Technical Report

UCAM-CL-TR-535
ISSN 1476-2986

Number 535

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Designs, disputes and strategies

Claudia Faggian, Martin Hyland

May 2002

JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

Technical reports published by the University of Cambridge
Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/TechReports/

Series editor: Markus Kuhn

ISSN 1476-2986

Abstract

Important progresses in logic are leading to interactive and dynamical models. Geometry of Interaction and Games Semantics are two major examples. Ludics, initiated by Girard, is a further step in this direction.

The objects of Ludics which correspond to proofs are *designs*. A design can be described as the skeleton of a sequent calculus derivation, where we do not manipulate formulas, but their location (the address where the formula is stored). To study the traces of the interactions between designs as primitive leads to an alternative presentation, which is to describe a design as the set of its possible interactions, called disputes. This presentation has the advantage to make precise the correspondence between the basic notions of Ludics (designs, disputes and chronicles) and the basic notions of Games semantics (strategies, plays and views).

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 necessarely the more interesting aspect. The dynamics, the process itself of computation plays a 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 computations. This paradigm is particularly significant today, since interaction often appears to be more visible and even more important than computation.

Important progresses in logic are also leading to interactive and dynamical models. Two 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 an important development in logic, but also 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 HO Games [8]. In fact, we are going to make precise the following correspondences:

actions – moves

disputes – plays

chronicles – views

designs – innocent strategies

The crucial correspondence is

"view - chronicle - sequent calculus branch"

This correspondence is the key to move between Ludics and Games Semantics. To keep all notions concrete, one should always remember that a chronicle is a branch in a sequent calculus derivation, a design being the "skeleton" of a sequent calculus derivation. Conditions on views, as conditions on chronicles, can easily be understood as conditions on the branches of a sequent calculus derivation.

We expect to be able to transfer experiences and techniques between the two settings.

2 Ludics in a nutshell

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, all properties of proofs should be determined and tested internally to the system, where internally means interactively: proofs are tested with proofs. To do so, it is necessary to generalize the notion of proofs.

In Ludics, a proof must be thought of in the sense of "proof search" or "proof construction": we start from the conclusion, and guess a last rule, then the rule above. To stop (for example, because no rule can be applied) a new rule is introduced, the daimon:

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

When using such a rule, we assume the conclusion, without providing a justification. Such an aborted proof is still a well defined formal object The gain is that now the universe of proofs has *enough inhabitants* for any *proof* of A be tested by *proofs* of $A^{\perp}$.

2.2 Designs

We begin by giving an intuition of what is a design. This is enough to follow the rest of the paper. We recall the precise definitions (to which the proofs refer) in the appendix. Instead, we do 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 ($\mathbf{M}ALL_2$).

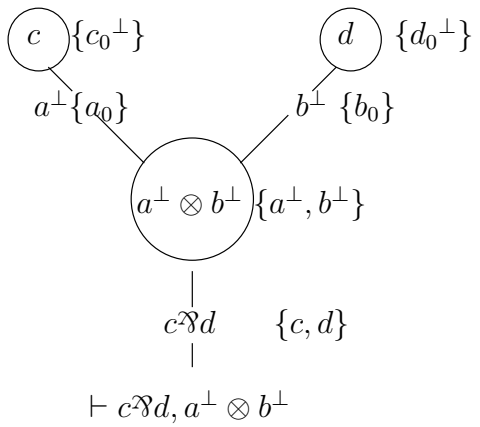
Designs capture the geometrical structure of sequent calculus derivations. The simplest way to introduce designs is to start from sequent calculus. Let us consider the following derivation, where the rules are labelled by the active formula and the subformulas which

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:



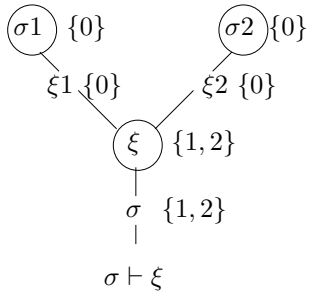
This formalism is more concise than the original sequent proof, but still carries all relevant information to retrieve its sequent calculus counter-part.

Remark 2.1 (Focalization) *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 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; this is possible when working with slices, which essentially are multiplicative derivations



Definition 2.2 (Actions) *The pair* (ξ, I) *is called an* action. *As we have seen,* ξ *is an address (a list of natural numbers, which is 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, while* I *is called ramification.*

† *is also an action, but an* improper *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 the Appendix. A branch in the tree is called a *chronicle*. If κ_1 is before κ_2 we write $\kappa_1 < \kappa_2$.

Slices and additives. The notion of slice was introduced as part of the theory of proof-nets in Linear Logic, to speak of the two components of a &-rule. Informally speaking, an &-rule is seen as the super-imposition of two unary rules: $(a\&b, a$ and $(a\&b, b)$. Given a derivation, we obtain a slice if for any &-rule we select one of the premises. The derivation

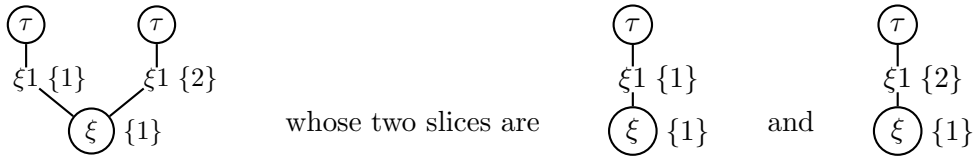
$$\dfrac{\dfrac{\dfrac{\dots}{\vdash a, c}\ a \quad \dfrac{\dots}{\vdash b, c}\ b}{\vdash a\&b, c}\ (a\&b, \{a\}), (a\&b, \{b\})}{\vdash (a\&b) \oplus d, c}\ ((a\&b) \oplus d, \{a\&b\})$$

can be decomposed as the super-imposition of two slices (in each slice the &-rule is unary)

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

In the same way, a design is a superimposition of slices.

