\perp.
\end{array}$$

The next theorem illustrates a special feature of behaviours defined by logical connectives. It also suggests that nonlinearity and universal nondeterminism play dual roles.

Theorem 2.15. *Let \mathbf{P} be an arbitrary positive behaviour.*

- (1) $P \models x_1 : \mathbf{P}, x_2 : \mathbf{P} \implies P[x_0/x_1, x_0/x_2] \in \mathbf{P}$.
- (2) $\bigwedge \mathbf{X} \in \mathbf{P}^\perp \implies \mathbf{X} \subseteq \mathbf{P}^\perp$.

Moreover, if \mathbf{P} is obtained by applying a logical connective, that is $\mathbf{P} = \bar{\alpha} \langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle$ for some α , $\mathbf{N}_1, \dots, \mathbf{N}_n$, then:

- (3) the converse of (1) (duplicability) and
- (4) the converse of (2) (closure under \bigwedge) hold.

Proof.

- (1) For any $N \in \mathbf{P}^\perp$, we have $\llbracket P[N/x_1, N/x_2] \rrbracket = \wp$. Hence, $\llbracket P[x_0/x_1, x_0/x_2][N/x_0] \rrbracket = \wp$, and so $P[x_0/x_1, x_0/x_2] \in \mathbf{P}^{\perp\perp} = \mathbf{P}$.
- (2) By Proposition 2.3 (3), we have $\bigwedge \mathbf{X} \preceq \bigwedge \{N\} = N$ for any $N \in \mathbf{X}$. Since \mathbf{P}^\perp is a behaviour, it is upward closed with respect to \preceq . Hence the claim holds.
- (4) For the sake of readability, we consider the binary case and show that $N, M \models \mathbf{P}^\perp$ implies $N \wedge M \models \mathbf{P}^\perp$. The general case can be proven using the same argument.

Let $\mathbf{P}^\perp = \bar{\alpha} \langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle^\perp = \alpha \langle \mathbf{N}_1^\perp, \dots, \mathbf{N}_n^\perp \rangle$. To prove $N \wedge M \in \mathbf{P}^\perp$, by Remark 2.12, it is sufficient to show that $N \wedge M$ is orthogonal to any $x_0 | \bar{\alpha} \langle \vec{K} \rangle \in \bar{\alpha}_{\text{eth}} \langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle$. Since by construction x_0 occurs only once at the head position of $x_0 | \bar{\alpha} \langle \vec{K} \rangle$, we only have to show that $\llbracket N \wedge M \mid \bar{\alpha} \langle \vec{K} \rangle \rrbracket = \wp$.

Let $N = \sum a(\vec{x}).P_a$ and $M = \sum a(\vec{x}).Q_a$ so that $N \wedge M = \sum a(\vec{x}).(P_a \wedge Q_a)$. Since $N \wedge M \mid \bar{\alpha} \langle \vec{K} \rangle$ is a predesign, we have by Lemma 1.10 (2), (3):

$$\llbracket N \wedge M \mid \bar{\alpha} \langle \vec{K} \rangle \rrbracket = \llbracket P_a \wedge Q_a[\vec{K}/\vec{x}] \rrbracket = \llbracket P_a[\vec{K}/\vec{x}] \rrbracket \wedge \llbracket Q_a[\vec{K}/\vec{x}] \rrbracket = \llbracket N \mid \bar{\alpha} \langle \vec{K} \rangle \rrbracket \wedge \llbracket M \mid \bar{\alpha} \langle \vec{K} \rangle \rrbracket.$$

Since $N, M \in \mathbf{P}^\perp$, we have $\llbracket N \mid \bar{\alpha} \langle \vec{K} \rangle \rrbracket = \wp$ and $\llbracket M \mid \bar{\alpha} \langle \vec{K} \rangle \rrbracket = \wp$. Our claim then immediately follows.

- (3) Let $P[x_0/x_1, x_0/x_2] \in \mathbf{P} = \bar{\alpha} \langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle$. It suffices to show that $\llbracket P[N/x_1, M/x_2] \rrbracket = \wp$ holds for any $N, M \in \mathbf{P}^\perp$. But we have just proven that $N \wedge M \in \mathbf{P}^\perp$, and so $\llbracket P[x_0/x_1, x_0/x_2][N \wedge M/x_0] \rrbracket = \llbracket P[N \wedge M/x_1, N \wedge M/x_2] \rrbracket = \wp$. Since $N \wedge M \preceq N, M$ by Proposition 2.3 (3), we have $\llbracket P[N/x_1, M/x_2] \rrbracket = \wp$. \square

Remark 2.16. Theorem 2.15 can be considered as an internal, monistic form of soundness and completeness for the contraction rule: soundness corresponds to point (1) while completeness to its converse (3), duplicability.

However, in the sequel we only use point (1) (in Theorem 3.5) and point (4) (in Lemma 3.10) of Theorem 2.15.

2.3. Internal completeness. In [21], Girard proposes a purely monistic, local notion of completeness, called *internal completeness*. It means that we can give a precise and direct description to the elements of behaviours (built by logical connectives) without using the orthogonality and without referring to any proof system. It is easy to see that negative logical connectives enjoy internal completeness:

Theorem 2.17 (Internal completeness (negative case)). *Let $\alpha = (\vec{z}, \alpha_0)$ be a logical connective with $\vec{z} = z_1, \dots, z_n$ and $N = \sum a(\vec{x}).P_a$ an atomic negative design. We have:*

$$N \in \alpha(\mathbf{P}_1, \dots, \mathbf{P}_n) \iff P_a \models z_{i_1} : \mathbf{P}_{i_1}, \dots, z_{i_m} : \mathbf{P}_{i_m}, \text{ for every } a(\vec{x}) \in \alpha_0,$$

where the indices $i_1, \dots, i_m \in \{1, \dots, n\}$ are determined by the variables $\vec{x} = z_{i_1}, \dots, z_{i_m}$.

Proof. Let $N = \sum a(\vec{x}).P_a$ be an atomic negative design and $P = x_0 | \bar{a}\langle N_1, \dots, N_m \rangle \in \bar{\alpha}_{\text{eth}}\langle \mathbf{P}_1^\perp, \dots, \mathbf{P}_n^\perp \rangle = \bigcup_{a(\vec{x}) \in \alpha_0} \bar{a}\langle \mathbf{P}_{i_1}^\perp, \dots, \mathbf{P}_{i_m}^\perp \rangle$. Since $P[N/x_0]$ is a pre-design and x_0 occurs only at the head position of P , we have by Lemma 1.10 (2):

$$\llbracket P[N/x_0] \rrbracket = \llbracket \sum a(\vec{x}).P_a \mid \bar{a}\langle N_1, \dots, N_m \rangle \rrbracket = \llbracket P_a[N_1/z_{i_1}, \dots, N_m/z_{i_m}] \rrbracket.$$

This means that $N \in \bar{\alpha}_{\text{eth}}\langle \mathbf{P}_1^\perp, \dots, \mathbf{P}_n^\perp \rangle^\perp = \alpha(\mathbf{P}_1, \dots, \mathbf{P}_n)$ (see Remark 2.12) if and only if for every $a(\vec{x}) \in \alpha_0$ and for every $N_1 \in \mathbf{P}_{i_1}^\perp, \dots, N_m \in \mathbf{P}_{i_m}^\perp$, $\llbracket P_a[N_1/z_{i_1}, \dots, N_m/z_{i_m}] \rrbracket = \boxtimes$ if and only if for every $a(\vec{x}) \in \alpha_0$, $P_a \models z_{i_1} : \mathbf{P}_{i_1}, \dots, z_{i_m} : \mathbf{P}_{i_m}$ (see Definition 2.7 (b)). \square

Notice that in the above, P_b can be arbitrary when $b(\vec{y}) \notin \alpha_0$. Thus our approach is “immaterial” in that we do not consider material designs (see *e.g.*, [21, 8, 30] for the definition of material design). The original “material” version of internal completeness [21] can be easily derived from our immaterial one.

Remark 2.18. A remarkable example of internal completeness for negative behaviours is provided for the logical connective $\& = (x_1, x_2, \{\pi_1(x_1), \pi_2(x_2)\})$:

$$\begin{aligned} N \in \mathbf{P} \& \mathbf{Q} &\iff N = \pi_1(x_1).P + \pi_2(x_2).Q + \dots, \text{ for some } P \models x_1 : \mathbf{P} \text{ and } Q \models x_2 : \mathbf{Q} \\ &\iff N = \pi_1(x_0).P + \pi_2(x_0).Q + \dots, \text{ for some } P \in \mathbf{P} \text{ and } Q \in \mathbf{Q}. \end{aligned}$$

Above, the irrelevant components of the sum are suppressed by “ \dots ” Up to materiality (*i.e.*, removal of irrelevant additive components), $\mathbf{P} \& \mathbf{Q}$, which has been defined by *intersection*, is isomorphic to the *cartesian product* of \mathbf{P} and \mathbf{Q} . This isomorphism is called “*the mystery of incarnation*” in [21].

As to positive connectives, [21] proves internal completeness theorems for additive and multiplicative ones separately in the linear and deterministic setting. They are integrated in [30] as follows:

Theorem 2.19 (Internal completeness (linear, positive case)). *When the universe of standard designs is restricted to linear and deterministic ones, we have*

$$\bar{\alpha}\langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle = \bar{\alpha}_{\text{eth}}\langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle \cup \{\boxtimes\}.$$

\square

However, this is no more true with nonlinear designs. A counterexample is given below.

Example 2.20. Let us consider the behaviour $\mathbf{P} := \downarrow\langle\uparrow(\mathbf{0})\rangle = \downarrow_{\text{eth}}\langle\uparrow(\mathbf{0})\rangle^{\perp\perp}$ and the designs $P = x_0|\downarrow\langle\uparrow(y).\mathbf{X}\rangle$ and $Q = x_0|\downarrow\langle\uparrow(y).P\rangle$ of Remark 2.4. By construction, P belongs to \mathbf{P} . Since $P \preceq Q$, Q also belongs to \mathbf{P} . However, $Q \notin \downarrow_{\text{eth}}\langle\uparrow(\mathbf{0})\rangle$, since $\uparrow(y).P$ is not atomic and so cannot belong to $\uparrow(\mathbf{0})$.

