Designs, Disputes and Strategies

Claudia Faggian and Martin Hyland

DPMMS – University of Cambridge

Abstract. Ludics has been proposed by Girard as an abstract general approach to proof theory. We explain how its basic notions correspond to those of the "innocent strategy" appraoch to Games Semantics, and thus establish a clear connection between the two subjects.

1 Introduction

Interaction has become an important notion both in theoretical computer science and in proof theory. From the computational point of view, when running an application the result of computation (if there is any) is not necessarily the most interesting aspect. The dynamics, the process of computation itself may play the central role. Moreover, composition of programs is in general a rich two-directions process, which entails communication and exchanges between the components. A paradigm of *computation as interaction* underlies several models of computation. This paradigm is particularly significant today, since for reactive systems, the process of interaction rather than any final result is what is at issue.

Important progress in logic has also lead to interactive and dynamical models. Major examples are Geometry of Interaction and Games Semantics. The Geometry of Interaction [5], which arose from Linear Logic, interprets normalization (computation) as a flow of information circulating around a net. Games Semantics interprets computation as a dialog between two parties, the program (player) and the environment (opponent), each one following its own "strategy". Games Semantics (see [2] for a survey) has been both an important development in logic, and a successful approach to the semantics of programming language. The strength of these models is to capture the dynamical aspects of computation, so as to take into account both qualitative (correctness) and quantitative (efficiency) aspects of programming languages. Ludics, recently introduced by Girard in [6], is a further step in this development, the fundamental notion in the theory being that of interaction.

The basic objects of Ludics are *designs,* which are both (i) an abstraction of formal proofs and (ii) a concretion of their semantical interpretation. A design can be described as the skeleton of a sequent calculus derivation, where we do not manipulate formulas, but their locations (the addresses where the formulas are stored). A design can also be presented in a very natural way as the collection of its possible interactions. Our paper focuses on this presentation . An advantage of the approach we follow is to establish a bridge with the notions of Game Semantics, in particular with HON Games [7], [9]. In fact, we are going to make precise the following correspondences:

J. Bradfield (Ed.): CSL 2002, LNCS 2471, pp. 442–457, 2002.

actions – moves
disputes – plays
chronicles – views
designs – innocent strategies

The crucial correspondence is

"view - chronicle - sequent calculus branch."

(In what follows one should keep in mind the concrete interpretation of a chronicle as a branch in a sequent calculus derivation, a design being the "skeleton" of a sequent calculus derivation.)

The correspondence view-chronicle is the key to translating between Ludics and Games Semantics settings. We expect to be able to transfer experiences and techniques between the two settings.

2 Ludics: Designs

2.1 The Universe of Proofs

The program of Ludics is to overcome the distinction between syntax (the formal system) on one side and semantics (its interpretation) on the other side. Rather then having two separate worlds, proofs are interpreted via proofs. To determine and test properties, a *proof* of A should be tested with *proofs* of $A^{\perp}$. Ludics provides a setting in which proofs of A interact with proofs of $A^{\perp}$; to this end, it generalizes the notion of proof.

A proof should be thought in the sense of "proof search" or "proof construction": we start from the conclusion, and guess a last rule, then the rule above. What if we cannot apply any rule? A new rule is introduced, called daimon:

$$\frac{}{\vdash \Gamma}\ \dagger$$

Such a rule allow us to assume any conclusion, without providing a justification.

The syntax of proofs is not the sequent calculus, but a more abstract formalism, close to Böhm trees, and called "design". The proofs do not manipulate formulas, but addresses. These are sequences of natural number, which can be thought of as the address in the memory where the formula is stored.

2.2 Designs

Let us first give an intuition of what is a design. This should be enough to follow the rest of the paper. At the end of the section we recall the formal definitions. We will not really enter in the details of the logical calculus associated to designs, which is a focalized version of second order multiplicative-additive Linear Logic ($\mathbf{M}ALL_2$).

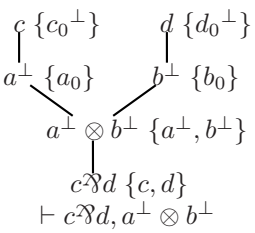
Designs capture the geometrical structure of sequent calculus derivations. The simplest way to introduce designs is to start from the sequent calculus. Let

us consider the following derivation, where the rules are labelled by the active formula and the subformulas which appear in the premises[1]: for example, $\oplus_L$ would be labelled as $(a \oplus b, \{a\})$.

$$\dfrac{\dfrac{\dfrac{\vdash a_0, {c_0}^\perp}{\vdash a_0, c}\,(c, \{{c_0}^\perp\})}{\vdash a^\perp, c}\,(a, \{a_0\}) \qquad \dfrac{\dfrac{\vdash b_0, {d_0}^\perp}{\vdash b_0, d}\,(d, \{{d_0}^\perp\})}{\vdash b^\perp, d}\,(b^\perp, \{b_0\})}{\dfrac{\vdash c, d, a^\perp \otimes b^\perp}{\vdash c \parr d, a^\perp \otimes b^\perp}\,(c \parr d, \{c, d\})}\,(a^\perp \otimes b^\perp, \{\, a^\perp, b^\perp\})$$

$a^\perp, b^\perp, c, d$ are formulas that respectively decompose into $a_0, b_0, {c_0}^\perp, {d_0}^\perp$.