We locate c in the address τ, $\downarrow (a\&b)$ in the address ξ, $(a\&b)$ in $\xi 0$, a in $\xi 01$, and b in $\xi 02$. The derivation of our previous example corresponds to the following design



The actions $(\xi 1\{1\})/(a\&b, a)$ and $(\xi 1\{2\})/(a\&b, b)$ should be be thought of as unary &. The usual binary rule is a set of two actions.

Notation 2.3 *In a slice, each address only appears once, therefore each action is uniquely determined by its focus. For this reason, when working with slices we often identify an action $\kappa = (\sigma, I)$ with its focus σ.*

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. The uncut addresses form a base. By far, the most important case in Ludics is the *closed* case: all addresses are cut and the base is empty

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 result of normalization is indicated as $[\![\mathfrak{D}, \mathfrak{E}]\!]$. If $\mathfrak{D}, \mathfrak{E}$ have opposite base, since all addresses are cut, there are only two possible results for normalization: either it converges ($[\![\mathfrak{D}, \mathfrak{E}]\!] = \dagger$) or it fails ($[\![\mathfrak{D}, \mathfrak{E}]\!] = \Omega$). In the first case the two designs are said *orthogonal*.

While the normal form presents no surprise, what we are interested in is the interaction itself. An easy way to present normalization of designs is by a token traveling along a cut-net (see Appendix). As the token travels on the net, it draws a path: the sequence of visited actions, which is the trace of the interaction among the designs, is called a *dispute*, and indicated as $[\mathfrak{D} \rightleftharpoons \mathfrak{E}]$. The part of the cut-net visited during the normalization is called the *pull-back* of that dispute.

In a design, the same action may appear several times, because of additive duplications. What allows us to identify a specific occurrence of an action κ is the chronicle that leads us from the base to that action. In the dispute we have enough information to retrieve the chronicle that identifies any of its actions (see Proposition 3.6).

The interaction of designs and counter-designs produces a dispute. Conversely, given a dispute, we can reconstruct the design which produced it (the pull-back).

Plan Our aim is to present a design as the collection of its possible interactions. The first step will be to characterize the *sequences of actions that correspond to a dispute.* We will then need to 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

In this section we only consider designs (more precisely cut-nets) on the empty base $<>$. The associated "dependency tree" is the universal Arena. The generalization to the base $\Xi \vdash \Lambda$ is straightforward.

Players: The universe of addresses splits into two *players*: one owning the even-length addresses (Even), the other owning the odd-length addresses (Odd). As soon as we fix a point of view, one will be called Proponent (P), the other Opponent (O). There is a complete symmetry between the two players: they are of the same nature, and therefore they obey to the same rules.

Arena: An *arena* is given by a set of *moves*, a *labelling function* from the moves to $\{P, O\}$, and an *enabling relation*. The *Universal Arena* is the forest of actions induced by the sub-address relation.

Definition 3.1 (Universal Arena) *The universal arena is given by a set of moves, a labelling function and an enabling relation, as follows:*

Moves: *The moves are all the actions* (ξ, J)*, where* ξ *is an address and* $J \in \wp_{fin}(\mathbf{N})$.

Both labelling and enabling are already coded in the action:

Labels: *The labelling is implicit in the address: all even-length addresses are attributed to one Player, all odd-length addresses are attributed to the other.*

Enabling relation:

We say that (ξ, I) *justifies* $(\xi i, J)$*, if* $i \in I$*. We call* initial move *an action which is not justified (the actions whose focus is* $<>$*).*

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

We call such a structure an atomic arena. An atomic arena is identified by an atomic base. The universal arena has base $\vdash <>$*.*

Polarity The polarity is relative to the Player: a move is positive for a player if it belongs to that player, negative if it belongs to the other. Positive means same parity ("mine"), negative means opposite parity ("yours").

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

Notation 3.2 *When we need to specify a player but do not wish to take a point of view, we will use the variables* X *where* $X \in \{P, O\}$ *and* $\overline{X}$ *for its dual.*

To make explicit if a move κ *is* P*,* O*, positive or negative we use the notation:* $\kappa^P, \kappa^O, \kappa^+, \kappa^-$*.*

Since it is convenient to fix a point of view, let us fix Proponent to be the one who starts (the player owning the initial move) and Opponent the other.

3.1 Plays

Definition 3.3 (Linear positions) *A sequence of actions* s *is a* linear position*, if it satisfies the following conditions:*

Parity *Parity alternates*

Justification *Each move is either initial or is justified by an earlier move.*

Linearity *Any address appears at most once.*

Each position belongs to one of the players, according to the last move (the following definition takes care of the case where no move has been played yet).

Definition 3.4 (X-position) *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 that expects an action by Proponent. Observe that if we choose to call Proponent the player who starts, ϵ is an O-position.*

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^-.

The key notion is that of view.