This motivates us to directly prove completeness for proofs, rather than deriving it from internal completeness as in the original work [21].

In [3] a weaker form of internal completeness is proved, which is enough to derive a weaker form of full completeness: all *finite* “winning” designs are interpretations of proofs. While such a finiteness assumption is quite common in game semantics, we will show that it can be avoided in ludics.

We end this section with the following remark.

Remark 2.21. The main linear logic isomorphism, namely the exponential one $!A \otimes !B \cong !(A \& B)$ can be expressed in our notation as $\uparrow\mathbf{P} \otimes \uparrow\mathbf{Q} \cong \downarrow(\mathbf{P} \& \mathbf{Q})$.

In our setting it is possible to prove that those behaviours are “morally” isomorphic, in the sense that they are isomorphic if we consider designs equal up to materiality³.

We can in fact define a pair of maps (f, g) on designs such that:

- $f : \uparrow\mathbf{P} \otimes \uparrow\mathbf{Q} \longrightarrow \downarrow(\mathbf{P} \& \mathbf{Q})$ and $g : \downarrow(\mathbf{P} \& \mathbf{Q}) \longrightarrow \uparrow\mathbf{P} \otimes \uparrow\mathbf{Q}$;
- if P and Q are equal up to materiality in $\uparrow\mathbf{P} \otimes \uparrow\mathbf{Q}$, then $f(P)$ and $f(Q)$ are equal up to materiality in $\downarrow(\mathbf{P} \& \mathbf{Q})$, and similarly for g ;
- for any $P \in \uparrow\mathbf{P} \otimes \uparrow\mathbf{Q}$, we have that $g(f(P))$ and P are equal up to materiality in $\uparrow\mathbf{P} \otimes \uparrow\mathbf{Q}$, and similarly for the other direction.

We postpone a detailed study of isomorphisms of types and related issues to a subsequent work.

3. PROOF SYSTEM AND COMPLETENESS FOR PROOFS

Having set up the framework, we now address the main problem: an interactive form of Gödel completeness. We first introduce the proof system in 3.1, then examine its soundness in 3.2, and finally prove completeness in 3.3, in a way quite analogous to the proof of Gödel’s theorem based on proof search (often attributed to Schütte [29]).

3.1. Proof system. We will now introduce a proof system. In our system, logical rules are automatically generated by logical connectives. Since the names which constitute the logical connectives are chosen among the names of a signature \mathcal{A} , the set of logical connectives vary for each signature \mathcal{A} . Thus, our proof system is parameterized by \mathcal{A} .

If one chooses \mathcal{A} rich enough, the constant-only fragment of polarized linear logic ([25]; see also [8]) can be embedded, as we will show in Appendix A.

In the sequel, we focus on *logical* behaviours, which are composed by using logical connectives only.

³Informally, two designs D and E are equal up to materiality in a behaviour \mathbf{G} if they only differ in occurrences of positive subdesigns which are irrelevant for the normalization against designs of \mathbf{G}^{\perp} .

Definition 3.1 (Logical behaviours). A behaviour is **logical** if it is inductively built as follows (α denotes an arbitrary logical connective):

$$\mathbf{P} := \bar{\alpha}\langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle, \quad \mathbf{N} := \alpha(\mathbf{P}_1, \dots, \mathbf{P}_n).$$

Notice that the orthogonal of a logical behaviour is again logical.

As advocated in the introduction, our monistic framework renders both proofs and models as homogeneous objects: designs.

Definition 3.2 (Proofs, Models). A **proof** is a standard design (Definition 2.1) in which all the conjunctions are unary. In other words, a proof is a total, deterministic and \mathbf{X} -free design without cuts and identities. A **model** is a linear standard design (in which conjunctions of arbitrary cardinality may occur).

We will use proofs as proof-terms for syntactic derivations in the proof system to be introduced below. In that perspective, it is reasonable to exclude designs with non-unary conjunctions from proofs, because they do not have natural counterparts in logical reasoning. For instance, the nullary conjunction (daimon) and the binary one would correspond to the following “inference rules” respectively:

$$\frac{}{\vdash \Gamma} \quad \frac{\vdash \Gamma \quad \vdash \Gamma}{\vdash \Gamma}$$

with $\vdash \Gamma$ an arbitrary sequent. Notice that we have not specified yet what a proof actually proves. Hence it might be better called “proof attempt” or “untyped proof” or “para-proof.”

On the other hand, we restrict models to linear designs just to emphasize the remarkable fact that *linear* designs do suffice for defeating any failed proof attempt that is possibly *nonlinear*.

Given a design D , let $\text{ac}^+(D)$ be the set of occurrences of positive actions \bar{a} in D . The *cardinality* of D is defined to be the cardinality of $\text{ac}^+(D)$. For instance, the fax $\eta(x) = \sum a(y_1, \dots, y_n).x|\bar{a}\langle \eta(y_1), \dots, \eta(y_n) \rangle$ (see Section 2.1) is an infinite design in this sense. Also, both proofs and models can be infinite.

A *positive* (resp. *negative*) *sequent* is a pair of the form $P \vdash \Gamma$ (resp. $N \vdash \Gamma, \mathbf{N}$) where P is a positive proof (resp. N is a negative proof) and Γ is a positive context of logical behaviours (Definition 2.7 (a)) such that $\text{fv}(P) \subseteq \text{fv}(\Gamma)$ (resp. $\text{fv}(N) \subseteq \text{fv}(\Gamma)$).

We write $D \vdash \mathbf{A}$ for a generic sequent. Intuitively, a sequent $D \vdash \mathbf{A}$ should be understood as a claim that “ D is a proof of $\vdash \mathbf{A}$ ” or “ D is of type $\vdash \mathbf{A}$.”

Our proof system consists of two sorts of inference rules:

- A *positive* rule $(\bar{\alpha}, \bar{a})$:

$$\frac{M_1 \vdash \Gamma, \mathbf{N}_{i_1} \quad \dots \quad M_m \vdash \Gamma, \mathbf{N}_{i_m} \quad (z : \bar{\alpha}\langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle \in \Gamma)}{z|\bar{a}\langle M_1, \dots, M_m \rangle \vdash \Gamma} \quad (\bar{\alpha}, \bar{a})$$

where $\alpha = (\vec{z}, \alpha_0)$, $\vec{z} = z_1, \dots, z_n$ and $a(\vec{x}) \in \alpha_0$ so that the indices $i_1, \dots, i_m \in \{1, \dots, n\}$ are determined by the variables $\vec{x} = z_{i_1}, \dots, z_{i_m}$.

- A *negative* rule (α) :

$$\frac{\{P_a \vdash \Gamma, z_{i_1} : \mathbf{P}_{i_1}, \dots, z_{i_m} : \mathbf{P}_{i_m}\}_{a(\vec{x}) \in \alpha_0}}{\sum a(\vec{x}).P_a \vdash \Gamma, \alpha(\mathbf{P}_1, \dots, \mathbf{P}_n)} \quad (\alpha)$$

where, as in the positive rule, the indices i_1, \dots, i_m are determined by the variables $\vec{x} = z_{i_1}, \dots, z_{i_m}$ for each $a(\vec{x}) \in \alpha_0$.

We assume that \vec{x} are fresh, *i.e.*, do not occur in Γ . This does not cause a loss of generality since variables in α can be renamed (see Definition 2.9).

Notice that a component $b(\vec{y}).P_b$ of $\sum a(\vec{x}).P_a$ can be arbitrary when $b(\vec{y}) \notin \alpha_0$. Hence we again take an “immaterial” approach (*cf.* Theorem 2.17).

Observe that the positive rule $(\bar{\alpha}, \bar{\alpha})$ involves implicit uses of the contraction rule on positive behaviours. The weakening rule for positive behaviours is implicit too; in the bottom up reading of a proof derivation, unused formulas are always propagated to the premises of any instance of rule. It should also be noted that proof search in our system is deterministic. In particular, given a positive sequent $z|\bar{\alpha}\langle M_1, \dots, M_m \rangle \vdash \Gamma$, the head variable z and the first positive action $\bar{\alpha}$ completely determine the next positive rule to be applied bottom-up (if there is any).

It is also possible to adopt a “material” approach in the proof system by simply requiring $P_b = \Omega$ when $b(\vec{y}) \notin \alpha_0$ in the rule (α) . Then a proof D is finite (*i.e.*, $\text{ac}^+(D)$ is a finite set) whenever $D \vdash \Lambda$ is derivable for some Λ . Thus, as in ordinary sequent calculi, our proof system accepts only *essentially finite* proofs for derivable sequents (*i.e.*, finite up to removal of irrelevant parts).

Remark 3.3. To clarify the last point, we observe that for any (*possibly infinite*) negative proof N with $\text{fv}(N) \subseteq \text{fv}(\Gamma)$, the sequent $N \vdash \Gamma, \top$ is derivable by the instance of the negative rule with $\alpha = \top = (\epsilon, \emptyset)$. In fact, this corresponds to the usual top-rule of linear logic (see also Example 3.4):

$$\frac{}{N \vdash \Gamma, \top} (\top)$$

This means that for a (possibly infinite) negative proof N there is a finite derivation of $N \vdash \Gamma, \top$. By contrast, in the “material” approach we only have

$$\frac{}{\sum a(\vec{x}).\Omega \vdash \Gamma, \top} (\top)$$

where $\sum a(\vec{x}).\Omega$ is the unique negative proof which has cardinality 0.

