Electronic Notes in Theoretical Computer Science 122 (2005) 171–192 

[www.elsevier.com/locate/entcs](http://www.elsevier.com/locate/entcs)

Asynchronous Games 3

An Innocent Model of Linear Logic

# Paul-Andr´e Melli`es

*Equipe Preuves Programmes Syst`emes CNRS et Universit´e Denis Diderot Paris, France*

Abstract

Since its early days, deterministic sequential game semantics has been limited to linear or polarized fragments of linear logic. Every attempt to extend the semantics to full propositional linear logic has bumped against the so-called Blass problem, which indicates (misleadingly) that a category of sequential games cannot be self-dual and cartesian at the same time. We circumvent this problem by considering (1) that sequential games are inherently positional; (2) that they admit internal positions as well as external positions. We construct in this way a sequential game model of propositional linear logic, which incorporates two variants of the innocent arena game model: the well-bracketed and the non well-bracketed ones.

*Keywords:* Game semantics, linear logic, categorical models.

# Foreword

This paper does not simply introduce our innocent model of propositional linear logic. It also explains in detail the conceptual stages which brought it to existence. We hope that this presentation will satisfy a categorically-minded audience. The paper is organized in six sections. We start by recalling Andr´e Joyal’s category *Y* of Conway games and winning strategies (Section [1](#_bookmark0)). We prove that the category *Y* does not have binary products (Section [2](#_bookmark1)). This fact is well-known, but the proof does not appear anywhere in full details. We then reduce the *Blass problem* to the fact that the linear continuation monad *A* '→((*A* −• ⊥) −• ⊥) is strong but not commutative on Conway games. Finally, after a crash course on asynchronous games (Section [4](#_bookmark10)), we construct a linear continuation monad equivalent to the identity functor, by

1571-0661 © 2005 Elsevier B.V. Open access under [CC BY-NC-ND license.](http://creativecommons.org/licenses/by-nc-nd/3.0/)

doi:10.1016/j.entcs.2004.06.057

allowing internal positions in our games. This circumvents the Blass problem, and defines a model of linear logic (Section [5](#_bookmark13)). We conclude (Section [6](#_bookmark15)).

# Introduction: Conway games

Twenty-five years ago, Andr´e Joyal realized after a lecture by John H. Con- way on surreal numbers, that he could construct a category *Y* with Conway games as objects, and winning strategies as morphisms, composed by sequen- tial interaction. The construction appears in an article of 7 pages, written in French, and published in 1977 in the *Gazette des Sciences Math´ematiques du Qu´ebec* [[7](#_bookmark19)]. Since it is extremely difficult to get a copy of the *Gazette* today, we find useful to recall below Andr´e Joyal’s construction of the category *Y* of *Conway games*.

Before explaining the category, it may be worth discussing briefly what makes the category *Y* so interesting today. Two reasons at least. Historically, it is a precursor of game semantics for proof-theory and programming lan- guages. Conceptually, it is a self-dual category of sequential games. We are particularly interested in this last point here. The categories of games con- sidered today are generally symmetric monoidal closed, with a tensor product (noted ⊗) and a monoidal closure (noted ). Except for a few exceptions, they are not self-dual. In contrast, the category *Y* is ∗-autonomous, that is, symmetric monoidal closed, with a *dualizing* object ⊥ making the canonical morphism:

*A* −→ ((*A* ⊥) ⊥)

an isomorphism in the category *Y* , for every Conway game *A*. Since we are looking for game models of full propositional linear logic, and since linear logic is based on a duality between proofs and counter-proofs, we find extremely instructive to study the category *Y* more closely. For the reader’s comfort, we will recast the original set-theoretic formulation of Conway games [[7](#_bookmark19)] in a graph-theoretic style. This choice is also made in the recent account of (money) games by Andr´e Joyal [[8](#_bookmark20)]. This may not be the best presentation, but it clarifies the connections with our own game-theoretic model of linear logic, given in Section [5](#_bookmark13).

Conway games. A Conway game is an oriented graph (*V, E, λ*) consisting of a set *V* of vertices, a set *E* ⊂ *V* × *V* of edges, and a function *λ* : *E* −→

{−1*,* +1} associating a polarity −1 or +1 to every edge of the graph. The ver- tices are called the *positions* of the game, and the edges its *moves*. Intuitively, a move *m* ∈ *E* is played by Player when *λ*(*m*) = +1 and by Opponent when *λ*(*m*) = −1. As is usual in graph-theory, we write *x* → *y* when (*x, y*) ∈ *E*,

and call *path* any sequence of positions *s* = (*x*0*, x*1*, ..., xk*) in which *xi* → *xi*+1 for every *i* ∈ {0*, ..., k* − 1}. In that case, we write *s* : *x*0 −−→→ *xk* to indicate that *s* is a path from the position *x*0 to the position *xk*.

In order to be a Conway game, the graph (*V, E, λ*) is required to verify two additional properties:

* the graph is rooted: there exists a position ∗ called the *root* of the game, such that for every other position *x* ∈ *V* , there exists a path from the root

∗ to the position *x*:

∗→ *x*1 → *x*2 → *x*3 ··· → *xk* → *x,*

* the graph is well-founded: every sequence of positions

∗ → *x*1 → *x*2 → *x*3 → ···

starting from the root is finite.

A path *s* = (*x*0*, x*1*, ..., xk, xk*+1) is called *alternating* when:

∀*i* ∈ {1*, ..., k* − 1}*, λ*(*xi* → *xi*+1)= −*λ*(*xi*−1 → *xi*)*.*

A *play* is defined as a path *s* : ∗ −−→→ *x* starting from the root. The set of plays of a Conway game *A* is denoted *PA*.

Winning strategies. A *strategy σ* of the Conway game (*E, V, λ*) is defined as a set of alternating plays such that, for every positions *x, y, z, z*1*, z*2:

1. the empty play (∗) is element of *σ*,
2. every play *s* ∈ *σ* starts by an Opponent move, and ends by a Player move,
3. for every play *s* : ∗ −−→→ *x*, for every Opponent move *x* → *y* and Player move *y* → *z*,

∗ −−→*s*→ *x* → *y* → *z* ∈ *σ* ⇒ ∗ −−→*s*→ *x* ∈ *σ,*

1. for every play *s* : ∗ −−→→ *x*, for every Opponent move *x* → *y* and Player moves *y* → *z*1 and *y* → *z*2,

∗ −−→*s*→ *x* → *y* → *z* ∈ *σ* and ∗ −−→*s*→ *x* → *y* → *z* ∈ *σ* ⇒ *z* = *z .*

1 2 1 2

Thus, a strategy is a set of plays closed under even-length prefix (Clause 3) and deterministic (Clause 4). A strategy *σ* is called *winning* when for every play *s* : ∗ −−→→ *x* element of *σ* and every Opponent move *x* → *y*, there exists

a position *z* and a Player move *y* → *z* such that the play

∗ −−→*s*→ *x* → *y* → *z*

is element of the strategy *σ*. Note that the position *z* is unique in that case, by determinism. We write *σ* : *A* to mean that *σ* is a winning strategy of *A*.