Definition 3.5 (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 $\ulcorner q\kappa^- \urcorner = \ulcorner q\kappa \urcorner^{\overline{X}}$.*

In a design, the same action may appear several times, because of the use of n-ary negative rules. What allows us to identify a specific occurrence of an action κ is the chronicle that leads us from the base to that action. The operation of view allows one to extract from the dispute the chronicle that identifies any of its actions (see [4]):

Proposition 3.6 (Chronicles extraction) *$p = [\mathfrak{D} \rightleftharpoons \mathfrak{E}]$ and $q\kappa \sqsubseteq p$. Assume κ has parity X. The chronicle that identifies κ^+ is $\ulcorner q\kappa^+ \urcorner$ (that is $\ulcorner q\kappa \urcorner^X$). The chronicle that identifies κ^- is $\ulcorner q\kappa^- \urcorner$ (that is $\ulcorner q\kappa \urcorner^{\overline{X}}$).*

The following (standard) definition of play allows us to characterize the sequence of actions that correspond to a dispute.

Definition 3.7 (Legal positions/Plays) *We say that a linear position p is* legal*, or a* play*, if it satisfies the following condition:*

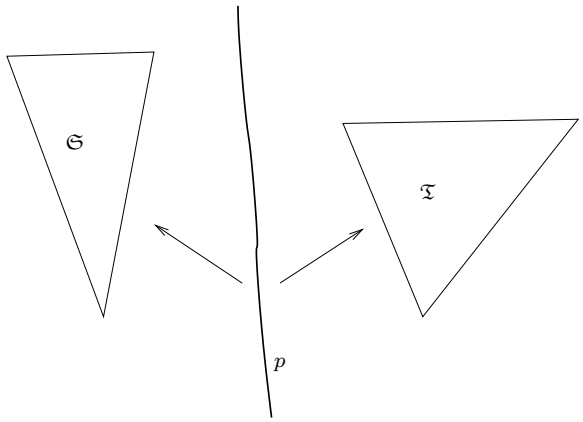
Visibility *If $t\kappa \sqsubseteq p$ where κ is non initial, then the justifier of κ occurs in $\ulcorner t\kappa^+ \urcorner$.*

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 $\ulcorner t \urcorner^O$.

Proposition 3.8 (Disputes as plays) *Any dispute p on a closed net of base $\vdash <>$ is a legal position on the universal arena.*

Proof. *Parity* and *justification* are obvious.
Visibility. Let $t\kappa \subseteq$ and κ, say, a P-move. $\ulcorner t\kappa \urcorner^P = \mathfrak{c}\kappa$ is a chronicle. Hence, by definition of chronicle, the justifier belong to $\mathfrak{c}$. □

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. $[\mathfrak{S} \rightleftharpoons \mathfrak{T}] = p$.



$\{\mathfrak{S}, \mathfrak{T}\}$ is exactly the pull-back associated to p.

To move from disputes to design we need to deal with the daimon. Our choice is to deal with it implicitly, as we shall explain, and retrieve it when we need. However, we also add † to the arena as a special move. We use this as an intermediate step which allows us more compact definitions.

Definition 3.9 (Daimon) *We extend the universal arena with a formal action* †. † *is fixed positive for any player; it does not justify and is not justified by any other action.*

Given a collection of legal positions, we define an operation of (positive) closure w.r.t. either of the player: we complete all maximal negative plays with a daimon.

Definition 3.10 (*dai*-closure) *Let S be a collection of plays on the universal arena. We define its positive closure w.r.t. the player X as follows:*
$p^{\dagger_X} = p * \dagger$ *if p is X-negative,* $p^{\dagger_X} = p$ *otherwise.*
$S^{\dagger_X} = \{p^{\dagger_X}, p \in S\}$

Definition 3.11 *Let p be a legal position.*
$Ch^P(p) = \{\ulcorner q \urcorner^P : q \sqsubseteq p, q \neq \epsilon\}$.
$Ch^O(p) = \{\ulcorner q \urcorner^O : q \sqsubseteq p, q \neq \epsilon\}$.

Proposition 3.12 (Pull-back) *Let p be a play.*

$$Ch^P(p^{\dagger_P}) \text{ is a P-slice, } Ch^O(p^{\dagger_O}) \text{ is an O-slice.}$$

Proof. Let us fix a point of view, either O or P and check that $Ch^X(p)$ is a slice.
(i) $s \in Ch(p)$ is a chronicle.
Alternation. obvious. *Daimon.* obvious. *Negative focus.* Immediate by the definition

of view on a negative action. *Positive focus.* This exactly corresponds to the visibility condition. *Destruction of foci:* imposed by linearity.

(ii) $Ch(p)$ is a design, in fact a slice.

Closure under prefix. Let $\mathfrak{c}k'k = \ulcorner qk \urcorner$, $q\kappa \sqsubseteq p$. If κ is positive: $\mathfrak{c}k' = \ulcorner q \urcorner$ and $q \subseteq p$. If κ is negative then $\mathfrak{c}\kappa'\kappa = \ulcorner s\kappa' t\kappa \urcorner = \ulcorner s \urcorner \kappa'\kappa$. We have $\mathfrak{c}\kappa' = \ulcorner s\kappa' \urcorner, s\kappa' \sqsubseteq q \sqsubseteq p$.

Coherence. $\mathfrak{c}_1, \mathfrak{c}_2 \in Ch(p)$ are coherent. If $\mathfrak{c}_1, \mathfrak{c}_2$ are incomparable, let us consider $\mathfrak{c}_1 \bigwedge \mathfrak{c}_2 = \mathfrak{c}\kappa$. If $\mathfrak{c}_1 \sqsupseteq \mathfrak{c}\kappa\kappa_1$ and $\mathfrak{c}_2 \sqsupseteq \mathfrak{c}\kappa\kappa_2$, with $\kappa_1 \neq \kappa_2$. then κ is positive. Otherwise κ_1, κ_2 would be positive. Then $\mathfrak{c}\kappa\kappa_1 = \ulcorner s_1 \kappa \kappa_1 \urcorner^P$, $\mathfrak{c}\kappa\kappa_2 = \ulcorner s_2 \kappa \kappa_2 \urcorner^P$, and since linearity forces $s_1 = s_2$, thus $\kappa_1 = \kappa_2$.