Example 3.4. For linear logic connectives (Example 2.14), the positive and negative rules specialize to the following (taking here the “material” approach):

$$\begin{array}{c} \frac{M_1 \vdash \Gamma, \mathbf{N}_1 \quad M_2 \vdash \Gamma, \mathbf{N}_2 \quad (z : \mathbf{N}_1 \otimes \mathbf{N}_2 \in \Gamma)}{z|\bullet\langle M_1, M_2 \rangle \vdash \Gamma} (\otimes, \bullet) \quad \frac{P \vdash \Gamma, x_1 : \mathbf{P}_1, x_2 : \mathbf{P}_2}{\wp(x_1, x_2).P \vdash \Gamma, \mathbf{P}_1 \wp \mathbf{P}_2} \wp \\ \\ \frac{M \vdash \Gamma, \mathbf{N}_i \quad (z : \mathbf{N}_1 \oplus \mathbf{N}_2 \in \Gamma)}{z|\iota_i\langle M \rangle \vdash \Gamma} (\oplus, \iota_i) \quad \frac{P_1 \vdash \Gamma, x_1 : \mathbf{P}_1 \quad P_2 \vdash \Gamma, x_2 : \mathbf{P}_2}{\pi_1(x_1).P_1 + \pi_2(x_2).P_2 \vdash \Gamma, \mathbf{P}_1 \& \mathbf{P}_2} \& \\ \\ \frac{N \vdash \Gamma, \mathbf{N} \quad (z : \downarrow \mathbf{N} \in \Gamma)}{z|\downarrow\langle N \rangle \vdash \Gamma} (\downarrow, \downarrow) \quad \frac{P \vdash \Gamma, x : \mathbf{P}}{\uparrow(x).P \vdash \Gamma, \uparrow \mathbf{P}} \uparrow \\ \\ \frac{(z : \mathbf{1} \in \Gamma)}{z|\bar{*} \vdash \Gamma} (\mathbf{1}, \bar{*}) \quad \frac{P \vdash \Gamma}{*.P \vdash \Gamma, \perp} (\perp) \quad \frac{}{\sum a(\vec{x}).\Omega \vdash \Gamma, \top} (\top) \end{array}$$

3.2. Soundness. The inference rules given above are all sound. Namely we have:

Theorem 3.5 (Soundness). *If $D \vdash \mathbf{\Lambda}$ is derivable in the proof system, then $D \models \mathbf{\Lambda}$.*

Proof. By induction on the length of the derivation of $D \vdash \mathbf{\Lambda}$. We have two cases, one for each sort of rule.

(1) Suppose that the last inference rule is

$$\frac{M_1 \vdash \mathbf{\Gamma}, \mathbf{N}_{i_1} \quad \dots \quad M_m \vdash \mathbf{\Gamma}, \mathbf{N}_{i_m} \quad (z : \bar{\alpha} \langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle \in \mathbf{\Gamma})}{z | \bar{\alpha} \langle M_1, \dots, M_m \rangle \vdash \mathbf{\Gamma}} \quad (\bar{\alpha}, \bar{a})$$

where $\mathbf{\Gamma} = x_1 : \mathbf{P}_1, \dots, x_l : \mathbf{P}_l$ and $z : \bar{\alpha} \langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle = x_k : \mathbf{P}_k$ for some $1 \leq k \leq l$.

The induction hypothesis gives us $M_j \models \mathbf{\Gamma}, \mathbf{N}_{i_j}$ for every $1 \leq j \leq m$. By Lemma 2.8 (3), $M'_j := \llbracket M_j[N_1/x_1, \dots, N_l/x_l] \rrbracket \in \mathbf{N}_{i_j}$ for every $N_1 \in \mathbf{P}_1^\perp, \dots, N_l \in \mathbf{P}_l^\perp$ and by Definition 2.9, we have that $x_0 | \bar{\alpha} \langle M'_1, \dots, M'_m \rangle \in \bar{\alpha} \langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle$, that is $x_0 | \bar{\alpha} \langle M'_1, \dots, M'_m \rangle \models x_0 : \bar{\alpha} \langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle$.

Applying Lemma 2.8 (1), we get $x_0 | \bar{\alpha} \langle M_1, \dots, M_m \rangle \models \mathbf{\Gamma}, x_0 : \bar{\alpha} \langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle$ and by Theorem 2.15 (1) we conclude $z | \bar{\alpha} \langle M_1, \dots, M_m \rangle \models \mathbf{\Gamma}$.

(2) Suppose now that the last inference rule is

$$\frac{\{P_a \vdash \mathbf{\Gamma}, \vec{x} : \vec{\mathbf{P}}_a\}_{a(\vec{x}) \in \alpha_0}}{\sum a(\vec{x}). P_a \vdash \mathbf{\Gamma}, \alpha(\mathbf{P}_1, \dots, \mathbf{P}_n)} \quad (\alpha)$$

where $\mathbf{\Gamma} = y_1 : \mathbf{Q}_1, \dots, y_l : \mathbf{Q}_l$ and $\vec{x} : \vec{\mathbf{P}}_a$ stands for $z_{i_1} : \mathbf{P}_{i_1}, \dots, z_{i_m} : \mathbf{P}_{i_m}$. We assume that the variables y_1, \dots, y_l and \vec{x} are disjoint in any premise.

The induction hypothesis gives us $P_a \models \mathbf{\Gamma}, \vec{x} : \vec{\mathbf{P}}_a$ for every $a(x) \in \alpha_0$. By Lemma 2.8 (1), for every $N_1 \in \mathbf{Q}_1^\perp, \dots, N_l \in \mathbf{Q}_l^\perp$, $P'_a := \llbracket P_a[N_1/x_1, \dots, N_l/x_l] \rrbracket \models \vec{x} : \vec{\mathbf{P}}_a$.

Then, we can apply Theorem 2.17 to obtain $\sum a(\vec{x}). P'_a \in \alpha(\mathbf{P}_1, \dots, \mathbf{P}_n)$, that is $\sum a(\vec{x}). P'_a \models \alpha(\mathbf{P}_1, \dots, \mathbf{P}_n)$. Notice that in $\sum a(\vec{x}). P'_a$ the components $b(\vec{y}). P_b$ for $b(\vec{y}) \notin \alpha_0$ can be arbitrary.

We finally apply Lemma 2.8 (3) and conclude $\sum a(\vec{x}). P_a \models \mathbf{\Gamma}, \alpha(\mathbf{P}_1, \dots, \mathbf{P}_n)$. \square

Although our proof system does not include a cut rule officially, the semantics validates it as follows.

Proposition 3.6.

- (1) *If $P \models \mathbf{\Gamma}, z : \mathbf{P}$ and $M \models \mathbf{\Gamma}, \mathbf{P}^\perp$, then $\llbracket P[M/z] \rrbracket \models \mathbf{\Gamma}$.*
- (2) *If $N \models \mathbf{N}, \mathbf{\Gamma}, z : \mathbf{P}$ and $M \models \mathbf{\Gamma}, \mathbf{P}^\perp$, then $\llbracket N[M/z] \rrbracket \models \mathbf{N}, \mathbf{\Gamma}$.*

Proof. Let $\mathbf{\Gamma}$ be $x_1 : \mathbf{P}_1, \dots, x_n : \mathbf{P}_n$, let $K_1 \in \mathbf{P}_1^\perp, \dots, K_n \in \mathbf{P}_n^\perp$ and write \vec{K}/\vec{x} for $K_1/x_1, \dots, K_n/x_n$.

(1) By Lemma 2.8, we have $P' := \llbracket P[\vec{K}/\vec{x}] \rrbracket \models z : \mathbf{P}$ and $M' := \llbracket M[\vec{K}/\vec{x}] \rrbracket \models \mathbf{P}^\perp$, so that $P'[x_0/z] \in \mathbf{P}$ and $M' \in \mathbf{P}^\perp$. Hence, $\llbracket P'[x_0/z][M'/x_0] \rrbracket = \llbracket P'[M'/z] \rrbracket = \mathbf{\boxtimes}$. From this fact and associativity (Theorem 1.12), we can derive $\llbracket \llbracket P[M/z] \rrbracket [\vec{K}/\vec{x}] \rrbracket = \mathbf{\boxtimes}$, which proves $\llbracket P[M/z] \rrbracket \models \mathbf{\Gamma}$.

(2) Let Q be an arbitrary design in \mathbf{N}^\perp . By Lemma 2.8, we obtain $\llbracket Q[N/x_0] \rrbracket \models \mathbf{\Gamma}, z : \mathbf{P}$ and $Q' = \llbracket \llbracket Q[N/x_0] \rrbracket [\vec{K}/\vec{x}] \rrbracket \models z : \mathbf{P}$. On the other side, we have $M' := \llbracket M[\vec{K}/\vec{x}] \rrbracket \models \mathbf{P}^\perp$. From $Q'[x_0/z] \in \mathbf{P}$ and $M' \in \mathbf{P}^\perp$, we obtain $\llbracket Q'[x_0/z][M'/x_0] \rrbracket = \llbracket Q'[M'/z] \rrbracket = \mathbf{\boxtimes}$. From this fact and associativity, we can derive $\llbracket Q[\llbracket N[M/z] \rrbracket [\vec{K}/\vec{x}] / x_0] \rrbracket = \mathbf{\boxtimes}$, which proves $\llbracket N[M/z] \rrbracket \models \mathbf{N}, \mathbf{\Gamma}$. \square

Thanks to the previous proposition, we can naturally strengthen our proof system as follows. First, we consider sequents of the form $D \vdash \mathbf{\Lambda}$ where D is a “proof with cuts” (*i.e.*, a proof in the sense of Definition 3.2 except that the cut-freeness condition is not imposed). Second, we add the following cut rule:

$$\frac{D \vdash \Xi, \Gamma, z : \mathbf{P} \quad N \vdash \Gamma, \mathbf{P}^\perp}{D[N/z] \vdash \Xi, \Gamma} \text{ (cut)}$$

where Ξ is either empty or it consists of a negative logical behaviour \mathbf{N} .

The soundness theorem can be naturally generalized as follows:

Theorem 3.7 (Soundness (with cut rule)). *If $D \vdash \mathbf{\Lambda}$ is derivable in the proof system with the cut rule above, then $\llbracket D \rrbracket \models \mathbf{\Lambda}$.* \square

3.3. Completeness for proofs. Let us finally establish the other direction of Theorem 3.5, namely:

Theorem 3.8 (Completeness for proofs). *A sequent $D \vdash \mathbf{\Lambda}$ is derivable in the proof system if and only if $D \models \mathbf{\Lambda}$.*

In particular, for any positive logical behaviour \mathbf{P} and a proof P , $P \vdash x_0 : \mathbf{P}$ is derivable if and only if $P \in \mathbf{P}$. Similarly for the negative case. \square

Before proving the theorem, let us recall a well-established method for proving Gödel completeness based on proof search (often attributed to Schütte [29]). It proceeds as follows:

- (1) Given an unprovable sequent $\vdash \Gamma$, find an open branch in the cut-free proof search tree.
- (2) From the open branch, build a countermodel M in which $\vdash \Gamma$ is false.