Duality and tensor product. The dual *A*⊥ of a Conway game *A* = (*V, E, λ*) is the Conway game *A*⊥ = (*V, E,* −*λ*) obtained by reversing the polarities of moves. The tensor product *A* ⊗ *B* of two Conway games *A* and *B* is the Conway game defined below:

* its positions are the pairs (*x, y*) noted *x* ⊗ *y* of a position *x* of the game *A*

and a position *y* of the game *B*,

* its moves from a position *x* ⊗ *y* are of two kinds:

*x* ⊗ *y* → ⎧⎨ *u* ⊗ *y* if *x* → *u*,

⎩ *x* ⊗ *v* if *y* → *v*,

* the move *x* ⊗ *y* → *u* ⊗ *y* is noted (*x* → *u*) ⊗ *y* and has the polarity of the move *x* → *u* in the game *A*; the move *x* ⊗ *y* → *x* ⊗ *v* is noted *x* ⊗ (*y* → *v*) and has the polarity of the move *y* → *v* in the game *B*.

Every play *s* of the tensor product *A* ⊗ *B* of two Conway games *A* and *B* may be projected to a play *s*|*A* ∈ *PA* and to a play *s*|*B* ∈ *PB*. The Conway game 1= (∅*,* ∅*, λ*) has an empty set of positions and moves.

The category *Y* of Conway games. The category *Y* has Conway games as objects, and winning strategies of *A*⊥ ⊗ *B* as morphisms *A* −→ *B*. The identity strategy id*A* : *A*⊥ ⊗*A* copycats every move received in one component *A* to the other component. The composite of two strategies *σ* : *A*⊥ ⊗ *B* and *τ* : *B*⊥ ⊗ *C* is the strategy *τ* ◦ *σ* : *A*⊥ ⊗ *C* obtained by letting the strategies *σ* and *τ* react to a Player move in *A* or to an Opponent move in *C*, possibly after a series of internal exchanges in *B*.

A formal definition of identities and composition is also possible, but it requires to introduce a few notations. A play is called *legal* when it is alter- nating and when it starts by an Opponent move. The set of legal plays is denoted *LA*. The set of legal plays of even-length, or equivalently ending by a Player move, is denoted *L*even. The identity of the Conway game *A* is the

*A*

strategy below:

*A*⊥⊗*A*

*A*⊥⊗*A*

id*A*

= {*s* ∈ *L*even

| ∀*t* ∈ *L*even

*,t* is prefix of *s* ⇒ *t*

|*A*⊥ = *t*|*A*}*.*

The composite of two strategies *σ* : *A*⊥ ⊗ *B* and *τ* : *B*⊥ ⊗ *C* is the strategy of

*τ* ◦ *σ* : *A*⊥ ⊗ *C* below:

*τ* ◦ *σ* = {*s* ∈ *L*even

*A*⊥⊗*C*

| ∃*t* ∈ *P*

*A*⊗*B*⊗*C*

*, t*|*A,B*

∈ *σ, t*|*B,C*

∈ *τ, t*|*A,C*

= *s*}*.*

The tensor product between Conway games gives rise to a bifunctor on the category *Y* , which makes the category *Y* ∗-autonomous, that is, symmetric monoidal closed, with a dualizing object noted ⊥. The category *Y* is more than just ∗-autonomous: it is compact closed, in the sense that there exists an isomorphism (*A* ⊗ *B*)⊥ ∼= *A*⊥ ⊗ *B*⊥ natural in *A* and *B*. As in any compact closed category, the dualizing object ⊥ is isomorphic to the identity object of the monoidal structure, in that case the Conway game 1. Thus, the monoidal closure *A*⊥ ⊗⊥ is isomorphic to *A*⊥, for every Conway game *A*.

# Key observation: the category *Y* does not have bi- nary products

The category *Y* has been rediscovered at the beginning of the 90’s in the con- text of linear logic and programming language semantics. As a ∗-autonomous category, the category *Y* defines a model of Multiplicative Linear Logic (MLL). In this model, every closed formula *F* of MLL is interpreted as a Conway game [*F* ]; and every proof *π* of the formula *F* is interpreted as a winning strategy [*π*] of the Conway game [*F* ]. This interpretation provides a precise and lively picture of proofs, understood as symbolic device interacting during cut-elimination.

Because MLL is only a small fragment of linear logic, many authors have tried to adapt the category *Y* in order to capture larger or more interesting fragments of the logic. One particularly resistant fragment is Multiplicative Additive Linear Logic (MALL) which is MLL extended with the *additive* con- nectives ⊕ and & and constants 0 and T. Every ∗-autonomous category with finite products defines a model of MALL. Alas, the category *Y* does not have binary products. To our knowledge, the proof of this well-known fact ap- pears nowhere in the litterature. We thus give it below, after introducing the subcategory *Y* − of negative Conway games.

Negative Conway games. A Conway game *A* is called *negative* when every nonempty play of *A* starts by an Opponent move. The category *Y* − is defined as the full subcategory of *Y* , whose objects are the negative Conway games. The category *Y* − is symmetric monoidal closed. The symmetric monoidal structure is inherited from the category *Y* , while the monoidal closure of *Y* − is slightly different. The category *Y* − is introduced here because it has finite

products. The terminal object of the category is the Conway game 1. The cartesian product of two negative Conway games *A* and *B* is the negative Conway game noted *A*&*B*, and defined below:

* its set of positions is the disjoint sum of the set of positions of *A* and the set of positions of *B*, in which the two roots ∗*A* and ∗*B* of *A* and *B* are identified as the root ∗*A*&*B* of *A*&*B*. This construction is similar to *lifted sum* in domain theory,
* its Opponent moves from the root position ∗*A*&*B* are of two kinds:

∗*A*&*B*

→ ⎧⎨ *x* if ∗*A* → *x* in the Conway game *A*,

⎩ *y* if ∗*B* → *y* in the Conway game *B*,

* its moves from a position *x* in the component *A* are exactly the moves from

*x* in the Conway game *A*, with the same polarity:

*x* → *y* in the game *A*&*B* ⇐⇒ *x* → *y* in the game *A*.