$Ch(p)$ is a Slice; propagation. Both propagation and the fact that $Ch(p)$ is a Slice are forced by linearity. If $\mathfrak{c}(\xi, I) = \ulcorner q(\xi, I) \urcorner$, $\mathfrak{c}'(\xi, I') = \ulcorner q'(\xi, I') \urcorner$, then by linearity $q(\xi, I) = q'(\xi, I')$ and $\mathfrak{c}(\xi, I) = \mathfrak{c}'(\xi, I)$.

□

Example 3.13 *On the empty base:*

$Ch^P(\epsilon^{\dagger_P}) = \{\dagger\}$, *which corresponds to the derivation* $\dfrac{}{\vdash <>}\ \dagger$.

$Ch^O(\epsilon^{\dagger_P}) = \emptyset$, *which corresponds to the derivation* $\dfrac{}{<> \vdash}\ (<>, \emptyset)$

It is immediate that

Remark 3.14 *If $q^X \sqsubseteq p$ then $Ch^X(q) \subseteq Ch^X(p)$.*

Proposition 3.15 (Plays as disputes) *To each play p on the universal arena we can associate a pair of slices $\mathfrak{S}, \mathfrak{T}$ of respective bases $\vdash <>$ $<> \vdash$ s.t. $[\mathfrak{S} \rightleftharpoons \mathfrak{T}] = p$; $\{\mathfrak{S}, \mathfrak{T}\}$ is the pull-back associated to p.*

Proof. Let $\mathfrak{S} = Ch^P(p)$ and $\mathfrak{T} = Ch^O(p)$. We need to check that $[\mathfrak{S} \rightleftharpoons \mathfrak{T}] = p$. This is immediate by the procedure of normalization on designs. If $[\mathfrak{S} \rightleftharpoons \mathfrak{T}] = t$ we show that for any prefix t_n of length n, $t_n \sqsubseteq p$.

If $p = \epsilon$ the result is immediate. Step 1. The first action in the cut-net is the same as the first move of p.

Let $t'\kappa_n = t_n = p_n$. To perform the n+1-ary step of normalization on $\mathfrak{S}, \mathfrak{T}$, we look in the slice where κ_n is negative, that is the chronicle $\mathfrak{c}\kappa_n^-$. Let $\kappa = Succ(\kappa_n^-)$; such an action exists, either proper or improper, because κ_n is negative. We know that the chronicle $\mathfrak{c}\kappa_n\kappa$ is the view of a prefix of p: $\ulcorner p'\kappa_n\kappa \urcorner^X, p'\kappa_n\kappa \sqsubseteq p$. Linearity implies that $p'\kappa_n = p_n$. Hence either $p = p_n = t_n$ and $\kappa_{n+1} = \dagger$ or $p_n\kappa_{n+1} \sqsubseteq p$ and $\kappa = \kappa_{n+1}$. This also guarantees the existence of a chronicle $\mathfrak{d}\kappa_{n+1}^-$, thus $t_n\kappa_n = p_n\kappa_n$.

□

4 Strategies

Now we want to describe designs as collections of the possible interactions, forgetting the notion of design.

Definition 4.1 (*Disp* $(\mathfrak{D})$) *Let $\mathfrak{D}$ be a design. Disp $\mathfrak{D} = \{[\mathfrak{D} \rightleftharpoons \mathfrak{E}] : \mathfrak{E} \perp \mathfrak{D}\}$.*

What we need is a "coherence" condition on disputes (characterizing disputes on the same design) and a "saturation" condition that guarantees we have all such possible interactions. To do so let us first define a coherent collection of disputes, a *strategy.*

Definition 4.2 (X-Strategy) *A P-strategy (O-strategy) S on the universal arena $\vdash <>$ is a collection of plays (on that arena) which is closed under positive prefix and such that:*

Coherence. *If $p \neq q \in S$ then $p \bigwedge q$ is a positive position (a P-position for a P-Strategy, an O-position for an O-strategy).*

Remark 4.3 *Maximal position (and only maximal position) can be negative. This means that the last action is followed by † in the design we are describing.*

Fact 4.4 *It is immediate that the above definition is equivalent to the following one, in line with the most standard Games Semantics definition:*

$S^\dagger$ is a collection of plays such that

s0. $S^\dagger$ is closed under positive prefix;

s1. $p \in S^\dagger$ then p is positive;

s2. determinism:
$sb^X, sc^X \in S^\dagger$ then $b = c$.

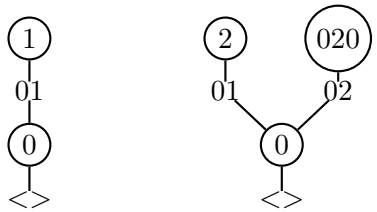
Definition 4.5 (Ch($\mathfrak{D}$)) *To an X- strategy S we associate a collection of chronicles $Ch^X(S) = \bigcup_{p \in S^{\dagger X}} Ch^X(p)$.*

Remark 4.6 *$Ch(S)$ can be seen as the super-imposition of the slices associated to the plays in S.*

It is easy to see that *Disp* $\mathfrak{D}$ is a strategy. However, a strategy does not necessarily correspond to any construct in Ludics.

Example 4.7 *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. $Ch^O(p_1)$ and $Ch^O(p_2)$ respectively produce the slices:



The two slices cannot co-exist in the same design because they differ in the way they complete the chronicle $\langle <>, 0, 01^- \rangle$. The two resulting chronicles are not coherent.