Let us forget everything in the sequent derivation, but the labels. The derivation above becomes the following tree of labels, which is in fact a (typed) design:



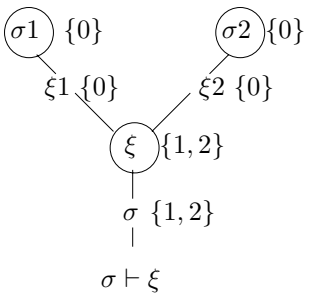
This formalism is more concise than the original sequent proof, but still carries all relevant information to retrieve its sequent calculus counter-part.

What makes this formalism possible is *focalization*. Multiplicative and additive connectives of Linear Logic (MALL) split into two families: positives $(\otimes, \oplus, 1, 0)$ and negatives $(\parr, \&, \perp, \top)$. A cluster of operations of the same polarity can be decomposed in a single step. Such a cluster can be written as a single connective, which is called a *synthetic connective*. For example the formula $(P^\perp \oplus Q^\perp) \otimes R^\perp$ has as immediate subformulas $P^\perp, Q^\perp, R^\perp$, to which we applied the connective $(- \oplus -) \otimes -$

As a consequence, in a derivation positive and negative synthetic connectives alternate at each step.

To complete the process, let us now abstract from the type annotation (the formulas), writing only the addresses. In the example above, we locate $a^\perp \otimes b^\perp$ at the address ξ; for its subformulas a and b we choose the sub-addresses $\xi 1$ and $\xi 2$. Finally we locate a_0 in $\xi 10$ and b_0 in $\xi 20$. In the same way, we locate $c \parr d$ at the address σ and so on for its subformulas. Our design becomes:

[1] In first approximation, we slightly simplify the labels.



where we have circled the address of positive formulas (we will give more detailed on the polarity –positive or negative– of the addresses in Section 2.3).

The pair (ξ, I) is called an *action*. ξ is an address (a list of natural numbers, intended as the address of the formula) and $I \in \mathcal{P}_f(\mathbb{N})$ is a finite set of natural numbers, the relative addresses of the immediate subformulas we are considering. ξ is called *focus* of the action.

† is also an action.

A *design* is given by:

a base, which is a sequent giving the conclusion of the proof (the specification of the process) and

a tree of actions with some properties that we recall in Section 2.3. A branch in the tree is called a *chronicle*. If κ_1 is before κ_2 we write $\kappa_1 < \kappa_2$.

Additives. The example we have used is simple, in that we have used a multiplicative proof, where each formula (each address) only appears once. What about the "additives"? Informally speaking, an &-rule can be seen as the super-imposition of two unary rules: $(a\&b, a)$ and $(a\&b, b)$. Given a derivation, if for any &-rule we select one of the premises, we obtain a derivation (where all &-rules are unary). This is called a slice. For example, the derivation

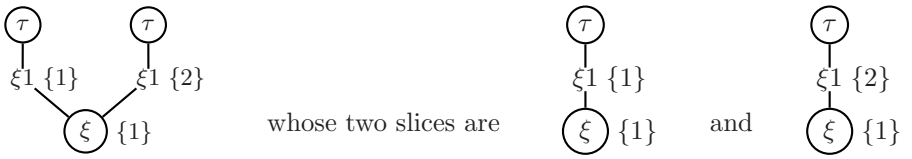
$$\dfrac{\dfrac{\dfrac{\dfrac{\cdots}{\vdash a, c}\ (a, ...) \quad \dfrac{\cdots}{\vdash b, c}\ (b, ...)}{\vdash a\&b, c}\ (a\&b, \{a\}), (a\&b, \{b\})}{\vdash (a\&b) \oplus d, c}\ ((a\&b) \oplus d, \{a\&b\})$$

can be decomposed into two slices:

$$\dfrac{\dfrac{\dfrac{\cdots}{\vdash a, c}\ (a, ...)}{\vdash a\&b, c}\ (a\&b, \{a\})}{\vdash (a\&b) \oplus d, c}\ ((a\&b) \oplus d, \{a\&b\}) \qquad \text{and} \qquad \dfrac{\dfrac{\dfrac{\cdots}{\vdash b, c}\ (b, ...)}{\vdash a\&b, c}\ (a\&b, \{b\})}{\vdash (a\&b) \oplus d, c}\ ((a\&b) \oplus d, \{a\&b\})$$

Therefore, the &-rule is a set (the super-imposition) of two actions on the *same address*.

This is the key to understand the &-rule in terms of designs. Taking again the above examples, let us locate c in the address τ, $(a\&b) \oplus d$ in the address ξ, $(a\&b)$ in $\xi 1$, a in $\xi 11$, and b in $\xi 12$. The derivation of our previous example corresponds to the following design



The actions $(\xi 1\{1\})$ and $(\xi 1\{2\})$ should be be thought of as unary &, while the usual binary rule is recovered as the set of actions on the address $\xi 1$.