* its moves from a position *x* in the component *B* are exactly the moves from

*x* in the Conway game *B*, with the same polarity.

It is not difficult to see that the game *A*&*B*, equipped with the accurate projection strategies *A*&*B* −→ *A* and *A*&*B* −→ *B*, defines a cartesian product of *A* and *B* in the category *Y* −. The end of the section is devoted to the proof that:

Proposition 2.1 *The category Y does not have binary products.*

Proof. The forgetful functor *U* : *Y* − −→ *Y* has a right adjoint Neg : *Y* −→ *Y* − which associates to every Conway game *A* = (*V, E, λ*) the negative Conway game Neg(*A*)= (*V* '*, E*'*, λ*) obtained by removing every Player move starting from the root ∗:

*E*' = *E* \ {(∗*, x*) ∈ *E* | *λ*(∗ → *x*)= +1}*,*

then removing every position in *V* not accessible from the root in the graph (*V, E*'):

*V* ' = {*x* ∈ *V* | there exists a path in (*V, E*') from the root to *x*}*.*

As a right adjoint, the functor Neg preserves limits. We proceed by contradic- tion, and suppose that every pair of Conway games *A* and *B* has a cartesian product noted *A* × *B* in the category *Y* . Then, the image Neg(*A* × *B*) of

this product is isomorphic to the cartesian product Neg(*A*)&Neg(*B*) in the category *Y* −.

Now, a Conway game *A* is called *positive* when its dual *A*⊥ is negative. We claim that the cartesian product of two positive Conway games *A* and *B* in *Y* , is a positive Conway game *A* × *B*. Note that a Conway game *A* is positive iff Neg(*A*) = 1. The negative game Neg(*A* × *B*) associated to the product of two positive games *A* and *B* is equal to Neg(*A*)&Neg(*B*) = 1&1 = 1. The game *A* × *B* is thus positive, as claimed.

Let *Y* + denote the full subcategory of *Y* consisting of positive Conway games. Since *Y* + is a full subcategory of *Y* , we have just established that if *Y* has binary products, then *Y* + has binary products as well. We conclude our proof of Proposition [2.1](#_bookmark2) by showing that *Y* + does not have binary products. Consider the negative game B interpreting the booleans, with four positions

∗, *q*, true, false, an Opponent move ∗ → *q* and two Player moves *q* → true

and *q* → false. Let *X* = B⊥ denote the positive game obtained by dualizing

B. Consider two positive games *A* and *B*, and suppose that *A* × *B* is their

cartesian product in *Y* +. Let the morphism *σ*true : *X* −→ *A* in the category

*Y* + denote the smallest strategy of B ⊗ *A* containing the play:

∗B ⊗ ∗*A* → *q* ⊗ ∗*A* → true ⊗ ∗*A.*

Similarly, let *τ*bool : *X* −→ *B* denote the smallest strategy of B ⊗ *B* containing the play:

∗B ⊗ ∗*B* → *q* ⊗ ∗*B* → bool ⊗ ∗*B,*

where bool is either the position true or false. Let ⟨*σ*true*, τ*false⟩ : *X* −→ *A* × *B*

denote the unique morphism in *Y* + such that

1. *σ*true = *π*1 ◦ ⟨*σ*true*, τ*false⟩ *τ*false = *π*2 ◦ ⟨*σ*true*, τ*false⟩

for *π*1 : *A* × *B* −→ *A* and *π*2 : *A* × *B* −→ *B* the projections associated to the Conway games *A* and *B* in *Y* +. A careful inspection of ([1](#_bookmark3)) establishes that the strategy ⟨*σ*true*, τ*false⟩ contains a play of the form:

∗B ⊗ ∗*A*×*B* → *q* ⊗ ∗*A*×*B* → *q* ⊗ *x.*

for a position *x* and a Player move ∗ → *x* of the game *A* × *B*; and that the strategy *π*1 : (*A* × *B*)⊥ ⊗ *A* contains a play of the form

∗*A*×*B* ⊗ ∗*A* → *x* ⊗ ∗*A* → *y*1 ⊗ ∗*A*

for a position *y*1 and an Opponent move ∗ → *y*1 of the game *A* × *B*. Similarly, the strategy *π*2 : (*A* × *B*)⊥ ⊗ *B* contains a play of the form

∗*A*×*B* ⊗ ∗*B* → *x* ⊗ ∗*A* → *y*2 ⊗ ∗*B*

for a position *y*2 and an Opponent move ∗ → *y*2 of the game *A* × *B*. Note that the two positions *y*1 and *y*2 may be equal in *A* × *B*. Now, let *ν* : *X* −→ *A* × *B* denote the smallest strategy containing the play:

∗B ⊗ ∗*A*×*B* → *q* ⊗ ∗*A*×*B* → true ⊗ ∗*A*×*B.*

And let *ν*' : *X* −→ *A* × *B* denote the smallest strategy containing all the plays of the form:

∗B ⊗ ∗*A*×*B* → *q* ⊗ ∗*A*×*B* → *q* ⊗ *x* → *q* ⊗ *y* → true ⊗ *y*

for *y* a position such that *x* → *y* is an Opponent move of *A* × *B*. The two equalities

*π*1 ◦ *ν* = *σ*true = *π*1 ◦ *ν*' *π*2 ◦ *ν* = *τ*true = *π*2 ◦ *ν*'*.*

follow immediately from these definitions. So, there exists more than one morphism *X* −→ *A* × *B* making the cartesian diagrams commute for *σ*true : *X* −→ *A* and *τ*true : *X* −→ *B*. We conclude that the category *Y* + does not have binary products. This concludes the proof of Proposition [2.1](#_bookmark2).

*Remark:* there is another more direct way to establish that the category *Y* does not have finite products, which is to show that the category *Y* does not have a terminal object. This alternative argument is less conclusive however, since it is possible to add *formally* an initial and a terminal object to the category *Y* , without breaking self-duality.