If two different disputes cover the same design, when they reach to the same negative action they continue in the same way. To arrive at the same action means they are on the same chronicle, that is the two path have the same view

The notion we need to express this condition is exactly that of innocent strategy.

Definition 4.8 (Innocent strategy) *An X-strategy S is innocent if $S^\dagger$ satisfies:*

$$sab^+ \in S^\dagger, \quad p^+ \in S^\dagger, \quad pa \text{ is a legal position}, \quad \ulcorner pa^{\neg X} = \ulcorner sa^{\neg X} \Rightarrow pab \in S^\dagger \quad (*)$$

Innocence plays two roles:

1. It assures the uniqueness of the move that follows a negative action (cf. Fact 4.15);

2. It is a condition of "saturation." It guarantees that all the possible disputes on a design are taken into account.

Remark 4.9 *As we show in Appendix C one can read "innocence" in a very concrete, procedural way, as an algorithm to calculate all possible dispute on a given design (or all designs orthogonal to a given design.)*

It is well known in Games Semantics that:

1. The collection of views of an innocent strategy generates, by innocence, the complete strategy.

2. The collections of views of an innocent strategy S is contained in S.

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 this ideas. Section 4.2 comes back to designs.

4.1 Innocent strategies: Views and Plays

The fact that an innocent strategy can be presented either as a set of plays or as a set of views is well known. The constructions $Views(-)$ and $Plays(-)$ correspond to similar operations $Fun(-)$ and $Traces(-)$ defined in [9] and [7].

The facts in this section apply to any innocent strategy. All along this section we consider w.l.o.g. strategies whose plays are all *positive*. W.r.t our previous definitions, this means that we always work with the closure of our strategies

$$S = S^\dagger.$$

Definition 4.10 (Views(S)) *Let S be an X-strategy. We define*

$$Views(S) = \{\ulcorner p^{\neg X}, p \in S\}$$

We say that a set of position $\mathcal{V}$ is stable under view *if $\ulcorner p^\neg = p$ for all $p \in \mathcal{V}$.*

Definition 4.11 (Plays($\mathcal{V}$)) *Let $\mathcal{V}$ be an X-strategy such that $Views(\mathcal{V}) = \mathcal{V}$. We define:*

$$P_0(\mathcal{V}) = \{p \in \mathcal{V} : p \text{ is minimal for } \sqsubseteq\}$$

$$P_{n+1}(\mathcal{V}) = \{pab \text{ s.t. } p \in P_n(\mathcal{V}), \exists\, \mathfrak{c}ab \in \mathcal{V}, \ulcorner pa^\neg = \mathfrak{c}a \text{ and } pa \text{ is a legal position}\}$$

$$Plays(\mathcal{V}) = \bigcup_n P_n(\mathcal{V})$$

Fact 4.12 *$Plays(\mathcal{V})$ satisfies the property:*
If $p \in Plays(\mathcal{V})$ and $\exists\ \mathfrak{c}ab \in \mathcal{V}$ s.t. $\ulcorner pa \urcorner = \mathfrak{c}a$ and pa is a legal position , then $pab \in Plays(\mathcal{V})$.

Fact 4.13 (Innocence by views) *If S is an X-strategy and $Views(S) \subseteq S$, then the property* (1) *in Definition 4.8 is equivalent to the following one:*

$$\mathfrak{c}ab^+ \in Views(S),\quad p \in S,\quad pa \text{ is a legal position },\quad \ulcorner pa \urcorner = \mathfrak{c}a \text{ then } pab \in S \quad (**)$$

Remark 4.14 *We indicate $\ulcorner p \urcorner$ as $\mathfrak{c}$ on purpose, to remember we can think of it as a chronicle.*

4.1.1 Some properties

Observe that determinism together with innocence implies in particular that

Fact 4.15 (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 solves the problem with example 4.7. It also means that $Views(S)$ satisfies itself determinism (cf. Definition 4.2) and thus it is a strategy.

Proposition 4.16 (Closure under view) *If S is an innocent X-strategy then $Views(S) \subseteq S$.*

Proposition 4.17 (Saturation) *Let T be any strategy and S an innocent strategy. If $Views(T) \subseteq S$ then $T \subseteq S$.*

4.1.2 Plays vs. Views

Proposition 4.18 *Let S be an innocent X-strategy.*

$$Views(S) \text{ is an X-strategy, stable under view.}$$

Proposition 4.19 *Let $\mathcal{V}$ be an X-strategy stable under view.*

$$Views(Plays(\mathcal{V})) = \mathcal{V}$$

Proposition 4.20 *Let $\mathcal{V}$ be an X-strategy stable under view.*

$$Plays(\mathcal{V}) \text{ is the smallest innocent strategy which contains } \mathcal{V}.$$

Proof. $Plays(\mathcal{V})$ is a strategy. It is deterministic because $\mathcal{V}$ is. Indeed, if $sa, sb \in Plays(\mathcal{V})$ then $\ulcorner sa \urcorner = \ulcorner s \urcorner a$ and $\ulcorner sb \urcorner = \ulcorner s \urcorner b$ are in $\mathcal{V}$, hence $a = b$. Moreover $Plays(\mathcal{V})$ is innocent because by construction it satisfies the condition $(**)$ of 4.13.