2.3 Designs as Sets of Chronicles

A *design* is given by a base and a tree of actions with some properties that we are going to present. A *base* is a sequent of addresses which correspond to the "initial" sequent of the derivation, the conclusion of the proof. Focalization leads to consider only sequents of the form $\Xi \vdash \Lambda$,where Ξ has at most one element (and Λ is finite). The base (i) gives the addresses of the formulas we are going to decompose, (ii) establishes the polarity of the addresses, (iii) establishes a dependency relation between the addresses.

A sequent has a positive side (right-hand side) and a negative side (left-hand side). According to its position (r.h.s. or l.h.s.), each address in the base has a *polarity: positive or negative.* We have seen that in a synthetic connective the polarity of subformulas alternates at each layer, so if ξ is positive, ξi is negative, ξij is positive. According to its length, we say that an address is even or odd. This is called the *parity* of an address. Relative to the addresses ξ given in the base, sub-addresses of ξ with the same parity as ξ have the same polarity, sub-addresses of ξ with opposite parity have opposite polarity.

Designs are described in [6] as sets of chronicles. The definition is in two steps:

1. definition of *chronicle,* that is a *formal branch* in a focalized sequent calculus derivation,

2. definition of a *coherence condition* making a set of chronicles all belong to the same proof.

Definition 1 (Chronicle). *A chronicle* $\mathfrak{c}$ *of base* $\Xi \vdash \Lambda$ *is a sequence of actions* $\langle \kappa_0, \kappa_1, ...\kappa_n \rangle$ *such that:*

Alternation. *The polarity of* κ_j *is equal to that of the base for* j *even, opposite for* j *odd.*

Daimon. *For* $j < n$, κ_j *is not a daimon.*

Positive focuses. *The focus of a positive action* κ_p *either belongs to the basis or is an address* ξi *generated by* a *previous action:* $\kappa_q = (\xi, I), i \in I$ *and* $\kappa_q < \kappa_p$.

Negative focuses. *The focus of a negative action* κ_p *either belongs to the basis or is an address* ξi *generated by* the *previous action:* $\kappa_{p-1} = (\xi, I), i \in I$

Destruction of Focuses. *Focuses are pairwise distinct.*

Definition 2 (Coherence). *The chronicles* $\mathfrak{c}, \mathfrak{c}'$ *are* coherent *when*

Comparability. *Either one extends the other, or they first differ on negative actions, i.e. if* $\mathfrak{c}_1 = \mathfrak{c}\kappa_1 * \mathfrak{e}_1, \mathfrak{c}_2 = \mathfrak{c} * k_2 * \mathfrak{e}_2$ *with* $\kappa_1 \neq \kappa_2$ *then* κ_1, κ_2 *are negative.*

Propagation. *If* $\mathfrak{c}_1, \mathfrak{c}_2$ *first differ on* κ_1, κ_2 *with distinct focuses, then all ulterior focuses are distinct.*

Definition 3 (Design). *A design* $\mathfrak{D}$ *of base* $\Xi \vdash \Lambda$ *is a set of chronicles of base* $\Xi \vdash \Lambda$ *such that:*

Arborescence. $\mathfrak{D}$ *is closed under restriction.*

Coherence. *The chronicles of* $\mathfrak{D}$ *are pairwise coherent.*

Positivity. *If* $\mathfrak{c} \in \mathfrak{D}$ *has no extension in* $\mathfrak{D}$*, then its last action is positive.*

Totality. *Then* $\mathfrak{D}$ *is non empty.*

One is also interested in the empty design on a positive base, which is called *partial* and indicated by $\vdash \Omega$.

Notice that the above definition admits the empty chronicle, which is more natural in the setting Game Semantics, even though this is not the case in [6].

Cuts and Normalization. A set of designs to be cut together is called a cut-net. A cut between two designs is a coincidence of addresses of opposite polarity in the base of the two designs (one appears on the right-hand side, one on the left-hand side of two distinct bases).

By far, in Ludics the most important case of cut-net is the *closed* case: all addresses are cut. Given a base, its opposite is the base (or family of bases) which allow us to close the net. The opposite of $\vdash \xi$ is $\xi \vdash$; the opposite of $\xi \vdash \lambda_1, \ldots, \lambda_n$ is the family $\vdash \xi, \quad \lambda_1 \vdash, \quad \lambda_n \vdash$.

Given two design $\mathfrak{D}, \mathfrak{E}$ the normal form is indicated as $[\![\mathfrak{D}, \mathfrak{E}]\!]$. This is a (possibly partial) design. Its base is given by the uncut addresses; if $\mathfrak{D}, \mathfrak{E}$ have opposite base, since all addresses are cut, $\mathfrak{D}, \mathfrak{E}$ has conclusion $\vdash$. The normalization process then builds the tree of actions (the proof) which justifies this conclusion, as the result of the interaction between the cut designs. We start we no data (the empty design): this is our initial *partial* result. If $\mathfrak{D}, \mathfrak{E}$ do "cooperate," eventually we have a rule (an action) to justify the conclusion. In this case we will obtain a design of the form $\dfrac{}{\vdash}\ \dagger$ (the †-rule is actually the only one able to justify the empty sequent). In this case, normalization is said to converge, and $\mathfrak{D}, \mathfrak{E}$ are said to be *orthogonal*. However, it could also be that the two disigns are just unable to communicate, and normalization does not deliver any result. In this case, we remain with the partial design: $\dfrac{}{\vdash}\ \Omega$.