The proof below follows the same line of argument. We can naturally adapt (1) to our setting, since the bottom-up cut-free proof search in our proof system is deterministic in the sense that at most one rule applies at each step. Moreover, it never gets stuck at the negative sequent, since a negative rule is always applicable bottom-up. Adapting (2) is more delicate.

For simplicity, we assume that the sequent $D \vdash \mathbf{\Lambda}$ is positive; the argument below can be easily adapted to the negative case. So, suppose that a positive sequent $P_0 \vdash \Theta_0$ with $\Theta_0 = x_1 : \mathbf{P}_1, \dots, x_n : \mathbf{P}_n$ does not have a derivation. By König’s Lemma, there exists a branch ob in the cut-free proof search tree,

$$\text{ob} = \frac{\frac{\frac{\vdots}{N_1 \vdash \Psi_1}}{P_1 \vdash \Theta_1}}{N_0 \vdash \Psi_0}}{P_0 \vdash \Theta_0},$$

which is *either* finite and has the topmost sequent $P_{max} \vdash \Theta_{max}$ with $max \in \mathbb{N}$ to which no rule applies anymore, *or* infinite. In the latter case, we set $max = \infty$.

Our goal is to build models $\mathcal{M}(x_1) \in \mathbf{P}_1^\perp, \dots, \mathcal{M}(x_n) \in \mathbf{P}_n^\perp$ such that

$$\llbracket P_0[\mathcal{M}(x_1)/x_1, \dots, \mathcal{M}(x_n)/x_n] \rrbracket = \Omega.$$

More generally, we define negative designs

- $\mathcal{M}(i)$ for every $i \geq 0$ ($0 \leq i \leq max$ if $max \in \mathbb{N}$);
- $\mathcal{M}(x)$ for every variable x occurring in the branch.

Below, α and β stand for logical connectives: $\alpha = (\vec{z}, \alpha_0)$, $\beta = (\vec{u}, \beta_0)$.

To define $\mathcal{M}(i)$ we distinguish three cases:

- (i) When $i = \max$ and $P_{\max} = \Omega$, let $\mathcal{M}(\max) := \mathfrak{X}^- (= \sum a(\vec{x}).\mathfrak{X})$.
- (ii) When $i = \max$ and $P_{\max} \neq \Omega$, suppose that $P_{\max} \vdash \Theta_{\max}$ is of the form $z|\vec{c}\langle\vec{M}\rangle \vdash \Gamma, z : \vec{\alpha}\langle\vec{N}\rangle$ but $c(\vec{w}) \notin \alpha_0$ so that the proof search gets stuck. Then let $\mathcal{M}(\max) := \sum_{a(\vec{x}) \in \alpha_0} a(\vec{x}).\mathfrak{X}$. Recall that the partial sum $\mathcal{M}(\max)$ has $c(\vec{w}).\Omega$ as component by our convention.
- (iii) For $i < \max$, suppose that the relevant part of the branch ob is of the form:

$$\text{ob} = \frac{\begin{array}{c} \vdots \\ P_{i+1} \vdash \Theta_{i+1} \\ N_i \vdash \Psi_i \\ P_i \vdash \Theta_i \\ \vdots \end{array}}{\begin{array}{c} \vdots \\ P_{i+1} \vdash \Theta_{i+1}, y_1 : \mathbf{Q}_1, \dots, y_l : \mathbf{Q}_l \\ \sum b(\vec{y}).P_{i+1} \vdash \Theta_i, \mathbf{N}_{i_k} \\ \vdots \end{array}} \frac{(\beta)}{z|\vec{\alpha}\langle M_1, \dots, M_{k-1}, \sum b(\vec{y}).P_{i+1}, M_{k+1}, \dots, M_m \rangle \vdash \Theta_i} (\vec{\alpha}, \vec{a})$$

where Θ_i contains $z : \vec{\alpha}\langle\mathbf{N}_1, \dots, \mathbf{N}_n\rangle$, $a(\vec{x}) \in \alpha_0$ with $\vec{x} = z_{i_1}, \dots, z_{i_m}$, and $\mathbf{N}_{i_k} = \beta(\mathbf{Q}_1, \dots, \mathbf{Q}_s)$, $b(\vec{y}) \in \beta_0$ with $\vec{y} = y_1, \dots, y_l$. Namely, the situation is as follows (to be read bottom-up):

- The head variable of P_i is z , so $z : \vec{\alpha}\langle\mathbf{N}_1, \dots, \mathbf{N}_n\rangle$ is chosen from the context Θ_i and the rule $(\vec{\alpha}, \vec{a})$ is applied. Among m upper sequents, the k th one is taken in the branch.
- $N_i = \sum b(\vec{y}).P_{i+1}$ is negative, and the unique negative behaviour in Ψ_i is $\mathbf{N}_{i_k} = \beta(\mathbf{Q}_1, \dots, \mathbf{Q}_s)$, so the rule (β) is applied. Among the upper sequents (recall that there is one sequent for each action in β_0), the one corresponding to $b(\vec{y}) \in \beta_0$ is taken in the branch.

In this case, we define

$$\mathcal{M}(i) := a(\vec{x}).z_{i_k}|\vec{b}\langle\mathcal{M}(y_1), \dots, \mathcal{M}(y_l)\rangle + \sum_{\alpha_0 \setminus \{a(\vec{x})\}} c(\vec{w}).\mathfrak{X}.$$

Here, the main additive component of $\mathcal{M}(i)$ begins with $a(\vec{x}).z_{i_k}|\vec{b}$ because (1) P_i begins with the positive action \vec{a} , (2) the k th upper sequent is taken in the branch, and (3) the upper sequent corresponding to $b(\vec{y})$ is taken. The other additive components $\sum_{\alpha_0 \setminus \{a(\vec{x})\}} c(\vec{w}).\mathfrak{X}$ are needed to ensure that our countermodel belongs to the behaviour $\vec{\alpha}\langle\mathbf{N}_1, \dots, \mathbf{N}_n\rangle^\perp$ (see Lemma 3.10 (1)).

The subdesigns $\mathcal{M}(y_1), \dots, \mathcal{M}(y_l)$ are given by

$$\mathcal{M}(y) := \bigwedge \{ \mathcal{M}(j) : P_j \text{ has head variable } y \}.$$

Notice that each $\mathcal{M}(j)$ is a negative design, so the above conjunction is a defined operation (in the sense of Definition 1.5 (2)).

We claim that $\mathcal{M}(i)$ is well-defined, because variables \vec{y} are chosen fresh, so do not appear freely below $N_i \vdash \Psi_i$. Hence subdesigns $\mathcal{M}(y_1), \dots, \mathcal{M}(y_l)$ do not have $\mathcal{M}(k)$ with $k \leq i$ as conjunct. Namely, $\mathcal{M}(i)$ depends only on $\mathcal{M}(j)$ with $j > i$. This gives rise to a recursive procedure and $\mathcal{M}(i)$ arises in the limit of the procedure.

Notice also that the set $\{ \mathcal{M}(j) : P_j \text{ has head variable } y \}$ can be empty and in such a case, we have that $\mathcal{M}(y) = \mathfrak{X}^-$.

Remark 3.9. The above is an instance of corecursive definition. It is possible to formally justify it by employing *design generators* developed in [30] (see in particular Theorem 2.12 of [30]). An alternative way is to define $\mathcal{M}(i)$ (and $\mathcal{M}(y)$) as the limit of its finite approximations. Here we briefly outline this latter approach.

We assume that $max = \infty$. The idea is to chop off the branch **ob** at height K , where K is an arbitrary natural number, and define finite approximations $\mathcal{M}^K(i)$ and $\mathcal{M}^K(y)$. Then $\mathcal{M}(i)$ and $\mathcal{M}(y)$ arise as the limit when $K \rightarrow \infty$.

More concretely, given a natural number K , we define $\mathcal{M}^K(i)$ by downward induction from $i = K$ to $i = 0$ as follows:

- When $i = K$, the sequent $P_K \vdash \Theta_K$ is of the form $z|\bar{a}\langle\vec{M}\rangle \vdash \Gamma, z : \bar{a}\langle\vec{N}\rangle$. We let $\mathcal{M}^K(K) := \sum_{a(\vec{x}) \in \alpha_0} a(\vec{x}).\mathfrak{X}$.
- When $i < K$, we proceed as in the case (iii) above. Namely,

$$\begin{aligned} \mathcal{M}^K(i) &:= a(\vec{x}).z_{i_k}|\bar{b}\langle\mathcal{M}^K(y_1), \dots, \mathcal{M}^K(y_l)\rangle + \sum_{\alpha_0 \setminus \{a(\vec{x})\}} c(\vec{w}).\mathfrak{X}, \\ \mathcal{M}^K(y) &:= \bigwedge \{ \mathcal{M}^K(j) : i < j \leq K \text{ and } P_j \text{ has head variable } y \}, \end{aligned}$$

where actions $a(\vec{x})$, \bar{b} and the index i_k are determined as before.

Now observe that the sequence $\{\mathcal{M}^K(y)\}_{K \in \mathbb{N}}$ is “monotone increasing” in the sense that $\mathcal{M}^{K_2}(y)$ has more conjuncts than $\mathcal{M}^{K_1}(y)$ whenever $K_1 < K_2$. The same for $\mathcal{M}^K(i)$ with $i \leq K$. Hence we can naturally obtain the “limits”

$$\mathcal{M}(i) = \lim_{K \rightarrow \infty} \mathcal{M}^K(i), \quad \mathcal{M}(y) = \lim_{K \rightarrow \infty} \mathcal{M}^K(y).$$

This construction ends up with the same as the previous recursive one.

Observe that each $\mathcal{M}(i)$ and $\mathcal{M}(x)$ thus constructed are surely models, *i.e.*, atomic linear designs. Theorem 3.8 is a direct consequence of the following two lemmas.

The first lemma crucially rests on *induction on logical behaviours*, that is an analogue of *induction on formulas*, which lies at the core of logical completeness in many cases.

Lemma 3.10. *For $P_i \vdash \Theta_i$ appearing in the branch **ob** above, suppose that P_i has a head variable z and $z : \mathbf{R} \in \Theta_i$. Then:*

- (1) $\mathcal{M}(i) \in \mathbf{R}^\perp$;
- (2) $\mathcal{M}(z) \in \mathbf{R}^\perp$.