If S is an innocent strategy and $\mathcal{V} \subseteq S$ then $Plays(\mathcal{V}) \subseteq S$, because $Views(Plays(\mathcal{V})) = \mathcal{V}$ and Proposition 4.17. □

Proposition 4.21 *Let S be an innocent X-strategy.*

$$Plays(Views(S)) = S$$

Proof. Let $Views(S) = \mathcal{V}$. $Views(S) \subseteq S$ implies $Plays(\mathcal{V}) \subseteq S$.

We show $S \subseteq Plays(Views(S))$ by induction on the length of $p \in S$. Assume $tab \in S$. Observe that $\ulcorner tab \urcorner = \ulcorner ta \urcorner b = \mathfrak{c}ab \in \mathcal{V}$. By induction, $t \in Plays(\mathcal{V})$, $\ulcorner ta \urcorner = \mathfrak{c}a$, hence $tab \in Plays(\mathcal{V})$.

□

4.2 Designs and innocent strategies

Definition 4.22 *Let us indicate by $Views^*(S)$ the set $Views(S) \setminus \{\epsilon\}$. Let us indicate by $\overline{S}$ the closure under non empty negative prefix of S.*

Fact 4.23 $Ch(S)^+ = Views^*(S^\dagger)$.
$\mathfrak{D} = \overline{\mathfrak{D}^+}$ *and* $Ch(S) = \overline{Views^*(S^\dagger)}$

Proposition 4.24 *Let $\mathfrak{D}$ be a design of base X. Then*

$$\mathfrak{D}^+ \text{ is an X-strategy stable under view.}$$

Unfortunately, a strategy stable under view is not a design, in that it does not satisfies the condition of linearity ("propagation"). To guarantee that all slices in a design are linear, it is not enough that each single play is linear. This phenomenon is of the same nature as similar phenomenons one observes when studying interactive observability (see [4]). Since this deserves a separate analysis, it will be discussed in Section 5.

Here we simply translate the condition of propagation from chronicles to views.

Definition 4.25 (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 $\xi = \xi'$.

Proposition 4.26 *Let $\mathcal{V}$ be an X-strategy which is stable under view and which also satisfies* propagation*, then*

$$\mathcal{V} \text{ is the positive part of a design } \mathfrak{D}\text{: } \mathcal{V} = \mathfrak{D}^+$$

Using the previous proposition and the fact (cf. C) that $Disp\ (\mathfrak{D}) = Plays(\mathfrak{D}) = Plays(\mathfrak{D}^+)$ we have that:

Proposition 4.27 *(i) Let $\mathfrak{D}$ be a design. Then:*

$$Disp\ (\mathfrak{D}) \text{ is an innocent strategy,}$$

the smallest innocent strategy which contains $\mathfrak{D}^+$.

(ii) Let S be an innocent strategy which satisfies propagation*. Then:*

$$Ch(S) \text{ is a design.}$$

As for $Views$ and $Plays$,

Proposition 4.28

$$Disp\ (Ch(S)) = S \qquad and \qquad Ch(Disp\ \mathfrak{D}) = \mathfrak{D}$$

Proof. $Disp\ (Ch(S)) = Plays(Ch(S)^+) = Plays(Views(S)) = S.$

$Ch(Disp\ \mathfrak{D}) = \overline{Views(Plays(\mathfrak{D}^+))} = \overline{\mathfrak{D}^+} = \mathfrak{D}$ □

5 Some issues on Linearity

Extracting strategies from a play. We have shown that to a play p we can associate both a design and a counter-design. Notice that the issue of lifting a play to a strategy (not a counter-strategy) is addressed by Danos Herbelin and Regnier in [3]. 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.

Propagation and Linear Plays. 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 plays. We need to explicitly ask the condition of propagation.

If we consider linear plays, without imposing propagation on strategies, what do we have in ludics? This is not the same as abolishing the condition of propagation in ludics. It rather corresponds to making linearity local. *W.r.t. computation,* a design that satisfies linearity locally will behave as a design that satisfies linearity globally.

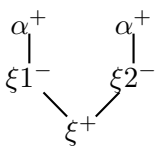
We may compare the situation to that in lambda calculus, where non-linearity of a term may be serious or not (think for example of the expression "if then else").

Propagation. The condition of propagation is a way to explicitly demand the separation of the contexts on a Tensor rule. It is immediate that we can reformulate propagation as:

"In each slice, any address only appears once".

If we abolish this condition, we need to radically change the theory. In particular, we loose linearity of the chronicles, and we do not know how to deal with separation (the analogous of Böhm theorem in Ludics).

Linear plays. Let us consider an innocent strategy of linear plays. We can associate to it a collection of chronicles. For example, to the innocent strategy $\{\langle \xi^+, \xi 1, \alpha \rangle, \langle \xi^+, \xi 2, \alpha \rangle\}$ we would associate



The objects described by innocent linear strategies are linear for all computational purpose.

However in this way we do not reach a full completeness result for **MALL**. Typically, we would find a proof such as the following one. This is exactly the proof associated to the design above, inside the behaviour which interprets the conclusion of the derivation.

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

No play satisfying justification can detect that α ($\downarrow \top$) is used twice, visiting both branches of the design. This is a typical phenomenon with designs, which also makes impossible to detect interactively the use of weakening. The plays approach suggests a possible solution in the use of a more liberal notion of play, as in [1].

If we do not have constants, and therefore all winning designs terminate by a fax, the notion of linear play entails propagation, because we can always perform some η-expansions to have enough space to allow all players to reach all addresses.

6 Further work

This work opens the way to several directions to be explored. A natural continuation is to develop a presentation of Ludics based on disputes.

Moreover, since this work establishes a bridge between Ludics and Games 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 deseigns correspond to innocent strategies. It is a natural question to ask what would be the analogous of general strategies in Ludics. Conversely, to what would lead the notion of location in Games?