More interesting than the normal form, is the process of calculation itself, that is the interaction between the designs. The sequence of actions produced by this interaction is called *dispute*.

A design can also be presented as the collection of its possible interactions. In the following we will first characterize the *sequences of actions that correspond*

to a dispute. We will then characterize the *set of disputes which correspond to* interactions of the same *design,* and verify that we have *all* of them. We therefore need: (i) a "coherence condition" to guarantee that a set of disputes is compatible, meaning that all the disputes are paths on the same design, and (ii) a "saturation condition" to guarantee we have all the possible paths.

3 Arenas, Players and Legal Positions

Let us revisit some basic notions of Game Semantics in order to express the setting of Ludics.

As we have seen, an action is a pair (ξ, I) where ξ is a sequence of natural numbers, called an address, and I is a finite set of natural numbers. Each action is a "move" in Games Semantics term. The associated "dependency tree" is the universal Arena $\mathcal{U}$.

Players: The universe of addresses (and therefore of actions) is split between two *players*: one owning the even-length addresses, the other owning the odd-length addresses. Since it is convenient to fix a point of view, we will call Proponent (P) the player who starts, and Opponent (O) the other. Notice that in Ludics there is a complete symmetry between the two players: they obey the same rules. Games Semantics is generally biased toward Proponent. We will come back to this in Section 5.

Arena: An *arena* is given by a set of *moves*, a *labelling function* telling which player owns each move, and an *enabling relation* establishing a dependency relation between moves.

In the setting of Ludics, the dependency is induced by the sub-address relation *and* by the base. We say that (ξ, I) justifies $(\xi i, J)$, if $i \in I$. Moreover, if the base is $\eta \vdash \xi$, one can access ξ only after having accessed η. In this sense, ξ depends upon η.

Definition 4 (Universal Arena). *The Universal Arena $\mathcal{U}$ on the base $\vdash <>$ is given by (the initial solution of)*

$$\mathcal{U} = \textstyle\sum_{J \in P}(1 + J \times \mathcal{U})$$

where $\sum$ is the parallel composition of partial orders, and $+$ is the serial composition.

The universal arena $\mathcal{U}$ can be relocated to any initial address ξ. The moves of $\xi(\mathcal{U})$ are those of $\mathcal{U}$ with the renaming $\xi(\sigma, I) = (\xi\sigma, I)$.

The Arena on the base $\vdash \xi_1, \ldots, \xi_n$ is given by $\sum_{1 \le i \le n} \{\xi_i\} \times \mathcal{U}$

The Arena on the base $\eta \vdash \xi_1, \ldots, \xi_n$ is given by $\{\eta\} \times \mathcal{U} \leftarrow \sum_{1 \le i \le n} \{\xi_i\} \times \mathcal{U}$

(We do not explain the familiar operator $\leftarrow$ further here)

We extend the universal arena with a formal action † called *daimon.* † can be played by any player. It does not justify and is not justified by any other action.

Given the arena on a certain base, we call *initial* any action whose address belongs to the base. Notice that on the base $\eta \vdash \xi_i$, actions on either η or ξ_i are

initial moves, but any action of address ξ_i depends upon η.

Definition 5 (Linear positions). *A sequence of actions s is a* linear position, *or* play *if it satisfies the following conditions:*

Alternation *Parity alternates*
Justification *Each move is either initial or is justified by an earlier move.*
Linearity *Any address appears at most once.*
Daimon *Daimon (†) can only appear as last move.*

We call *terminating* the plays whose last move is †. ϵ indicates the empty sequence.

Notation. To indicate the players, we will use the variable X, $X \in \{P, O\}$, and $\overline{X}$ for its dual (the other player).

We will also use the notion of polarity: positive and negative. A move is positive for a player if it belongs to that player, negative if it belongs to the other.

P-move ("move belonging to P") = P-positive ("move positive for P") = O-negative ("move negative for O").

Each position belongs to one of the players, according to the last move (or more precisely to who is to play). We call P-Position a position that expects an action by Opponent, typically, a position whose last move is P. An O-Position is a position where P is to play. Since Proponent is the player who starts, we have that ϵ and all even-length positions are O-position, all odd-length positions are P-positions.

A P-position is a positive position for P, and a negative position for O. We use the notation p^P, p^O, p^+, p^-.

Let as recall the key notion of view.

Definition 6 (Views). *Let q be a linear position and $X \in \{O, P\}$ a player. Its view $\ulcorner q \urcorner^X$ of q is inductively defined as follows. When there is no ambiguity on the player, we simply write $\ulcorner q \urcorner$ for $\ulcorner q \urcorner^X$. Below, positive and negative is relative to X.*

- $\ulcorner \epsilon \urcorner = \epsilon$;
- $\ulcorner s\kappa^+ \urcorner = \ulcorner s^- \urcorner \kappa^+$;
- $\ulcorner s\kappa^- \urcorner = \kappa^-$ *if κ is initial;*
- $\ulcorner s\kappa' t\kappa^- \urcorner = \ulcorner s^- \urcorner \kappa' \kappa$, *if $\kappa = (\xi i, J)$ and $\kappa' = (\xi, I)^+$.*