Proof. By induction on the construction of \mathbf{R} .

- (1) Suppose that $i = max$. Since Ω does not have a head variable, the case (i) does not apply. Hence we are in the case (ii), namely $\mathbf{R} = \bar{\alpha}\langle\mathbf{N}_1, \dots, \mathbf{N}_n\rangle$, for some logical connective α and logical behaviours $\mathbf{N}_1, \dots, \mathbf{N}_n$. Thus, $\mathbf{R}^\perp = \alpha(\mathbf{N}_1^\perp, \dots, \mathbf{N}_n^\perp)$, and $\mathcal{M}(max) := \sum_{a(\vec{x}) \in \alpha_0} a(\vec{x}).\mathfrak{X}$.

By internal completeness for negative connectives (Theorem 2.17), we have

$$\sum a(\vec{x}).P_a \in \alpha(\mathbf{N}_1^\perp, \dots, \mathbf{N}_n^\perp) \iff P_a \models \vec{x} : \vec{\mathbf{N}}_a^\perp, \text{ for every } a(\vec{x}) \in \alpha_0,$$

where $\vec{x} = z_{i_1}, \dots, z_{i_m}$ and the expression $\vec{x} : \vec{\mathbf{N}}_a^\perp$ abbreviates the positive context $z_{i_1} : \mathbf{N}_{i_1}^\perp, \dots, z_{i_m} : \mathbf{N}_{i_m}^\perp$. Since $\mathfrak{X} \models \vec{x} : \vec{\mathbf{N}}_a^\perp$ trivially holds for every $a(\vec{x}) \in \alpha_0$, we have $\mathcal{M}(max) \in \alpha(\mathbf{N}_1^\perp, \dots, \mathbf{N}_n^\perp) = \mathbf{R}^\perp$.

When $i < max$, the case (iii) applies. In the same notation, we have that $\mathbf{R} = \bar{\alpha}\langle\mathbf{N}_1, \dots, \mathbf{N}_n\rangle$, $\mathbf{N}_{i_k} = \beta(\mathbf{Q}_1, \dots, \mathbf{Q}_s)$, and

$$\mathcal{M}(i) = a(\vec{x}).z_{i_k}|\bar{b}\langle\mathcal{M}(y_1), \dots, \mathcal{M}(y_l)\rangle + \sum_{\alpha_0 \setminus \{a(\vec{x})\}} c(\vec{w}).\mathfrak{X},$$

where actions $a(\vec{x})$, \bar{b} , the index i_k and the variables y_1, \dots, y_l are determined by the relevant part of the branch **ob** as described above.

By induction hypothesis on (2), we have that $\mathcal{M}(y_1) \in \mathbf{Q}_1^\perp, \dots, \mathcal{M}(y_l) \in \mathbf{Q}_l^\perp$. Hence, $x_0 \bar{b}\langle \mathcal{M}(y_1), \dots, \mathcal{M}(y_l) \rangle \in \bar{\beta}\langle \mathbf{Q}_1^\perp, \dots, \mathbf{Q}_s^\perp \rangle = \mathbf{N}_{i_k}^\perp$. Since $\mathcal{M}(y_1), \dots, \mathcal{M}(y_l)$ are atomic (*i.e.*, closed), we may derive $z_{i_k} \bar{b}\langle \mathcal{M}(y_1), \dots, \mathcal{M}(y_l) \rangle \models \vec{x} : \vec{\mathbf{N}}_a^\perp$. We also have $\mathfrak{X} \models \vec{w} : \vec{\mathbf{N}}_c^\perp$ for every $c(\vec{w}) \in \alpha_0 \setminus \{a(\vec{x})\}$. Hence, by internal completeness again, $\mathcal{M}(i) \in \alpha(\mathbf{N}_1^\perp, \dots, \mathbf{N}_n^\perp) = \mathbf{R}^\perp$.

- (2) It follows from (1) since \mathbf{R}^\perp is a negative logical behaviour and so closed under \bigwedge (Theorem 2.15 (4)). \square

The proof of the next lemma suggests a similarity between the construction of our countermodels and the *Böhm-out* technique (see, *e.g.*, [2]), that constructs a suitable term context in order to visit a specific position in the Böhm tree of a given λ -term.

Recall that the initial sequent of our open branch *ob* is $P_0 \vdash \Theta_0$ with $\Theta_0 = x_1 : \mathbf{P}_1, \dots, x_n : \mathbf{P}_n$, so that $\text{fv}(P_0) \subseteq \{x_1, \dots, x_n\}$. We have:

Lemma 3.11.

$$\llbracket P_0[\mathcal{M}(x_1)/x_1, \dots, \mathcal{M}(x_n)/x_n] \rrbracket = \Omega.$$

Proof. We first prove that there is a reduction sequence

$$P_i[\mathcal{M}(v_1)/v_1, \dots, \mathcal{M}(v_s)/v_s] \multimap^* P_{i+1}[\mathcal{M}(w_1)/w_1, \dots, \mathcal{M}(w_t)/w_t]$$

for any $i < \text{max}$, where v_1, \dots, v_s and w_1, \dots, w_t are the free variables of P_i and P_{i+1} , respectively. Suppose that P_i is as in the case (iii) above, so has the head variable $z \in \{v_1, \dots, v_s\}$. By writing $[\theta]$ for $[\mathcal{M}(v_1)/v_1, \dots, \mathcal{M}(v_s)/v_s]$ and noting that $\mathcal{M}(z)$ is a (defined) conjunction that contains $\mathcal{M}(i) = a(\vec{x}).z_{i_k} \bar{b}\langle \mathcal{M}(y_1), \dots, \mathcal{M}(y_l) \rangle + \sum_{\alpha_0 \setminus \{a(\vec{x})\}} c(\vec{w}).\mathfrak{X}$ as conjunct, we have:

$$\begin{aligned} P_i[\theta] &= \mathcal{M}(z) \mid \bar{a}\langle M_1[\theta], \dots, M_{k-1}[\theta], \sum b(\vec{y}).P_{i+1}[\theta], M_{k+1}[\theta], \dots, M_m[\theta] \rangle \\ &\multimap (\sum b(\vec{y}).P_{i+1}[\theta]) \mid \bar{b}\langle \mathcal{M}(y_1), \dots, \mathcal{M}(y_l) \rangle \wedge \dots \\ &\multimap P_{i+1}[\theta, \mathcal{M}(y_1)/y_1, \dots, \mathcal{M}(y_l)/y_l], \end{aligned}$$

as desired. When $\text{max} = \infty$, we have obtained an infinite reduction sequence from $P_0[\mathcal{M}(x_1)/x_1, \dots, \mathcal{M}(x_n)/x_n]$. Otherwise, $P_0[\mathcal{M}(x_1)/x_1, \dots, \mathcal{M}(x_n)/x_n] \multimap^* P_{\text{max}}[\theta]$, for some substitution $[\theta]$.

In case (i), we have $P_{\text{max}} = P_{\text{max}}[\theta] = \Omega$, while in case (ii), we have $P_{\text{max}} = z \mid \bar{c}\langle \vec{M} \rangle$. So,

$$P_{\text{max}}[\theta] = z \mid \bar{c}\langle \vec{M} \rangle[\theta] = \mathcal{M}(z) \mid \bar{c}\langle \vec{M} \rangle[\theta] \multimap \Omega,$$

because $\mathcal{M}(z)$ contains $\mathcal{M}(\text{max})$ as conjunct, and $\mathcal{M}(\text{max}) = \sum_{a(\vec{x}) \in \alpha_0} a(\vec{x}).\mathfrak{X}$ has $c(\vec{w}).\Omega$ as component. \square

Theorem 3.8 now follows easily. Suppose that $P_0 \vdash x_1 : \mathbf{P}_1, \dots, x_n : \mathbf{P}_n$ is not derivable. Then we obtain models $\mathcal{M}(x_1) \in \mathbf{P}_1^\perp, \dots, \mathcal{M}(x_n) \in \mathbf{P}_n^\perp$ by Lemma 3.10 and $\llbracket P_0[\mathcal{M}(x_1)/x_1, \dots, \mathcal{M}(x_n)/x_n] \rrbracket = \Omega$ by Lemma 3.11. This means that $P_0 \not\vdash x_1 : \mathbf{P}_1, \dots, x_n : \mathbf{P}_n$.

Our explicit construction of the countermodels yields a by-product:

Corollary 3.12 (Downward Löwenheim-Skolem, Finite model property).

- (1) Let P be a proof and \mathbf{P} a logical behaviour. If $P \notin \mathbf{P}$, then there is a countable model $M \in \mathbf{P}^\perp$ (*i.e.*, $\text{ac}^+(M)$ is a countable set) such that $P \not\vdash M$.
- (2) Furthermore, when P is linear, there is a finite and deterministic model $M \in \mathbf{P}^\perp$ such that $P \not\vdash M$. \square

The second statement is due to the observation that when P is linear the positive rule $(\bar{\alpha}, \bar{a})$ can be replaced with a linear variant:

$$\frac{M_1 \vdash \Gamma_1, \mathbf{N}_{i_1} \quad \dots \quad M_m \vdash \Gamma_m, \mathbf{N}_{i_m}}{z|\bar{a}\langle M_1, \dots, M_m \rangle \vdash \Gamma, z : \bar{\alpha}\langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle} (\bar{\alpha}, \bar{a})_{lin},$$

where $\Gamma_1, \dots, \Gamma_m$ are disjoint subsets of Γ . We then immediately see that the proof search tree is always finite, and so is the model $\mathcal{M}(x)$. It is deterministic, since each variable occurs at most once as head variable in a branch so that all conjunctions are at most unary.

4. CONCLUSION AND RELATED WORK

We have presented a Gödel-like completeness theorem for proofs in the framework of ludics, aiming at linking completeness theorems for provability with those for proofs. We have explicitly constructed a countermodel against any failed proof attempt, following Schütte’s idea based on cut-free proof search. Our proof employs König’s lemma and reveals a sharp opposition between finite proofs and infinite models, leading to a clear analogy with Löwenheim-Skolem theorem. Our proof also employs an analogue of the Böhm-out technique [4, 2] (see the proof of Lemma 3.11), though it does not lead to the separation property (Remark 2.4).