# A categorical formulation of the Blass problem

We have just seen in section [2](#_bookmark1) that

* the category *Y* is ∗-autonomous but does not have finite products,
* its subcategory *Y* − of negative Conway games is symmetric monoidal closed and has all products.

This explains why game semantics is generally more concerned with variants of the category *Y* − than with variants of the category *Y* . Prima facie, self- duality is less important than cartesianity in order to interpret a programming language built on top of the *λ*-calculus. Besides, it is much simpler to interpret the exponential modality ! of linear logic in the category *Y* − (or a variant) than in the category *Y* . By starting from the category *Y* −, one obtains a model of Intuitionistic Linear Logic (ILL) whose categorical axiomatization ensures that

the kleisli category associated to the comonad ! is cartesian closed, and thus defines a model of the simply-typed *λ*-calculus with products, see [[14](#_bookmark28),[5](#_bookmark21),[10](#_bookmark25)] among other works. By selecting among variants of the category *Y* −, and among variants of the comonad, one generates a wide range of models of the *λ*-calculus, some of them capturing the essence of particular syntactic programming languages (cf. the full abstraction results).

The methodology is nice and fruitful. We claim however that the lack of self-duality of the category *Y* − is a serious conceptual limitation of game semantics. Our ambition here is to clarify the foundations of the subject, by reunderstanding *Y* − as part of a larger ∗*-autonomous* category *Z* with *products* and *coproducts*. In this section, we try to deduce the general shape of the category *Z* from a categorical reformulation of the so-called Blass problem. We proceed by keeping the symmetry between the category *Y* − and its opposite category *Y* + as far as possible, in order to let unexpected structures emerge from the symmetry. This prepares Section [5](#_bookmark13), in which we construct a candidate for the category *Z*, a category of asynchronous games and innocent strategies.

First adjunction between lifting functors. We start our analysis by the so-called *lifting* functor ⇓ : *Y* − −→ *Y* + which associates to every negative Conway game *A*, the positive Conway game ⇓ *A* defined below:

* a position of ⇓ *A* is a position of *A* or a new position ∗,
* the only move from ∗ is the Player move ∗ → ∗*A* to the root ∗*A* of *A*,
* the moves from a position in *A* are the same as in *A*, with same polarities. By duality, there is a lifting functor ⇑ : *Y* + −→ *Y* − defined by the equation

⇑ *A* = (⇓ (*A*⊥))⊥*.* Interestingly, the functor ⇓ is left adjoint to the functor

⇑ . What this adjunction means on Conway games is very simple. Consider a negative Conway game *A*, and a positive Conway game *B*. The elements of *Y* +(⇓ *A, B*) and of *Y* −(*A,* ⇑ *B*) are the winning strategies of ⇑ (*A*⊥) ⊗ *B* and the winning strategies of *A*⊥⊗ ⇑ *B*, respectively. Note that both *A*⊥ and *B* are positive Conway games. So, the plays starting by an Opponent move are the same in the Conway games ⇑ (*A*⊥) ⊗ *B* and *A*⊥⊗ ⇑ *B*: in each case, the dummy move followed by a play in *A*⊥ ⊗ *B*. This induces a bijection between the set of strategies *Y* +(⇓ *A, B*) and *Y* −(*A,* ⇑ *B*) which is natural in *A* and

*B*. From this follows that ⇓ is left adjoint to ⇑. This adjunction induces a monad on *Y* − and a comonad on *Y* +, obtained by lifting every game twice. Note that variants of the monad on *Y* − have been already observed, typically in the litterature on arena games.

Second adjunction between lifting functors. From now on, we focus on another adjunction ↑E ↓ which follows from the adjunction ⇓E ⇑ , and which plays a fundamental role in the formulation of games as continuation passing style models. Note that the category *Y* + has coproducts, since its opposite category *Y* − has products. From this follows that the functor ⇓ factors as:

(1)

*Y* −

−→ Σ*Y*

(2)

−→ *Y*

—

+

where Σ*Y* − is the free completion of *Y* − with respect to coproducts. This completion is also called the *family construction* in [[2](#_bookmark16)]. We recall that:

* an object of Σ*Y* − is a family {*Ai* | *i* ∈ *I*} of negative Conway games *Ai*, indexed by the elements of a set *I*,
* a morphism {*Ai* | *i* ∈ *I*} −→ {*Bj* | *j* ∈ *J* } consists of a *reindexing function*

*f* : *I* −→ *J* and of a winning strategy *σi* : *Ai* −→ *Bf*(*i*), for each index *i* ∈ *I*.

Dually, the lifting functor ⇑ factors as:

(3)

*Y* +

−→ Π*Y*

(4)

−→ *Y*

+

—

where Π*Y* + is the free completion of *Y* − with respect to products. Note that the category Π*Y* + is the opposite of the category Σ*Y* −.

By composing the resulting functors together, one obtains two new “lifting” functors ↑ and ↓ defined below:

↑ : Σ*Y* − (2)

−→

(3)

−→

*Y* +

Π*Y* +*,* ↓ : Π*Y* + (4)

(1)

−→

*Y* −

Σ*Y* −*.*

Our notation ↑ and ↓ for the lifting functors indicates already that we consider Σ*Y* − as a category of positive games, and Π*Y* + as a category of negative games. Typically, we like to think of an object of Σ*Y* −, presented as a family

−→

{*Ai* | *i* ∈ *I*} of negative games, as a positive game whose initial moves by Player are the indices *i* ∈ *I*. We come back to this point later in the Section.

Interestingly, the functor ↑ is left adjoint to the functor ↓. Indeed, consider a family *A* = {*Ai* | *i* ∈ *I*} of negative Conway games, and a family *B* =

{*Bj* | *j* ∈ *J* } of positive Conway games. The family *A* is transported (or lifted) by ↑ to the singleton family consisting of the positive Conway game

⊕*i* ⇓ *Ai*, where ⊕ denotes the coproduct in *Y* +. Dually, the family *B* is transported (or lifted) by ↓ to the singleton family consisting of the negative

Conway game &*j* ⇑ *Bj*. Now, we have a series of bijections between sets:

Σ*Y* −(*A,* ↓ *B*)

=∼ Π*i*∈*IY* −(*Ai,* &*j*∈*J* ⇑ *Bj*) by definition of Σ*Y* −,

=∼ Π(*i,j*)∈*I*×*J Y* −(*Ai,* ⇑ *Bj*) because & is product in *Y* −,

=∼ Π(*i,j*)∈*I*×*J Y* +(⇓ *Ai, Bj*) thanks to the adjunction ⇓ E ⇑ ,

=∼ Π*j*∈*JY* +(⊕*i*∈*I* ⇓ *Ai, Bj*) because ⊕ is coproduct in *Y* +,

=∼ Π*Y* +(↑ *A, B*) by definition of Σ*Y* +,

whose naturality in *A* and *B* is easily established.