We denote Opponent view *by $\ulcorner q \urcorner^O$ and* Proponent view *by $\ulcorner q \urcorner^P$. Moreover, by $\ulcorner q\kappa^+ \urcorner$ we mean the view of the player for which κ is positive. If κ belongs to X, then $\ulcorner q\kappa^+ \urcorner = \ulcorner q\kappa \urcorner^X$ and $\ulcorner q\kappa^- \urcorner = \ulcorner q\kappa \urcorner^{\overline{X}}$.*

Definition 7 (Legal positions). *A linear position p is* legal *if it satisfies the following condition:*

Visibility *If* $t\kappa \sqsubseteq p$ *and* κ *is non initial, then the justifier of* κ *occurs in* $\ulcorner t\kappa^{+}\urcorner$.

According to our convention, this means that if κ *is a P-move, its justifier occurs in* $\ulcorner t\kappa\urcorner^{P}$*, and therefore in* $\ulcorner t\urcorner^{P}$*, if* κ *is an O-move, its justifier occurs in* $\ulcorner t\urcorner^{O}$.

3.1 Designs

Let us revisit the presentation of designs. We first recall normalization. The interaction among the designs of a cut-net leads us to access some of the actions of the two designs, in a sequence which in Ludics is called dispute. Normalization converges if eventually we reach a daimon (†). Daimon is in fact is a special symbol which indicates termination (one of the players gives up). Otherwise, normalization diverges, and the result is "partial". When normalization converges, $\mathfrak{D}$ is said orthogonal to $\mathfrak{E}$ ($\mathfrak{D}\bot\mathfrak{E}$).

Let $\mathfrak{D}$ be a design of base $\vdash<>$ and $\mathfrak{E}$ a counter-design of base $<>\vdash$. From now on we focus on this case to simplify presentation[2].

We define the plays according to these two designs, $\mathcal{P} = Plays(\mathfrak{D};\mathfrak{E})$, as:
$\epsilon \in \mathcal{P}$
$p \in \mathcal{P}$ is a P to play position and $\ulcorner p\urcorner^{P}\kappa \in \mathfrak{D}$ then $p\kappa \in \mathcal{P}$
$p \in \mathcal{P}$ is an O to play position and $\ulcorner p\urcorner^{O}\kappa \in \mathfrak{E}$ then $p\kappa \in \mathcal{P}$

Fact 1 $\mathcal{P}$ *is totally ordered by initial segment.*

We indicate by $[\mathfrak{D} \rightleftharpoons \mathfrak{E}]$ the (possibly infinite) sequence of actions which is the sup of $Plays(\mathfrak{D};\mathfrak{E})$. A sequence ending with a daimon is called a *dispute*. In such a case, $\mathfrak{D}$ is said to be orthogonal to $\mathfrak{E}$: $\mathfrak{D}\bot\mathfrak{E}$.

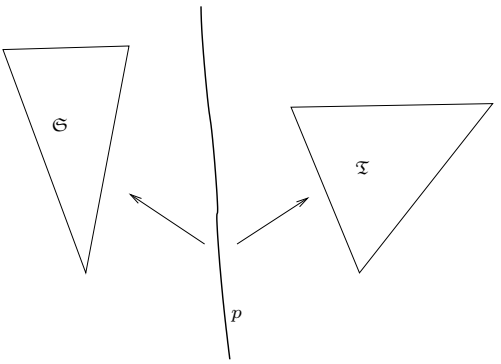
Fact 2 (Chronicles) *If* $p \in Plays(\mathfrak{D};\mathfrak{E})$*, then for any* $q \sqsubseteq r^{P} \sqsubseteq p$, $\ulcorner q\urcorner^{P}$ *is a chronicle of* $\mathfrak{D}$*. For any* $q \sqsubseteq r^{O} \sqsubseteq p$, $\ulcorner q\urcorner^{O} \in \mathfrak{E}$

Proposition 1 (Disputes as legal positions). *If* $p \in Plays(\mathfrak{D};\mathfrak{E})$ *then* p *is a legal position. Therefore in particular any dispute is a legal position on the universal arena.*

Conversely, we shall show that, given a legal position p, we can extract a design $\mathfrak{S}$ *and* a counter-design $\mathfrak{T}$ s.t. $[\mathfrak{S} \rightleftharpoons \mathfrak{T}] = p$. $\mathfrak{S}, \mathfrak{T}$ are minimal such designs.

Definition 8. *Let* p *be a finite legal position on the universal arena.*
$Des^{P}(p) = \{\ulcorner q\urcorner^{P} : q \sqsubseteq r^{P} \sqsubseteq p\}$.
$Des^{O}(p) = \{\ulcorner q\urcorner^{O} : q \sqsubseteq r^{O} \sqsubseteq p\}$.

[2] In the general case one deals with a family of designs.



Proposition 2. *Let p be a legal position on the universal arena.*

(i) $Des^P(p)$, $Des^O(p)$ *are designs on bases* $\vdash<>$, $<>\vdash$ *respectively.*
(ii) $[Des^P(p) \rightleftharpoons Des^O(p)] = p$.
(iii) If $p \in Plays(\mathfrak{D}; \mathfrak{E})$ *then* $Des^P(p) \subseteq \mathfrak{D}$ *and* $Des^O(p) \subseteq \mathfrak{E}$.