In Hyland-Ong game semantics, Player’s innocent strategies most naturally correspond to possibly infinite Böhm trees (see, *e.g.*, [9]). One could of course impose finiteness (or compactness) on them to have correspondence with finite proofs. But it would not lead to an explicit construction of Opponent’s strategies defeating infinite proof attempts. Although finiteness is imposed in [3] too, our current work shows that it is not necessary in ludics.

Our work also highlights the duality:

$$\begin{array}{ccc} \mathbf{proof} & \rightleftharpoons & \mathbf{model} \\ \textit{deterministic, nonlinear} & & \textit{nondeterministic, linear} \end{array}$$

The principle is that *when proofs admit contraction, models have to be nondeterministic* (whereas they do not have to be nonlinear).

A similar situation arises in some variants of λ -calculus and linear logic, when one proves the separation property.

We mention [12], where the authors add a nondeterministic choice operator and a numeral system to the pure λ -calculus in order to internally (interactively) discriminate two pure λ -terms that have different Böhm trees. However, in contrast to our work, the nondeterminism needed for their purpose is of existential nature: a term converges if at least one of the possible reduction sequences starting from it terminates.

In [27], the separation property for differential interaction nets [14] is proven. A key point is that the exponential modalities in differential interaction nets are more “symmetrical” than in linear logic. In our setting, the symmetry shows up between nonlinearity and nondeterministic conjunctions (*i.e.*, nonuniform elements). It is typically found in Theorem 2.15, which reveals a tight connection between duplicability of positive logical behaviours and closure under nondeterministic conjunctions of negative logical behaviours. Similar nonuniform structures naturally arise in various semantical models based on coherence spaces and games, such as finiteness spaces [13], indexed linear logic and nonuniform coherence spaces [6], nonuniform hypercoherences [5], and asynchronous games [28] (see also [3]).

For future work, we plan to extend our setting by enriching the proof system with propositional variables, second order quantifiers and nonlogical axioms. By moving to the second order setting, we hope to give an *interactive* account to Gödel's incompleteness theorems as well.

ACKNOWLEDGEMENT

We are deeply indebted to Pierre-Louis Curien, who gave us a lot of useful comments. Our thanks are also due to the anonymous referees.

REFERENCES

- [1] Andreoli, J.-M.: Logic Programming with Focusing Proofs in Linear Logic. *J. Log. Comput.* **2**(3) (1992) 297–347.
- [2] Barendregt, H. P.: The lambda calculus: its syntax and semantics. North-Holland (1981).
- [3] Basaldella, M., Faggian, C.: Ludics with repetition (exponentials, interactive types and completeness). In: *LICS*. (2009) 375–384.
- [4] Böhm, C.: Alcune proprietà delle forme $\beta - \eta$ -normali nel $\lambda - K$ -calcolo. *Publicazioni dell'Istituto per le Applicazioni del Calcolo* **696** (1968).
- [5] Boudes, P.: Non-Uniform Hypercoherences. *Electr. Notes Theor. Comput. Sci.* **69** (2002) 62–82.
- [6] Bucciarelli, A., Ehrhard, T.: On phase semantics and denotational semantics: the exponentials. *Ann. Pure Appl. Logic* **109**(3) (2001) : 205–241.
- [7] Curien, P.-L.: Abstract Böhm trees. *Math. Struct. in Comp. Sci.* **8**(6) (1998) 559–591.
- [8] Curien, P.-L.: Introduction to linear logic and ludics, part II. *Advances in Mathematics (China)* **35**(1) (2006) 1–44.
- [9] Curien, P.-L.: Notes on game semantics. Manuscript (2006).
- [10] Curien, P.-L., Herbelin, H.: Abstract machines for dialogue games. *Panoramas et Synthèses* **27** (2009) 231–275.
- [11] Curien, P.-L., Munch-Maccagnoni, G.: The duality of computation under focus. In : *Proc. of IFIP TCS*. (2010).
- [12] Dezani-Ciancaglini, M., Intrigila, B., Venturini-Zilli, M.: Böhm's theorem for Böhm trees. In: *ICTCS'98*. (1998) 1–23.
- [13] Ehrhard, T.: Finiteness spaces. *Math. Struct. in Comp. Sci.* **15**(4) (2005) 615–646.
- [14] Ehrhard, T., Regnier, L.: Differential interaction nets. *Theor. Comput. Sci.* **364**(2) (2006) 166–195.
- [15] Faggian, C.: Travelling on designs. In: *CSL*. (2002) 427–441.
- [16] Faggian, C.: Interactive observability in ludics: The geometry of tests. *Theor. Comput. Sci.* **350**(2) (2006) 213–233.
- [17] Faggian, C., Piccolo, M.: Ludics is a model for the finitary linear pi-calculus. In: *TLCA*. (2007) 148–162.
- [18] Faggian, C., Piccolo, M.: Partial Orders, Event Structures, and Linear Strategies. In: *TLCA*. (2009) 95–111.
- [19] Girard, J.-Y.: On the meaning of logical rules I: syntax vs. semantics. In Berger, U., Schwichtenberg, H., eds.: *Computational Logic*. Heidelberg Springer-Verlag (1999) 215–272.
- [20] Girard, J.-Y.: On the meaning of logical rules II: multiplicatives and additives. *Foundation of Secure Computation*, Berger and Schwichtenberg eds (2000) 183–212.
- [21] Girard, J.-Y.: Locus solum: From the rules of logic to the logic of rules. *Math. Struct. in Comp. Sci.* **11**(3) (2001) 301–506.
- [22] Hyland, J.M.E., Ong, C.H.L.: On full abstraction for PCF: I, II, and III. *Inf. Comput.* **163**(2) (2000) 285–408.
- [23] Lafont, Y.: The finite model property for various fragments of linear logic. *J. Symb. Log.* **62**(4) (1997) 1202–1208.
- [24] Laurent, O.: Étude de la polarization en logique. PhD thesis, Univ. Aix-Marseille II (2002).
- [25] Laurent, O.: Polarized games. *Ann. Pure Appl. Logic* **130**(1-3) (2004) 79–123.
- [26] Maurel, F.: Un cadre quantitatif pour la Ludique. PhD Thesis, Univ. Paris VII (2004).

- [27] Mazza, D., Pagani, M.: The separation theorem for differential interaction nets. In: LPAR. (2007) 393–407.
- [28] Melliès, P.-A.: Asynchronous games 2: The true concurrency of innocence. *Theor. Comput. Sci.* **358**(2-3) (2006) 200–228.
- [29] Schütte, K.: Ein System des Verknüpfenden Schliessens. *Archiv. Math. Logic Grundlagent.* **2** (1956) 55–67.
- [30] Terui, K.: Computational ludics. (2008) To appear in *Theor. Comput. Sci.*

APPENDIX A. CORRESPONDENCE WITH POLARIZED LINEAR LOGIC

In this appendix, we show a correspondence between the proof system for ludics introduced in 3.1 and the constant-only propositional fragment of *polarized linear logic* **LLP** [25]. This will ensure that our proof system is rich enough to capture a constructive variant of constant-only propositional classical logic.

A.1. Syntax of LLP. We recall the syntax of the constant-only propositional fragment of **LLP**. The *formulas* are split into positive and negative ones and generated by the following grammar:

$$\begin{aligned} P &::= 0 \mid 1 \mid P \otimes P \mid P \oplus P \mid !N, \\ N &::= \top \mid \perp \mid N \wp N \mid N \& N \mid ?P. \end{aligned}$$

The linear negation is defined in the usual way. A *sequent* is of the form $\vdash \Gamma$ with Γ a multiset of formulas. The inference rules of **LLP** are given below:

$$\begin{array}{c} \frac{}{\vdash \Gamma, \top} \qquad \frac{}{\vdash 1} \qquad \frac{\vdash \Gamma}{\vdash \Gamma, \perp} \qquad \frac{\vdash \Gamma, P \quad \vdash \Delta, Q}{\vdash \Gamma, \Delta, P \otimes Q} \\ \\ \frac{\vdash \Gamma, P_i}{\vdash \Gamma, P_1 \oplus P_2} \qquad \frac{\vdash \Gamma, N, M}{\vdash \Gamma, N \wp M} \qquad \frac{\vdash \Gamma, N \quad \vdash \Gamma, M}{\vdash \Gamma, N \& M} \qquad \frac{\vdash \mathcal{N}, N}{\vdash \mathcal{N}, !N} \\ \\ \frac{\vdash \Gamma, P}{\vdash \Gamma, ?P} \qquad \frac{\vdash \Gamma}{\vdash \Gamma, N} \qquad \frac{\vdash \Gamma, N, N}{\vdash \Gamma, N} \qquad \frac{\vdash \Gamma, N \quad \vdash \Delta, N^\perp}{\vdash \Gamma, \Delta} \end{array}$$

where:

- in the \top -rule above Γ contains at most one positive formula;
- \mathcal{N} denotes a context consisting of negative formulas only.

In [24] it is proven that if $\vdash \Gamma$ is provable in **LLP**, then Γ contains at most one positive formula. Notice that it is strictly opposite to the ludics discipline [21]; in the latter, any sequent contains at most one *negative* behaviour. To resolve this mismatch, we modify **LLP** in several steps, making it closer to the ludics discipline.

Precisely, in Section A.2 we introduce the concept of strict sequent which leads us to the formulation of syntectic connectives in **LLP** (Section A.3). In Section A.4, we give an embedding of **LLP** with synthetic connectives into the proof system of ludics we gave in Section 3.1. Finally, in Section A.5 we give a converse embedding of the ludics proof system into **LLP**.

A.2. Restriction to strict derivations. We call a sequent of **LLP** *strict* if it is of the form $\vdash ?\Gamma, D$, where D is an arbitrary formula. In particular, $\vdash ?\Gamma$ is strict. We modify the inference rules as follows:

- Structural rules are made implicit by absorbing weakening and contraction into logical inference rules.
- The rules for positive connectives and the $?$ -dereliction rule are restricted to strict sequents.
- The cut rule is omitted.