Σ*Y* − as a linear continuation category. As free completion of a symmet- ric monoidal closed category with products, the category Σ*Y* − is symmetric monoidal closed. The functor:

1. (−⊗−) : Σ*Y* − × Σ*Y* − −→ Σ*Y* −

is defined on the families of positive Conway games, as follows:

1. {*Ai* | *i* ∈ *I*}⊗{*Bj* | *j* ∈ *J* } = {*Ai* ⊗ *Bj* | (*i, j*) ∈ *I* × *J* }*.*

The monoidal closure *A* −• *B* is defined as follows:

1. {*Ai* | *i* ∈ *I*} −• {*Bj* | *j* ∈ *J* } = {&*i*∈*I*(*A*⊥ ⊗ *Bf*(*i*)) | *f* ∈ *I* → *J* }

*i*

So, the initial Player moves of the Conway game *A* −• *B* (equivalently, the indices of the family *A* −• *B*) are the set-theoretic functions *f* from the set *I* of initial Player moves in *A*, to the set *J* of initial Player moves in *B*. This way of defining the initial moves of *A* −• *B* does not fit in with the general philosophy of game semantics, which is to avoid “extensional” constructions like set-theoretic function spaces. Quite fortunately, one may specialize the construction to the case where *B* = ⊥ is the singleton family with the empty Conway game 1 as unique element. This defines what one calls a linear contin- uation category, that is, a symmetric monoidal category with finite coproducts distributive over the tensor product, and an exponentiable object ⊥. Besides, the resulting endofunctor *A* '→(*A* −• ⊥) of the category Σ*Y* − coincides with the endofunctor *A* '→(*A*↓ ⊥).

Σ*Y* − and Π*Y* + as categories of central maps. We have indicated that we like to think of the category Σ*Y* − as a category of *positive* Conway games. This is justified by the existence of the functor Σ*Y* − −→ *Y* + mentioned ear- lier, which transports every family {*Ai* | *i* ∈ *I*} of negative games to the positive game with initial moves the indices *i* ∈ *I*, followed by the plays of *Ai*. The functor is faithful, and injective of objects. The category Σ*Y* − is thus

isomorphic to its image in the category *Y* +, which we note *Y* +−.

•

The category *Y* +− may defined directly as follows. The objects of *Y* +− are

• •

the Conway games in which:

* every initial move in a play is by Player,
* every second move in a play is by Opponent.

The morphisms *A* −→ *B* of *Y* +− are the winning strategies *σ* : *A*⊥ ⊗ *B* such that, for every Player move ∗*A* → *x* in *A*, there exists a Player move ∗*B* → *y* in *B*, such that the play ∗*A* ⊗ ∗*B* → *x* ⊗ ∗*B* → *x* ⊗ *y* is element of the strategy *σ*. Dually, the functor Π*Y* + −→ *Y* − defines an isomorphism of categories

•

Π*Y* + ∼= *Y* −+ where *Y* −+ is defined as the opposite category of *Y* +−. It is not

• • •

difficult to see that the resulting functors:

↑ : *Y* +− −→ *Y* −+*,* ↓ : *Y* −+ −→ *Y* +−*.*

• • • •

coincide with the lifting functors ⇑ and ⇓ restricted to the subcategories *Y* +− and *Y* −+ of *Y* + and *Y* −, respectively. This justifies our notations for ↑ and ↓ . Now, let *Y* −+ denote the *full* subcategory of *Y* − with the objects of *Y* −+, and let *Y* +− denote the *full* subcategory of *Y* + with the objects of *Y* +−. By

•

•

•

•

construction, the category *Y* +− is opposite to the category *Y* −+.

There is a crucial observation to make here: the category *Y* −+ is the co- kleisli category over the category *Y* −+, induced by the comonad ↑ ↓. It is not difficult indeed to check that the set *Y* −+(*A, B*) of morphisms between two negative Conway games *A* and *B* of *Y* −+, is equal to the set *Y* +−(↓ *A,* ↓ *B*) of morphisms in the category *Y* −+. This implies that the category *Y* −+ is the *category of continuations* associated to the category *Y* +−.

•

•

•

•

The category *Y* −+ thus defines what Peter Selinger calls a (linear) *control category* in [[15](#_bookmark30)]. The category *Y* −+ is the category of *central maps* associated to this control category *Y* −+. This is the key to understand together the family construction by Samson Abramsky and Guy McCusker in [[2](#_bookmark16)], the polarized presentation of games by Olivier Laurent in [[9](#_bookmark24)], and the representation theorem of control categories by Peter Selinger in [[15](#_bookmark30)].

•

The adjunction ↑ E ↓ simulates synchronization. After this long dis- cussion, we are ready to clarify the computational meaning of the adjunction

↑E ↓ . Suppose that *A* denotes a positive Conway game in *Y* +−, and *B* a

•

negative Conway game in *Y* −+. Every element of *Y* +−(*A,* ↓ *B*) is a strat-

• •

egy *σ* of *A*⊥⊗ ↓ *B* which waits for an Opponent move *m* : ∗*A* → *x* in *A*⊥,

plays the dummy move in ↓ *B* after receiving *m*, waits for an Opponent move *n* : ∗*B* → *y* in *B*, and carries on after receiving *n*. Symmetrically, every ele- ment of *Y* −+(↑ *A, B*) is a strategy *τ* of ↓ *A*⊥ ⊗ *B* which waits for an Opponent

•

move *n* : ∗*B* → *y* in *B*, plays the dummy move in ↓ *A*⊥ after receiving *n*, waits for an Opponent move *m* : ∗*A* → *x* in *A*⊥, and carries on after receiving *m*. In both cases, the strategy *σ* or *τ* waits for the two inputs *m* : ∗*A* → *x* and *n* : ∗*B* → *y*, then carries on. In that way, the two strategies *σ* and *τ* imple- ment the *synchronized* input of *m* in *A* and *n* in *B*: the strategy *σ* simulates synchronization of *A* and *B* by asking in *A* then in *B* (in the call-by-value order) whereas the strategy *σ* asks in *B* then in *A* (in the call-by-name order).