Proof. (i) Let us just check Coherence. Assume $\mathfrak{c}_1, \mathfrak{c}_2$ are incomparable, $\mathfrak{c}_1 \sqsupseteq \mathfrak{c}\kappa\kappa_1$ and $\mathfrak{c}_2 \sqsupseteq \mathfrak{c}\kappa\kappa_2$, where $\kappa_1 \neq \kappa_2$. If $\mathfrak{c}_1 \bigwedge \mathfrak{c}_2$ were not positive, then κ_1, κ_2 would be. Therefore $\mathfrak{c}\kappa\kappa_1 = \ulcorner s_1\kappa\kappa_1 \urcorner^P$, $\mathfrak{c}\kappa\kappa_2 = \ulcorner s_2\kappa\kappa_2 \urcorner^P$, and since linearity forces $s_1\kappa = s_2\kappa$, thus $\kappa_1 = \kappa_2$.

Examples. $Des^P(\epsilon) = \emptyset$, which corresponds to the partial design $\dfrac{}{\vdash<>}\ \Omega$
$Des^O(\epsilon) = \{\epsilon\}$, which corresponds to the derivation $\dfrac{}{<>\vdash}\ (<>, \emptyset)$
$Des^P(< \dagger >) = \emptyset$, which corresponds to the partial design $\dfrac{}{\vdash<>}\ \dagger$
$Des^O(< \dagger >) = \{\epsilon\}$, as before.

3.2 Designs as Set of Disputes

A design can be described by the set of its possible interaction (plays or disputes).

Given a design $\mathfrak{D}$, let us define

$$Plays(\mathfrak{D}) = \bigcup Plays(\mathfrak{D}; \mathfrak{E}), \text{ for } \mathfrak{E} \text{ design of opposite base.}$$

We have that $Plays(\mathfrak{D}) \cap Plays(\mathfrak{E}) = Plays(\mathfrak{D}; \mathfrak{E})$.

Fact 3 *If* $p \in \mathfrak{D}$ *then* $p \in Plays(\mathfrak{D})$

Fact 4 $Plays(\mathfrak{D}) = \{q \text{ legal positions, } s.t. \forall q \sqsubseteq r^+ \sqsubseteq p, \ulcorner q \urcorner \in \mathfrak{D}\}$

Proposition 3. $\mathfrak{D}$ *is recovered from* $Plays(\mathfrak{D})$ *by*

$$\mathfrak{D} = \{\ulcorner q \urcorner, q \sqsubseteq r, r \text{ is a positive position}, r \in Plays(\mathfrak{D})\}$$

Let *Disp* $\mathfrak{D}$ be the set $\{[\mathfrak{D} \rightleftharpoons \mathfrak{E}], \mathfrak{D} \perp \mathfrak{E}\}$ of terminating plays.

As any non-terminating positive play can be immediately terminated by the opponent with a daimon, any positive play belongs to *Disp* $\mathfrak{D}$. Therefore the set of disputes is enough to recover $\mathfrak{D}$.

Proposition 4. *$\mathfrak{D}$ is recovered from Disp $\mathfrak{D}$ by*

$$\mathfrak{D} = \{\ulcorner q \urcorner, q \sqsubseteq r^+ \in \mathit{Disp}\ (\mathfrak{D})\}$$

The set of possible interactions of a design can be characterized directly using the notions of Game Semantics.

4 Strategies

The universal arena gives us a game in the usual sense. There are two natural choices to give the game tree: either we consider the tree of all plays (all linear positions), or we consider only the tree of legal plays. We start by adopting this second choice, but we will come back to the first in Section 5.

There are a number of standard representations of the simplest notion of deterministic P-strategy for games. One can take any of the following.

(i) All finite plays "in accord" with the strategy.
(ii) All prefix of finite plays ending with P-moves.
(iii) All finite plays ending with P-moves.
(iv) All finite plays ending with P-moves plus all finite plays ending in O-moves to which P has no response.

These are equivalent and here is convenient to use the form (i).

Definition 9 (X-Strategy). *A P-strategy (O-strategy) S on the universal arena is a non-empty collection of plays (on that arena) which is*

1. *closed under prefix;*
2. *if $p, q \in S$ are* incomparable *then $p \bigwedge q$ is a positive position (a P-position for a P-Strategy, an O-position for an O-strategy);*
3. *if $p \in S$ is a positive position, then for all legal positions $p\kappa$, $p\kappa \in S$.*

We will call *pre-strategy* the way to present a strategy corresponding to the alternative (ii). A pre-strategy is therefore a non empty collection of plays which satisfies conditions (1) and (2) above, and such that all maximal plays are positive.

Definition 10 (Innocent Strategy). *An X-strategy S is innocent when:*
if p, q are negative positions, $\ulcorner p \urcorner^X = \ulcorner q \urcorner^X$, $p\kappa \in S$ and $q \in S$ then $q\kappa \in S$.