Several developments arising in different contexts appear to study structures and constructions which are closely related: designs and a number of variations on the theme of innocent strategies, behaviours, orthogonality and double gluing. Our work establishes the bases for a deeper investigation in this direction.

Behaviours and Games Let us sketch the way one would follow to develop a presentation of Ludics based on disputes.

It is immediate that two strategies belonging to opposite players are orthogonal if they intersect in a play.

Definition 6.1 (Orthogonality) $\mathfrak{S} \perp \mathfrak{T}$ *if* $\mathfrak{S} \cap \mathfrak{T} = p$.

One can then proceed as in [6]. In particular, one can define a type (a game, a behaviour) in an internal way, that is without setting special rules for each type:

Definition 6.2 (Behaviours / Games) *A game* $\mathbf{G}$ *on the arena* $\vdash <>$ *is a set of innocent strategies on the same arena equal to its biorthogonal.*

One can retrieve a more standard definition of game when looking at the incarnation.

Definition 6.3 (Incarnation) *The incarnation $|S|$ of S is the set of disputes which occur both in S and a strategy of $\mathbf{G}^{\perp}$.*

A strategy is incarnated *or* material *when $S = |S|$. We define the incarnation $|\mathbf{G}|$ of $\mathbf{G}$ as the set of its material designs*

It is immediate that we have again pleasant phenomenons such as that $\mathbf{A}\&\mathbf{B} = \mathbf{A} \cap \mathbf{B}$ and $|\mathbf{A}\&\mathbf{B}| = |\mathbf{A}| \times |\mathbf{B}|$.

The "standard" definition of games would correspond to a direct definition of the incarnation.

A Designs as set of chronicles

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

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

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

Definition A.1 (Chronicle) *A chronicle $\mathfrak{c}$ of base $\Xi \vdash \Lambda$ is a non empty 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 proper.*

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 A.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 $\mathfrak{c} = \mathfrak{c} \wedge \mathfrak{c}' * k * \mathfrak{e}$,*
$c = \mathfrak{c} \wedge \mathfrak{c}' * k' * \mathfrak{e}'$ then κ, κ' are negative.

Propagation. *If $\mathfrak{c}, \mathfrak{c}'$ first differ on κ, κ' with distinct focuses, then all ulterior focuses are distinct.*

Definition A.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.*

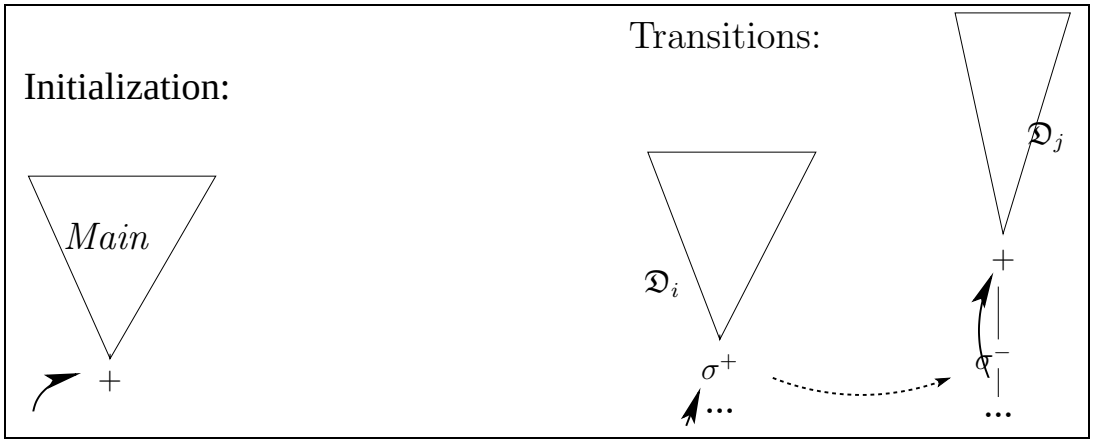
Positivity. *If $\mathfrak{c} \in \mathfrak{D}$ has no extension in $\mathfrak{D}$, then its last action is positive.*

Totality. *If the base is positive, then $\mathfrak{D}$ is non empty.*

B Normalization

Normalization of a closed net, that is of a cut-net where all addresses are cut, is especially simple. We start on the first action of the main design (the only design starting with a positive action).

Transitions: When entering a *positive action*, we exit at the corresponding negative action (then *changing of design*), and move to the unique action that follows.



Observe that if we do not work with slices, in a design the same action may appear several times, because of additive duplications. However, the sequence of visited actions carries all information needed to retrieve the position of any its action (Proposition 3.6). In particular, when we enter a positive action κ^+ we are able to retrieve the chronicle that identifies the negative action κ^- to which we have to move. Assume p is the sequence of actions we have visited so far, and we enter the positive action κ^+. We then move to the action κ^- identified by the chronicle $\mathfrak{d} = \ulcorner p\kappa^- \urcorner$.

C Generating *Disp* $\mathfrak{D}$

If we want to calculate all possible disputes on a given design $\mathfrak{D}$ we would not generate all possible designs of opposite base and execute the normalization... What we can do is to trace all possible paths, and verify that they correspond to a counter-slice. As we have just seen, this amounts to verify that the path is a dispute/play. This guarantees that the tree of actions of opposite polarity satisfies the the sub-address condition, which is the sense of the visibility condition for plays. Actually, calculating the paths which are disputes is also the easiest way to calculate the orthogonal of a design. Let us write a procedure to do this.