The Conway game *A* *B*. This discussion on synchronization has a categorical counterpart. The functor associated to the adjunction ↑E ↓ :

1. (*Y* +−)op × *Y* −+ −→ Set*.*

• •

factorizes as a functor on Conway games:

1. (− −) : (*Y* +−)op × *Y* −+ −→ *Y* −+

• • •

postcomposed to the global element functor *Y* −+ −→ Set which associates to every negative Conway game its set of winning strategies. The Conway game *A* *B* is defined just as *A*⊥ ⊗ *B* except that the initial Opponent moves are pairs (*m, n*) of a Player move in *A* and an Opponent move in *B*.

•

The Blass problem. The definition of the Conway game *A* *B* coincides with the definition given by Andrea Blass in his game-theoretic account of linear logic [[3](#_bookmark18)]. Interestingly, the synchronization of the initial moves is pre- cisely what leads (apparently) to the so-called *Blass problem*. The problem is the following one: there seems to be a natural way to build a category of negative and positive games — unfortunately, this natural construction does work, because it defines to a non-associative structure, see the comprehensive account by Samson Abramsky in [[1](#_bookmark17)].

The Blass problem may be reformulated categorically in the following way. As any profunctor, the functor ([5](#_bookmark7)) induces a category *Y*• with Conway games

of *Y* +− and *Y* −+ as objects, and:

• •

* the morphisms of *Y* +− between two positive Conway games *A* and *B*,

•

* the morphisms of *Y* −+ between two negative Conway games *A* and *B*,

•

* the strategies of *A* *B* from a positive game *A* to a negative game *B*,
* no morphism from a negative game *A* to a positive game *B*.

The composition law of the category *Y*• is deduced from the composition laws of the categories *Y* +− and *Y* −+, as well as from the functor ([5](#_bookmark7)). Associativity

• •

is ensured by the bifunctoriality of ([5](#_bookmark7)).

The Blass problem arises when one tries to replace the two categories *Y* +−

•

and *Y* −+ in the construction of *Y*•, by their kleisli categories *Y* +− and *Y* −+.

•

Suppose indeed that one tries to compose a morphism *hA* : *A*' −→ *A* in the kleisli category *Y* +−, a strategy *σ* : *A* *B*, and a morphism *σ* : *B* −→ *B*' in the co-kleisli category *Y* −+. This amounts to extending the functor ([6](#_bookmark8)) to a functor

1. (− −) : (*Y* +−)op × *Y* −+ −→ *Y* −+*.*

The Blass problem amounts to the fact that there is no such functor ([7](#_bookmark9)) but only a functor:

1. (− −) : (*Y* +−)op ⊗ *Y* −+ −→ *Y* −+*.*

where (*Y* +−)op ⊗ *Y* −+ is a variant of (*Y* +−)op × *Y* −+ without the interchange law between composition and tensor product, see [[13](#_bookmark29)] for a definition. In other words, the equality:

(id*A*' *hB*) ◦ (*hA* id*B*)= (*hA* id*B*' ) ◦ (id*A* *hB*) is not necessarily verified.

Now, observe that the functors ([6](#_bookmark8)) and ([2](#_bookmark5)) are related by the natural

isomorphism *A* *B* = (*A*⊗*B*⊥)⊥. Thus, extending the functor from the categories *Y* +− and *Y* −+ to their kleisli categories *Y* +− and *Y* −+, is just like

• •

extending the functor ⊗ from *Y* +− to its kleisli category *Y* +−. This enables to

•

apply this well-known fact of the theory of monads, see [[6](#_bookmark22),[13](#_bookmark29)], that the functor

⊗ defines a premonoidal structure on *Y* +− because the linear continuation monad ↓↑ on the category *Y* +−, is strong but not commutative.

•

Towards the category *Z*. We have just reduced Blass problem to the property that the linear continuation monad *A* '→((*A* −• ⊥) −• ⊥) is strong but not commutative. This provides us with a recipe to get a model of linear logic: find an analogue of the category *Y* +− in which the linear continuation monad *A* '→((*A* −• ⊥) −• ⊥) would be *commutative*. More than that: in order to obtain a ∗-autonomous category, we want this linear continuation monad to be equivalent (as a monad) to the identity. The category of asynchronous games introduced in Section [5](#_bookmark13) is designed precisely for that purpose.

•

# A crash course on asynchronous game semantics

In this section, we recall the definitions of asynchronous games and innocent strategy given in [[12](#_bookmark27)]. The original definition of asynchronous game is adapted in three ways. First, we consider asynchronous games with a well-founded event structure, in order to relate them to Conway games. This is only a detail of presentation, since all our definitions work perfectly well in non well- founded asynchronous games. We also add an incompatibility relation #

between the moves of the game, in order to interpret the additive connectives and constants of linear logic. Finally, we polarize every position of the game with a *payoff* in {−∞*,* −1*,* +1*,* ∞} in order to distinguish between Player positions (+1*,* +∞) and Opponent positions (−1*,* −∞) as well as between internal positions (+∞*,* −∞) and external positions (+1*,* −1).

Event structures. An *event structure* (M*,* ≤*,* #) is a partially ordered set (M*,* ≤) of *events* equipped with a binary symmetric irreflexive relation # verifying:

* the set *m* ↓ = {*n* ∈ M | *n* ≤ *m*} is *ﬁnite* for every event *m* ∈ M,
* *m*#*n* ≤ *p* implies *m*#*p* for every events *m, n, p* ∈ M.

Two events *m, n* ∈ M are called *incompatible* when *m*#*n*, and *compatible* oth- erwise. Two moves *m* and *n* are called *independent* when they are compatible, and different. We write *m* I *n* in that case.

Positions. A *position* of an event structure *A* is a *ﬁnite* downward closed sub- set of (M*A,* ≤*A*), consisting of pairwise compatible events. The set of positions of *A* is denoted D*A*.

The positional graph. Every event structure *A* induces a graph G*A* whose nodes are the positions *x, y* ∈ D*A*, whose edges *m* : *x* −→ *y* are the events verifying *y* = *x*+{*m*}, where + indicates a disjoint union, that is, *y* = *x*∪{*m*} and the move *m* is not element of *x*. An event structure is called *well-founded* when the graph G*A* is well-founded.