It is immediate by construction that

Fact 5 *If $\mathfrak{D}$ is a design of base $\vdash <>$ then $\mathit{Plays}(\mathfrak{D})$ is an innocent Player strategy (in the game given by the tree of legal plays). If $\mathfrak{E}$ is a design of base $<> \vdash$ then $\mathit{Plays}(\mathfrak{E})$ is an innocent opponent strategy.*

It is well known in Games Semantics that

(i) the collection of views of an innocent strategy generates the complete strategy, and
(ii) the collections of views of an innocent strategy S is contained in S.

Our main claim is that a design can be seen as the collection of views of an innocent strategy. From the views we can recover the strategy, from the strategy we can extract the views. Section 4.1 reviews these notions. Section 4.2 comes back to designs.

4.1 Innocent Strategies: Views and Plays

The views of an innocent strategy are enough to describe the strategy. When we do this, it is rather natural not to consider the views to which the player does not reply.

Definition 11 (Views(S)). *Let S be an X-strategy. We define*

$$Views(S) = \{\ulcorner q \urcorner^X, q \sqsubseteq p^+ \in S\}$$

We recall some properties of innocent strategies from this perspective.

Fact 6 (Closure under view) *If S is an innocent X-strategy then we have $Views(S) \subseteq S$.*

Fact 7 (Saturation) *Let T be any strategy and S an innocent strategy. If $Views(T) \subseteq S$ then $T \subseteq S$.*

Fact 8 (Determinism under view) *Let S be an innocent X-strategy. If $pab \in S, qac \in S, \ulcorner pa \urcorner = \ulcorner qa \urcorner$ then $b = c$.*

This in particular means that $Views(S)$ itself satisfies determinism (cf. (ii) in Definition 9).

Plays vs. Views. We say that a set of positions $\mathcal{V}$ is *stable under view* if $\ulcorner p \urcorner = p$ for all $p \in \mathcal{V}$.

Definition 12 (Plays($\mathcal{V}$)). *Let $\mathcal{V}$ be a pre-strategy stable under view. We define $Plays(\mathcal{V})$ as in 4:*
$Plays(\mathcal{V}) = \{q$ *legal positions, s.t.*$\forall q \sqsubseteq r^+ \sqsubseteq p, \ulcorner q \urcorner \in \mathcal{V}\}$

Proposition 5. *Let S be an innocent strategy.*
$Plays(Views(S)) = S$
$Views(S)$ *is a pre-strategy, stable under view.*

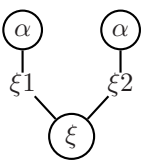
Proposition 6. *Let $\mathcal{V}$ be a pre-strategy stable under view.*
$Views(Plays(\mathcal{V})) = \mathcal{V}$
$Plays(\mathcal{V})$ is the smallest innocent strategy which contains $\mathcal{V}$.

Proof. Notice that $Plays(\mathcal{V})$ is deterministic because $\mathcal{V}$ is.
If S is an innocent strategy and $\mathcal{V} \subseteq S$ then from $Views(Plays(\mathcal{V})) = \mathcal{V}$ and Proposition 7 we deduce that $Plays(\mathcal{V}) \subseteq S$.

4.2 Designs as Innocent Strategies

Fact 9 *Let $\mathfrak{D}$ be a design. Then $\mathfrak{D}$ is a pre-strategy stable under view.*

Unfortunately, the converse is not necessarily true.
Consider for example the innocent strategy generated by the following two plays: $\{\langle \xi^+, \xi 1, \alpha\rangle, \langle \xi^+, \xi 2, \alpha\rangle\}$. To this we would associate the following tree of views:



Even though all plays are linear, we do not obtain a design, in that propagation is not satisfied (in the next section we explain better what does this mean).
A first solution is simply to translate the condition of propagation from chronicles to views. We will give a more natural solution in Section 5.

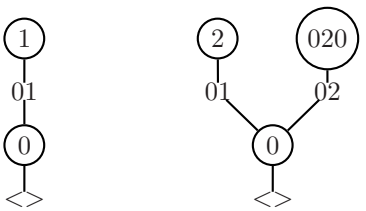
Definition 13 (Propagation). *A strategy S satisfies the propagation condition if:*
If $t\kappa, t'\kappa \in Views(S)$ and $t = \mathfrak{c} * (\xi, I) * \mathfrak{d}$, $t' = \mathfrak{c} * (\xi', I') * \mathfrak{d}'$ then $\xi = \xi'$.

Fact 10 *Let $\mathcal{V}$ be a pre-strategy which is stable under view and which also satisfies* propagation*, then $\mathcal{V}$ is a design.*

Fact 11 *(i) Let $\mathfrak{D}$ be a design. $Plays(\mathfrak{D})$ is an innocent strategy, the smallest innocent strategy which contains $\mathfrak{D}$.*
(ii) Let S be an innocent strategy which satisfies propagation*. Then $ViewS$ is a design.*
(iii) $Plays(ViewsS) = S$ and $Views(Plays\mathfrak{D}) = \mathfrak{D}$

Innocence. Notice that a strategy which is not innocent does not correspond to any construct in Ludics.
Let us consider the strategy S on $\vdash <>$ given by the closure under prefix of
$\{p_1 = \langle (<>, \{0,1,2\}), (0, I_0), (01, I_{01}), (1, J)\rangle$ and
$p_2 = \langle (<>, \{0,1,2\}), ((0, I_0), (02, I_{02}), (020, I_{020}), (01, I_{01}), (2, K))\}\rangle$

S is an O-strategy. $Des^O(p_1)$ and $Des^O(p_2)$ respectively produce the trees:



The first two chronicles cannot co-exist in the same design.