Definition C.1 *A chronicle* $\mathfrak{c}$ *is positive (negative) if its last action is positive (negative). Let us indicate by* $\mathfrak{D}^+$ *the subset of positive chronicles of* $\mathfrak{D}$*. We call it the* positive part *of* $\mathfrak{D}$*.*

Observe that:

1. From $\mathfrak{D}$'s point of view, a disputes always terminates on a positive action: either $\dagger$ is in $\mathfrak{D}$ (therefore normalization terminates on the chronicle $\ulcorner p^{\neg X}\dagger\urcorner$) or $\dagger$ is in $\mathfrak{E}$, therefore the last action of p is negative in $\mathfrak{E}$ and positive in $\mathfrak{D}$.

2. Moreover:

$$\mathfrak{D}^+ \subseteq \mathit{Disp}\ \mathfrak{D}$$

because for any $\mathfrak{c}^+ \in \mathfrak{D}$, $[\mathfrak{D} \rightleftharpoons \mathit{Opp}_{\mathfrak{c}}] = \mathfrak{c}$.

A completed dispute will always stop on a positive action (possibly †). To do an exhaustive search, we trace all possible positive paths of a given length, starting from the minimal ones.

Definition C.2 (Plays($\mathfrak{D}$)) *Let $\mathfrak{D}$ be a design of base X. We define:*

$P_0(\mathfrak{D}) = \{\mathfrak{c} \in \mathfrak{D}^+ : \mathfrak{c}$ *is minimal for* $\sqsubseteq\}$

$P_{n+1}(\mathfrak{D}) = \{pab$ *s.t.* $p \in P_n(\mathfrak{D})$ *and* $\exists\ \mathfrak{c}ab \in \mathfrak{D}^+ : pa$ *is a legal play and* $\ulcorner pa^{\neg X} = \mathfrak{c}a\}$

$Plays(\mathfrak{D}) = \bigcup_n P_n(\mathfrak{D})$, *augmented of* $\{\epsilon\}$ *if* $\mathfrak{D}$ *has negative base.*

This procedure describes all disputes on $\mathfrak{D}$. It is easy to understand the step P_{n+1} if one has in mind normalization. Let $\mathfrak{E}_p$ be a counterdesigns that with $\mathfrak{D}$ realizes the dispute p: $p = [\mathfrak{D} \rightleftharpoons \mathfrak{E}_p]$. If $\ulcorner p^{\neg X} \in \mathfrak{D}$ is not followed by † then $\ulcorner p^{\neg \overline{X}} = \mathfrak{d}^- \in \mathfrak{E}$, which continues with †.

Let us look for a design $\mathfrak{E}_{pab}$, such that $[\mathfrak{D} \rightleftharpoons \mathfrak{E}_p] = pab$. To do so, we substitute † with an action a such that that (i) the new design will converge against $\mathfrak{D}$ and (ii) $\mathfrak{d}^- a^+$ is actually chronicle (it must satisfy the sub-address condition). Therefore we need an action a s.t. (i) $\ulcorner pa^{\neg} = \mathfrak{c}a$ is in $\mathfrak{D}$ and (ii) pa is a play. Since $\ulcorner pa+^{\neg} = \mathfrak{d}a^+$, visibility implies that a^+ is justified by an action in $\mathfrak{d}$.

Normalization will proceed as with $\mathfrak{E}_p$ until the token is on the last action of d^-. From here it moves on to a, and then to a in $\ulcorner pa^{\neg X} = \mathfrak{c}a^- \in \mathfrak{D}$. There is a unique action b which completes this chronicle. Either b is †, and we are done, or we add b^+ to $\mathfrak{E}$ (we have seen several times that this is always possible) and complete the new chronicle with †. We have built the design $\mathfrak{E}_{pab}$ we wanted.

Proposition C.3 *Disp* $\mathfrak{D} = Plays(\mathfrak{D})$.

Proof. Let $pab^+ \in Disp\ \mathfrak{D}$. Being a dispute, $pa \sqsubseteq pab$ is a legal position. For the induction, assume $p \in Plays(\mathfrak{D})$. There is an i such that $p \in P_i(\mathfrak{D})$. Moreover, $\ulcorner qpab^{\neg} = \mathfrak{c}ab \in \mathfrak{D}^+$ and $\ulcorner pa^{\neg} = \mathfrak{c}a$. Hence, $pab \in P_{i+1}$.

Let $p \in Plays\mathfrak{D}$. If $p \in \mathfrak{D}^+$, then $p \in Disp\ \mathfrak{D}$. Since p is a play, we already know that $Pull(p)$ is a cut-net and that $[Pull^X(p) \rightleftharpoons Pull^{\overline{X}}(p)] = p$. We check that $Pull^X(p) \subseteq \mathfrak{D}$, by induction on the length of p. Assume $p = qab^+$ and $Pull^X(q) \subseteq \mathfrak{D}$. There is an i for which $qab \in P_i, \ulcorner qa^{\neg} = \mathfrak{c}a$, and $\mathfrak{c}ab \in \mathfrak{D}^+$. Therefore $\ulcorner qa^{\neg} = \mathfrak{c}a$ and $\ulcorner qab^{\neg} = \mathfrak{c}ab$ are both chronicles of $\mathfrak{D}$, hence $Pull^X(qab) \subseteq \mathfrak{D}$.

□

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.

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

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

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

[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.

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

[7] R. Harmer. *Games Semantics and Full Abstraction for Nondeterministic Languages*. PhD thesis, Imperial College, 1999.

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

[9] G. McCusker. *Games and Full Abstraction for a Functional Metalanguage with Recursive Types*. PhD thesis, Imperial College, University of London, 1996.