We thus obtain the following inference rules:

$$\begin{array}{c}
\frac{}{\vdash \Gamma, \top} \qquad \frac{}{\vdash ?\Gamma, \perp} \qquad \frac{\vdash \Gamma}{\vdash \Gamma, \perp} \qquad \frac{\vdash ?\Gamma, P \quad \vdash ?\Gamma, Q}{\vdash ?\Gamma, P \otimes Q} \\
\\
\frac{\vdash ?\Gamma, P_i}{\vdash ?\Gamma, P_1 \oplus P_2} \qquad \frac{\vdash \Gamma, N, M}{\vdash \Gamma, N \wp M} \qquad \frac{\vdash \Gamma, N \quad \vdash \Gamma, M}{\vdash \Gamma, N \& M} \qquad \frac{\vdash ?\Gamma, N}{\vdash ?\Gamma, !N} \\
\\
\frac{\vdash ?\Gamma, P \quad (?P \in ?\Gamma)}{\vdash ?\Gamma}
\end{array}$$

We call the resulting proof system **LLP**_{str}.

Notice that a derivation of a strict sequent in **LLP**_{str} may involve sequents which are not strict. For instance, consider:

$$\frac{\frac{\frac{}{\vdash ?\Gamma, \top}}{\vdash ?\Gamma, \top, \perp}}{\vdash ?\Gamma, \top \wp \perp}$$

The following property can be easily verified by taking into account the invertibility of negative rules and the focalization property of positive rules [1].

Lemma A.1. *A strict sequent is provable in **LLP** if and only if it is provable in **LLP**_{str}.* \square

Strict sequents will play a crucial role for the correspondence between **LLP** and the proof system for ludics (Theorem A.4). The intuition, which we will formalize later, is that a strict sequent $\vdash ?P_1, \dots, ?P_n, D$ can be thought of as a sequent of the proof system of ludics (omitting the information about designs) of the form $\vdash \mathbf{P}_1, \dots, \mathbf{P}_n, \mathbf{D}$.

On the other hand, strict derivations serve as intermediate step to define synthetic connectives and the proof system **LLP**_{syn} we give in the next section.

A.3. Synthetic connectives. Any derivation of a strict sequent in **LLP**_{str} can be decomposed into subderivations of the following forms:

(i) Positive subderivation:

$$\frac{\vdash ?\Gamma, N_{i_1} \quad \dots \quad \vdash ?\Gamma, N_{i_m}}{\vdash ?\Gamma, P(N_1, \dots, N_n)}$$

that consists of positive inference rules only, where $P(N_1, \dots, N_n)$ is a positive formula obtained from N_1, \dots, N_n by applying positive connectives, and $i_1, \dots, i_m \in \{1, \dots, n\}$.

For instance, if $P(N_1, \dots, N_n)$ is of the form $1 \otimes (!N \otimes (!M \oplus !L))$, there are two positive subderivations with conclusion $\vdash ?\Gamma, 1 \otimes (!N \otimes (!M \oplus !L))$:

$$\frac{\frac{\frac{\frac{\vdash ?\Gamma, 1}{\vdash ?\Gamma, 1}}{\vdash ?\Gamma, !N} \quad \frac{\frac{\frac{\vdash ?\Gamma, M}{\vdash ?\Gamma, !M}}{\vdash ?\Gamma, !M \oplus !L}}{\vdash ?\Gamma, !N \otimes (!M \oplus !L)}}{\vdash ?\Gamma, 1 \otimes (!N \otimes (!M \oplus !L))}}{\vdash ?\Gamma, 1 \otimes (!N \otimes (!M \oplus !L))}} \quad \frac{\frac{\frac{\frac{\vdash ?\Gamma, L}{\vdash ?\Gamma, !L}}{\vdash ?\Gamma, !M \oplus !L}}{\vdash ?\Gamma, !N \otimes (!M \oplus !L)}}{\vdash ?\Gamma, 1 \otimes (!N \otimes (!M \oplus !L))}}{\vdash ?\Gamma, 1 \otimes (!N \otimes (!M \oplus !L))}}$$

(ii) Negative subderivation:

$$\frac{\vdash ?\Gamma, ?\vec{P}_1 \quad \dots \quad \vdash ?\Gamma, ?\vec{P}_k}{\vdash ?\Gamma, N(P_1, \dots, P_n)}$$

that consists of negative inference rules only, where $N(P_1, \dots, P_n)$ is a negative formula obtained from P_1, \dots, P_n by applying negative connectives, and $\vec{P}_1, \dots, \vec{P}_k$ consist of formulas in $\{P_1, \dots, P_n\}$. For instance, if $N(P_1, \dots, P_n)$ is of the form $\perp \wp (?P \wp (?Q \& ?R))$, then there is (essentially) one negative subderivation with conclusion $\vdash ?\Gamma, \perp \wp (?P \wp (?Q \& ?R))$:

$$\frac{\frac{\frac{\frac{\frac{\vdash ?\Gamma, ?P, ?Q \quad \vdash ?\Gamma, ?P, ?R}{\vdash ?\Gamma, ?P, ?Q \& ?R}}{\vdash ?\Gamma, ?P \wp (?Q \& ?R)}}{\vdash ?\Gamma, \perp, ?P \wp (?Q \& ?R)}}{\vdash ?\Gamma, \perp \wp (?P \wp (?Q \& ?R))}}{\vdash ?\Gamma, \perp \wp (?P \wp (?Q \& ?R))}}$$

(iii) $?$ -dereliction:

$$\frac{\vdash ?\Gamma, P \quad (?P \in ?\Gamma)}{\vdash ?\Gamma}$$

Notice that the premises and conclusion of each subderivation are assumed to be strict sequents; one can easily check that it is always the case in any derivation of a strict sequent in \mathbf{LLP}_{str} .

The above decomposition motivates us to cluster the logical connectives of the same polarity into *synthetic connectives* (cf. [20]). Consider the expressions finitely generated by:

$$\begin{array}{l} \mathbf{p} ::= \mathbf{0} \mid \mathbf{1} \mid \mathbf{p} \otimes \mathbf{p} \mid \mathbf{p} \oplus \mathbf{p} \mid !x, \\ \mathbf{n} ::= \top \mid \perp \mid \mathbf{n} \wp \mathbf{n} \mid \mathbf{n} \& \mathbf{n} \mid ?x, \end{array}$$

where x ranges over the set of variables.

We write $\text{var}(\mathbf{p})$ (resp. $\text{var}(\mathbf{n})$) to denote the set of variables occurring in \mathbf{p} (resp. \mathbf{n}). \mathbf{p} is a *positive synthetic connective* if for every subexpression of \mathbf{p} of the form $\mathbf{p}_1 \otimes \mathbf{p}_2$, $\text{var}(\mathbf{p}_1)$ and $\text{var}(\mathbf{p}_2)$ are disjoint. For instance, $!x \otimes (!y \oplus !y)$ is a positive synthetic connective while $!x \oplus (!y \otimes !y)$ is not. Likewise, \mathbf{n} is a *negative synthetic connective* if for every subexpression of \mathbf{n} of the form $\mathbf{n}_1 \wp \mathbf{n}_2$, $\text{var}(\mathbf{n}_1)$ and $\text{var}(\mathbf{n}_2)$ are disjoint. This condition is needed when we translate synthetic connectives to logical connectives of ludics.

We indicate the variables occurring in \mathbf{p} by writing $\mathbf{p} = \mathbf{p}(x_1, \dots, x_n)$, and similarly for \mathbf{n} . Given a negative synthetic connective \mathbf{n} , its *dual* \mathbf{n}^d is obtained by replacing \top with $\mathbf{0}$, \perp with $\mathbf{1}$, \wp with \otimes , $\&$ with \oplus , and $?$ with $!$ respectively, in each occurrence of symbol. \mathbf{p}^d

is similarly defined.

The *formulas* of **LLP** are then redefined inductively as follows:

$$\begin{aligned} \mathbf{P} &::= \mathbf{p}(\mathbf{N}_1, \dots, \mathbf{N}_n), \\ \mathbf{N} &::= \mathbf{n}(\mathbf{P}_1, \dots, \mathbf{P}_n), \end{aligned}$$

where $\mathbf{p}(\mathbf{N}_1, \dots, \mathbf{N}_n)$ is obtained from $\mathbf{p} = \mathbf{p}(x_1, \dots, x_n)$ by substituting \mathbf{N}_i for x_i ($1 \leq i \leq n$). Notice that when $n = 0$, \mathbf{P} can be any combination of 0 and 1 using \otimes and \oplus .

To each positive synthetic connective $\mathbf{p}(x_1, \dots, x_n)$, we can naturally associate a set of inference rules as follows. Consider all possible positive subderivations with conclusion $\vdash ?\Gamma, \mathbf{p}(\mathbf{N}_1, \dots, \mathbf{N}_n)$ in the sense of (i) above. To each such derivation

$$\begin{array}{c} \vdash ?\Gamma, \mathbf{N}_{i_1} \quad \dots \quad \vdash ?\Gamma, \mathbf{N}_{i_m} \\ \vdots \\ \vdash ?\Gamma, \mathbf{P}(\mathbf{N}_1, \dots, \mathbf{N}_n) \end{array}$$

we associate an inference rule:

$$\frac{\vdash ?\Gamma, \mathbf{N}_{i_1} \quad \dots \quad \vdash ?\Gamma, \mathbf{N}_{i_m}}{\vdash ?\Gamma, \mathbf{P}(\mathbf{N}_1, \dots, \mathbf{N}_n)}$$

For instance, to $\mathbf{p}(x, y, z) = 1 \otimes (!x \otimes (!y \oplus !z))$, we associate two inference rules:

$$\frac{\vdash ?\Gamma, \mathbf{N} \quad \vdash ?\Gamma, \mathbf{M}}{\vdash ?\Gamma, \mathbf{p}(\mathbf{N}, \mathbf{M}, \mathbf{L})} \quad \frac{\vdash ?\Gamma, \mathbf{N} \quad \vdash ?\Gamma, \mathbf{L}}{\vdash ?\Gamma, \mathbf{p}(\mathbf{N}, \mathbf{M}, \mathbf{L})}$$

Likewise, each negative synthetic connective $\mathbf{n}(x_1, \dots, x_n)$ comes equipped with a unique inference rule derived from the negative subderivation with conclusion $\vdash ?\Gamma, \mathbf{n}(\mathbf{P}_1, \dots, \mathbf{P}_n)$ (see (ii) above). For instance, $\mathbf{n}(x, y, z) = \perp \wp (?x \wp (?y \& ?z))$ is equipped with:

$$\frac{\vdash ?\Gamma, ?\mathbf{P}, ?\mathbf{Q} \quad \vdash ?\Gamma, ?\mathbf{P}, ?\mathbf{R}}{\vdash ?\Gamma, \mathbf{n}(\mathbf{P}, \mathbf{Q}, \mathbf{R})}$$

Observe the asymmetry between the positive and negative cases here; in the negative case, we leave the $?$ -formulas $?\mathbf{P}, ?\mathbf{Q}, ?\mathbf{R}$ in the premises. These formulas are to be dealt with by the $?$ -dereliction rule.