5 Linearity

As we have seen, there is only one delicate point to establish a correspondence between designs and innocent strategies, namely that it is not enough to consider linear legal positions.

The objects described by an innocent strategy are linear for all computational purpose, but we would not reach a full completeness result for **MALL**. Typically, to the example in Section 4.2, we could associate the following proof.

$$\dfrac{\dfrac{0 \vdash A}{\vdash \downarrow \top, A}}{\dfrac{\downarrow A^{\perp} \vdash \downarrow \top}{\vdash (\downarrow\uparrow A) \otimes (\downarrow\uparrow B), \downarrow \top}} \quad \dfrac{\dfrac{0 \vdash B}{\vdash \downarrow \top, B}}{\downarrow B^{\perp} \vdash \downarrow \top}$$

The formula $\downarrow \top$ appears in the context of both component of the tensor. No play satisfying visibility can detect that α (the address of the formula $\downarrow \top$) is used twice, visiting both branches of the design.

The solution we gave earlier was to ask the condition of propagation, which is a way explicitly to demand the separation of the contexts on a Tensor rule. Games suggest a better solution in the use of a more liberal notion of play, as in [1].

Let us come back to Section 4 and consider the other possible choice for the game tree: using all linear positions (not only legal ones).

Given a design $\mathfrak{D}$ let us consider

$$Plays^*(\mathfrak{D}) = \{p \text{ linear plays such that for all } q \sqsubseteq r^+ \sqsubseteq p, \ulcorner q \urcorner \in \mathfrak{D}\}$$

Fact 12 *If $\mathfrak{D}$ is a design, then $Plays^*(\mathfrak{D})$ is an innocent strategy (in the game given by the tree of all linear plays).*

Remark 1. In general, there will be $p \in Plays*(\mathfrak{D})$ in which the *opponent* does not play innocently. A position in which the *player* does not play innocently never appears in $Plays^*(\mathfrak{D})$.

Proposition 7. *If S is an innocent strategy in the tree of linear plays, then $Views(S)$ is a design.*

Extracting Strategies from a Play. We have shown that to a play p we can associate both a design and a counter-design. To be able to extract both a strategy and a counter-strategy it is essential that p is linear. For example, to the play $\langle \alpha, \alpha 0, \alpha \rangle$ we can associate a design, but not a counter-design. In other words, this play belongs to an innocent strategy, but not to an innocent counter-strategy. Notice that the issue of lifting a play to a strategy (not a counter-strategy) was addressed by Danos Herbelin and Regnier in [3].

6 Further Work

This work suggests several directions to be explored. A natural continuation is to develop a presentation of Ludics based on disputes. Moreover, since we establish a bridge between Ludics and Game Semantics, we expect to be able to transfer experiences and techniques between the two settings. The use of plays rather than views (chronicles) could allow for a finer analysis. We have seen that designs correspond to innocent strategies. It is a natural question to ask what would be the analogue of *general strategies* in Ludics. Conversely, to what would lead the notion of *location* in Games?

In this paper we only consider the first concepts in Ludics. We intend further to consider the constructions of *behaviour* and *incarnation* from the perspective of Game Semantics. It seems natural to apply the framework of Abstract Games [8]. A category of behaviours is obtained using orthogonality and double gluing. We would like to clarify the relation between that structure and the "realizability" structure on behaviours given in Ludics. Furthermore it would be interesting to investigate the extent to which behaviours regarded as abstract games can be presented as concrete games. (First steps in this direction were given in [4]).

References

[1] S. Abramsky, K. Honda, and G. McCusker. A fully abstract game semantics for general references. In *Proceedings LICS'98*. IEEE Computer Society Press, 1998. 455

[2] S. Abramsky and G. McCusker. *Computational Logic*, chapter Game semantics. Springer-Verlag, 1999. 442

[3] V. Danos, H. Herbelin, and L. Regnier. Games semantics and abstract machines. In *Proceedings LICS'96*. IEEE Computer Society Press, 1996. 456

[4] C. Faggian. *On the Dynamics of Ludics. A Study of Interaction*. PhD thesis, Université Aix-Marseille II, 2002. 456

[5] J.-Y. Girard. Geometry of interaction i: Interpretation of system f. In Z. A. Ferro R.m Bonotto C., Valentini S., editor, *Logic Colloquium 88*, pages 221–260. North Holland, 1989. 442

[6] J.-Y. Girard. Locus solum. *Mathematical Structures in Computer Science*, 2001. 442, 446, 447

[7] M. Hyland and L. Ong. On full abstraction for PCF. *Information and Computation*, 2000. 442

[8] M. Hyland and A. Schalk. Abstract Games for Linear Logic. *Electronic Notes in Theoretical Computer Science*, 29:1–24, 1999. 456

[9] H. Nickau. Hereditarily sequential functionals. In *Proceedings of the Symposium on Logical Foundations of Computer Science: Logic at St. Petersburg*, LNCS. Springer, 1994. 442