Asynchronous games. An *asynchronous game A* = (M*A,* ≤*A,* #*A, λA, κA*) is a well-founded event structure (M*A,* ≤*A,* #*A*) whose events are called the *moves* of the game, equipped with a *polarity* function *λA* : M*A* −→ {−1*,* +1} on moves, and a *payoff* function *κA* : D*A* −→ {−∞*,* −1*,* +1*,* +∞} on positions. A move with polarity +1 (resp. −1) is called a Player (resp. Opponent) move. A Player (resp. Opponent) position is a position with payoff in {+1*,* +∞} (resp. in {−1*,* −∞}). An external (resp. internal) position is a position with payoff in {+1*,* −1} (resp. in {+∞*,* −∞}).

The underlying Conway game. The positional graph attached to the asynchronous game *A* defines a Conway game G*A*, in which the polarity of a move *x* → *y* is given by the polarity of the underlying move *m* such that *y* = *x* + {*m*} in the asynchronous game *A*. For simplicity, we write *PA* instead of *P*G*A* for the set of plays of G*A*. There is more structure in G*A* than in a usual Conway game, since every position has a payoff, and moves may be

*permuted* in plays, as explained below. The set of external positions of G*A* is denoted D◦ .

*A*

Homotopy. Given two paths *s, s*' : *x* −−→→ *y* in G*A*, we write *s* ∼1 *s*' when the paths *s* and *s*' are of length 2, with *s* = *m* · *n* and *s*' = *n* · *m* for two

moves *m, n* ∈ M*A*. The *homotopy equivalence* ∼ between paths is defined as the least equivalence relation containing ∼1, and closed under composition. We also use the notation ∼ in our diagrams to indicate that two (necessarily

independent) moves *m* and *n* are permuted. The word *homotopy* is justi- fied mathematically by the work on *directed homotopy* by Philippe Gaucher and Eric Goubault [[4](#_bookmark23)]. Indeed, every asynchronous game defines a *directed* simplicial set, in which directed homotopy between paths coincides with our permutation equivalence ∼.

Strategy. A strategy *σ* of an asynchronous game is a strategy of the under- lying Conway game G*A*, such that, moreover, every play *s* : ∗ −−→→ *x* in the strategy *σ* has its target position *x* of positive payoff: +1 or +∞. A strategy *σ* of *A* is winning when it is winning in the underlying Conway game G*A*. We write *σ* : *A* when *σ* is a winning strategy of the asynchronous game *A*.

Innocence. We reformulate in [[12](#_bookmark27)] the usual notion of innocence found in arena games, as follows. A strategy *σ* is called *innocent*, when it is *side consistent* and *forward consistent* in the following sense.

Backward consistency. A strategy *σ* is *backward consistent* (see Figure [1](#_bookmark11)) when for every play *s*1 ∈ *PA*, for every path *s*2, for every moves *m*1*, n*1*, m*2*, n*2 ∈ M*A*, it follows from

*s*1 · *m*1 · *n*1 · *m*2 · *n*2 · *s*2 ∈ *σ* and ¬(*n*1 ▶*A m*2) and ¬(*m*1 ▶*A m*2)

that

¬(*n*1 ▶*A n*2) and ¬(*m*1 ▶*A n*2) and *s*1 · *m*2 · *n*2 · *m*1 · *n*1 · *s*2 ∈ *σ.*

Forward consistency. A strategy *σ* is *forward consistent* (see Figure [2](#_bookmark12)) when for every play *s*1 ∈ *PA* and for every moves *m*1*, n*1*, m*2*, n*2 ∈ M*A*, it follows from

*s*1 · *m*1 · *n*1 ∈ *σ* and *s*1 · *m*2 · *n*2 ∈ *σ* and *m*1 I *m*2 and *m*2 I *n*1

that

*m*1 I *n*2 and *n*1 I *n*2 and *s*1 · *m*1 · *n*1 · *m*2 · *n*2 ∈ *σ.*

,, ,,

*s*2 *s*2

*n*2 ¸, *n*2 ¸,,,,*n*¸1 ,

*m*2 ¸,,,,¸,

*m*2 ¸,,,¸∼,

,,

¸,,,*m*¸,

,

*σ* e ,¸∼,

,,

¸,,¸, ⇒

,,,

∼,

, 1

∼, , ∈ *σ*

,,, ,,,

,,,¸ ¸,,,,¸

¸,

*n*1 , ,

*n*1 ,, ,, *n*2

,,,¸∼, ¸,

*m*1 ,, *m*2 *s*1

,,

,,,¸∼, ¸,

*m*1 ,, *m*2 *s*1

,,

Fig. 1. Backward consistency

*n*2 ¸,,,,*n*¸1 ,

,,

¸,,,¸,

*m*2 ¸,,,,¸∼, ¸,,,,*m*¸,

*m*2 ,

,, , 1

,, ∼, ∼ ,

*σ* e , ¸∼,

¸,,¸,

¸∈,*σ*

*σ* e ,,,¸ ¸,,,,¸, ¸∈,*σ*

,,,, ,,,,

*n*1 ,, ,, *n*2

*n*1 , ¸∼,

,,

,,

*n*

¸, ⇒

2

,,,¸∼, ¸,

*m*1 ,, *m*2

*m*1 *m* ,,

2

,,

*s*1

*s*1

Fig. 2. Forward consistency

Positional strategy. A strategy *σ* : *A* is called *positional* when for every two plays *s*1*, s*2 : ∗*A* −−→→ *x* in the strategy *σ*, and every path *t* : *x* −−→→ *y* of G*A*, one has:

*s*1 ∼ *s*2 and *s*1 · *t* ∈ *σ* ⇒ *s*2 · *t* ∈ *σ.*

Proposition 4.1 ([[12](#_bookmark27)]) *Every innocent strategy σ is positional.*

Note that every positional strategy is characterized by the set of positions of D*A* it reaches, defined as *σ*• = {*x* ∈ D*A,* ∃*s* ∈ *σ, s* : ∗*A* −−→→ *x*}*.*

# An innocent model of full propositional linear logic

Lifting of asynchronous games. The lifting ⇓ *A* of any asynchronous game *A* is the asynchronous game defined by lifting the set of moves M*A* with

an Opponent move *m*, and giving the internal and Player payoff +∞ to the root ∗⇓*A* of the asynchronous game ⇓ *A*. The operation ⇑ *A* is defined dually.