We thus consider proof system **LLP**_{syn} that consists of three types of inference rules:

$$\frac{\vdash ?\Gamma, \mathbf{N}_{i_1} \quad \dots \quad \vdash ?\Gamma, \mathbf{N}_{i_m}}{\vdash ?\Gamma, \mathbf{p}(\mathbf{N}_1, \dots, \mathbf{N}_n)} \quad \frac{\vdash ?\Gamma, ?\vec{\mathbf{P}}_1 \quad \dots \quad \vdash ?\Gamma, ?\vec{\mathbf{P}}_k}{\vdash ?\Gamma, \mathbf{n}(\mathbf{P}_1, \dots, \mathbf{P}_n)} \quad \frac{\vdash ?\Gamma, \mathbf{P} \quad (? \mathbf{P} \in ?\Gamma)}{\vdash ?\Gamma}$$

In view of the decomposition of **LLP**_{str} derivations, we obviously have:

Lemma A.2. *A strict sequent is provable in **LLP**_{str} if and only if it is provable in **LLP**_{syn}.* \square

A.4. Relating to the ludics proof system. Let us now move on to the proof system for ludics described in 3.1. We assume that the signature \mathcal{A} is rich enough to interpret **LLP**:

- \mathcal{A} contains a nullary name $*$ and a unary name \uparrow .
- If \mathcal{A} contains an n -ary name a and an m -ary name b , it also contains n -ary names $\pi_1 a$, $\pi_2 a$ and an $(n + m)$ -ary name $a \wp b$ (cf. Example 2.14).

Given a negative synthetic connective n , we inductively associate a set \mathbf{n}_0^\bullet of negative actions of ludics as follows:

$$\begin{aligned} \top_0^\bullet &= \emptyset, \\ \perp_0^\bullet &= \{*\}, \\ ?x_0^\bullet &= \{\uparrow(x)\}, \\ (n \wp m)_0^\bullet &= \{a \wp b(\vec{x}, \vec{y}) : a(\vec{x}) \in \mathbf{n}_0^\bullet, b(\vec{y}) \in \mathbf{m}_0^\bullet\}, \\ (n \& m)_0^\bullet &= \{\pi_1 a(\vec{x}) : a(\vec{x}) \in \mathbf{n}_0^\bullet\} \cup \{\pi_2 b(\vec{y}) : b(\vec{y}) \in \mathbf{m}_0^\bullet\}. \end{aligned}$$

Notice that when $a(\vec{x}) \in \mathbf{n}_0^\bullet$, the variables \vec{x} occur in n . Hence $a \wp b(\vec{x}, \vec{y})$ above is certainly a negative action, since \vec{x} and \vec{y} are disjoint sequences due to the definition of negative synthetic connective. We finally let $\mathbf{n}^\bullet = (\vec{z}, \mathbf{n}_0^\bullet)$, where \vec{z} lists the variables occurring in n . A positive synthetic connective p is interpreted by $\mathbf{p}^\bullet = \mathbf{p}^{d^\bullet}$.

For instance, when $\mathbf{p}(x, y, z) = 1 \otimes (!x \otimes (!y \oplus !z))$ and $\mathbf{n}(x, y, z) = \perp \wp (?x \wp (?y \& ?z))$, we have $\mathbf{p}^\bullet = \mathbf{n}^\bullet = (x, y, z, \mathbf{n}_0^\bullet)$ with

$$\mathbf{n}_0^\bullet = \{*\wp(\uparrow\wp(\pi_1\uparrow))(x, y), *\wp(\uparrow\wp(\pi_2\uparrow))(x, z)\}.$$

This induces a polarity-preserving translation from the formulas of **LLP** to the logical behaviours of ludics:

$$\begin{aligned} \mathbf{n}(\mathbf{P}_1, \dots, \mathbf{P}_n)^\bullet &:= \mathbf{n}^\bullet(\mathbf{P}_1^\bullet, \dots, \mathbf{P}_n^\bullet), \\ \mathbf{p}(\mathbf{N}_1, \dots, \mathbf{N}_n)^\bullet &:= \overline{\mathbf{p}^\bullet} \langle \mathbf{N}_1^\bullet, \dots, \mathbf{N}_n^\bullet \rangle. \end{aligned}$$

To establish a connection with **LLP**, we simplify the proof system of 3.1 by taking its *skeleton*, namely by omitting all information about designs. The resulting proof system, which we call **L**, consists of two sorts of inference rules:

$$\frac{\vdash \Gamma, \mathbf{N}_{i_1} \quad \dots \quad \vdash \Gamma, \mathbf{N}_{i_m} \quad (\overline{\alpha} \langle \mathbf{N}_1, \dots, \mathbf{N}_n \rangle \in \Gamma)}{\vdash \Gamma} \quad (\overline{\alpha}, \overline{a}) \quad \frac{\{\vdash \Gamma, \mathbf{P}_{i_1}, \dots, \mathbf{P}_{i_m}\}_{a(\vec{x}) \in \alpha_0}}{\vdash \Gamma, \alpha(\mathbf{P}_1, \dots, \mathbf{P}_n)} \quad (\alpha)$$

where $\alpha = (\vec{z}, \alpha_0)$, $\vec{z} = z_1, \dots, z_n$, $a(\vec{x}) \in \alpha_0$ and the indices $i_1, \dots, i_m \in \{1, \dots, n\}$ are determined by the variables $\vec{x} = z_{i_1}, \dots, z_{i_m}$.

For instance, when $\mathbf{p}(x, y, z) = 1 \otimes (!x \otimes (!y \oplus !z))$ and $\mathbf{n}(x, y, z) = \perp \wp (?x \wp (?y \& ?z))$, we have the following inference rules for \mathbf{p}^\bullet and \mathbf{n}^\bullet :

$$\begin{aligned} \frac{\vdash \Gamma, \mathbf{N} \quad \vdash \Gamma, \mathbf{M} \quad (\overline{\mathbf{p}^\bullet} \langle \mathbf{N}, \mathbf{M}, \mathbf{L} \rangle \in \Gamma)}{\vdash \Gamma} \quad \frac{\vdash \Gamma, \mathbf{N} \quad \vdash \Gamma, \mathbf{L} \quad (\overline{\mathbf{p}^\bullet} \langle \mathbf{N}, \mathbf{M}, \mathbf{L} \rangle \in \Gamma)}{\vdash \Gamma} \\ \frac{\vdash \Gamma, \mathbf{P}, \mathbf{Q} \quad \vdash \Gamma, \mathbf{P}, \mathbf{R}}{\vdash \Gamma, \mathbf{n}^\bullet(\mathbf{P}, \mathbf{Q}, \mathbf{R})} \end{aligned}$$

It is now straightforward to verify:

Lemma A.3. *A strict sequent $\vdash ?\Gamma, D$ is derivable in **LLP**_{syn} if and only if $\vdash \Gamma^\bullet, D^\bullet$ is derivable in **L**. \square*

We therefore obtain:

Theorem A.4. *A strict sequent $\vdash ?\Gamma, D$ is derivable in **LLP** if and only if $\vdash \Gamma^\bullet, D^\bullet$ is derivable in **L**. \square*

One can annotate derivations in \mathbf{L} with designs as in Section 3.1. Therefore the above theorem means that ludics designs can be used as term syntax for \mathbf{LLP} , as far as strict sequents and derivations are concerned (although we have to verify carefully that the translation preserves the reduction relation).

A.5. From ludics to LLP. It is also possible to give a converse translation from the logical behaviours of ludics to the formulas of \mathbf{LLP} . To do so, we proceed as follows (*cf.* Example 2.10):

- to each action $a(x_1, \dots, x_m)$, we associate the synthetic connective $a(x_1, \dots, x_m)^\circ := ?x_1 \wp \dots \wp ?x_m$ ($a()^\circ := \perp$, if a is nullary);
- to each logical connective $\alpha = (\vec{z}, \{a_1(\vec{x}_1), \dots, a_k(\vec{x}_k)\})$, we associate the synthetic connective $\alpha^\circ := a_1(\vec{x}_1)^\circ \& \dots \& a_k(\vec{x}_k)^\circ$ ($\alpha^\circ := \top$, if $k = 0$);
- to each logical behaviour, we associate the formula of \mathbf{LLP}

$$\begin{aligned} \alpha(\mathbf{P}_1, \dots, \mathbf{P}_n)^\circ &:= \alpha^\circ(\mathbf{P}_1^\circ, \dots, \mathbf{P}_n^\circ), \\ \bar{\alpha}(\mathbf{N}_1, \dots, \mathbf{N}_n)^\circ &:= \alpha^{\circ d}(\mathbf{N}_1^\circ, \dots, \mathbf{N}_n^\circ); \end{aligned}$$

- to each positive context $\Gamma = \mathbf{P}_1, \dots, \mathbf{P}_n$ of system \mathbf{L} , we associate the multiset $\Gamma^\circ := ?\mathbf{P}_1^\circ, \dots, ?\mathbf{P}_n^\circ$ of formulas of \mathbf{LLP} ;
- to each negative context Γ, \mathbf{N} of system \mathbf{L} , we associate the multiset of formulas $(\Gamma, \mathbf{N})^\circ := \Gamma^\circ, \mathbf{N}^\circ$.

It is routine to define an isomorphism between \mathbf{D} and $\mathbf{D}^{\bullet\circ}$ (resp. between \mathbf{D} and $\mathbf{D}^{\bullet\bullet}$) in some natural sense. Moreover, the translation of a sequent of system \mathbf{L} always results in a strict sequent of \mathbf{LLP} . We therefore conclude by Theorem A.4:

Theorem A.5. *A sequent $\vdash \Lambda$ is derivable in proof system \mathbf{L} if and only if $\vdash \Lambda^\circ$ is derivable in \mathbf{LLP} . \square*