Tensor product of asynchronous games. The tensor product *A* ⊗ *B* of two asynchronous games

*A* = (M*A,* ≤*A,* #*A, λA, κA*) and *B* = (M*B,* ≤*B,* #*B, λB, κB*)

is defined by a disjoint sum of polarized event structures

(M*A* + M*B,* ≤*A* + ≤*A,* #*A* + #*B, λA* + *λB*)*.*

The underlying Conway game of *A* ⊗ *B* is thus equal to the tensor product of the underlying Conway games of *A* and *B*. The payoff *κA*⊗*B*(*x* ⊗ *y*) of a position *x* ⊗ *y* is given by the table below, in which the payoffs *κA*(*x*) and *κB*(*y*) appear in the first row and first column.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ⊗ | −∞ | −1 | +1 | +∞ |
| −∞ | −∞ | −∞ | −∞ | −∞ |
| −1 | −∞ | −∞ | −1 | +∞ |
| +1 | −∞ | −1 | +1 | +∞ |
| +∞ | −∞ | +∞ | +∞ | +∞ |

Note that the table is symmetric in *A* and *B*, and that the tensor product of an internal position with another position is always internal.

Linear games. A linear game is defined as a pair {*π* | *Ai* | *i* ∈ *I*} consisting of a polarity *π* ∈ {+1*,* −1} and of a family (*Ai*)*i*∈*I* of asynchronous games indexed by *I*. A *position* of *A* is defined as a position of any of the asynchronous games *Ai*. The *component* of a position of *A* is the asynchronous game *Ai* in which it appears. A position of *A* is called *initial* when it is the root of its component *Ai*. Every initial position in *A* is required to have a positive payoff when *π* = +1, and a negative payoff when *π* = −1. A linear game is called *negative* when *π* = −1 and *positive* when *π* = +1.

Lifting. The lifting of a negative game *A* = {−1 | *Ai* | *i* ∈ *I*} is the positive game ↓ *A* = {+1 | &*i*∈*I* ⇓ *Ai*} consisting of a unique asynchronous game &*i*∈*I* ⇓ *Ai* with polarized event structure the disjoint sum of the polarized event structures of the ⇓ *Ai*’s, with all moves in ⇓ *Ai* and ⇓ *Aj* incompatible when *i* /= *j*. Note that the underlying Conway game of &*i*∈*I* ⇓ *Ai* is the cartesian product of the underlying Conway game of each ⇓ *Ai* in the category *Y* −.

Multiplicatives. The tensor product *A*⊗*B* of two *positive* games *A* =

{+1 | *Ai* | *i* ∈ *I*} and *B* = {+1 | *Bj* | *j* ∈ *J* } is defined by synchronizing the initial positions of *A* and *B*:

def

1. *A*⊗*B* = {+1 | *Ai* ⊗ *Bj* | (*i, j*) ∈ *I* × *J* }

The tensor product of *positive* and *negative* linear games is deduced from ([9](#_bookmark14)) as:

*A*⊗*B* = *A*⊗ ↓ *B* when *A* is positive and *B* is negative,

def

*A*⊗*B* = (↓ *A*)⊗*B* when *A* is negative and *B* is positive,

def

*A*⊗*B* = (↓ *A*)⊗(↓ *B*) when *A* and *B* are negative.

def

**...**.**.**

The **...** -product of two linear games *A* and *B* is deduced by the de Morgan

**.**

**...**.**.**

def

⊥ **...**.**.** ⊥ ⊥

equality: *A* **..**. *B* = (*A* **...** *B* ) where duality is defined as expected. Note

**. .**

**.**

**.....**

that the linear game *A*⊗*B* is always positive, and that the linear game *A* **..**. *B*

**. .**

is always negative.

Strategies. A strategy *σ* of a negative linear game {−1 | *Ai* | *i* ∈ *I*} is defined as a family { *σi* | *i* ∈ *I*} of strategies *σi* of the asynchronous game *Ai*. The strategy *σ* is innocent (resp. winning) when each strategy *σi* is innocent (resp. winning).

External equivalence. The main idea underlying our model is that two innocent strategies should be identified when they meet the same *external* positions. The set of *external* positions of the strategy *σ* = { *σi* | *i* ∈ *I*} on a

negative linear game {−1 | *Ai* | *i* ∈ *I*} is the family *σ*◦ = { *σ*• ∩ D◦

| *i* ∈ *I*}*.*

*i Ai*

Two innocent strategies *σ* and *τ* of a linear game *A* are called *externally*

*equivalent* when *σ*◦ = *τ* ◦. We write this *σ* *A τ* .

The category *Z*. The category *Z* has linear games as objects, and -

⊥ **...**.**.**

equivalence classes of winning innocent strategies of *A* **...** *B* as morphisms

**.**

from *A* to *B*.

Proposition 5.1 *The category Z is* ∗*-autonomous and has all products.*

We indicate briefly how morphisms *σ* : *A* −→ *B* behave in the category *Z*.

When *A* and *B* are positive, the morphism *σ* is a strategy (modulo ) of

*A*  **..** ↓ *B* which thus waits for an initial position (*x,* ∗) in *A*↓ *B*, then plays

⊥ **.....**

⊥ **.....**

either (*x,* ∗) → (*x, y*) where *y* is an initial position of *B*, or (*x,* ∗) → (*x*'*,* ∗) where *x*' is a position of payoff −∞ in *A*. The payoff condition on *x*' follows from the definition of a strategy *σ*, which says that, if played, the position (*x*'*,* ∗) is necessarily of Player payoff. Since the payoff of ∗ is −∞ in ↓ *B*,

the payoff of *x*' is necessarily +∞. Call a linear game *external* when all its positions are external. In that case, a morphism between two *external* positive games *A* and *B* behaves in the same way as a central map on Conway games, discussed in Section [3](#_bookmark4). That is, after receiving the initial position *x* of *A*, the strategy *σ* plays necessarily an initial position *y* in *B*. Interestingly, the monoidal closure of two such external positive games *A* and *B* in the

⊥ **...**.**..**

category *Z* is the non-external and negative game *A**B* in which Opponent

**.**

plays an initial position of *A*, then Player answers with a Player move in *B*. Thus, a strategy *A*⊗*B* −→ *C* in which *A, B, C* are external positive, waits for a pair (*x, y*) of initial positions in *A* and *B*, then plays an initial position in *C*.

⊥ **....**

Exactly the same can be said of a strategy *A* −→ *B* **...** *C*. This improves the

**. .**

definition of monoidal closure ([4](#_bookmark6)) in a very nice way, since the set-theoretic definition of ([4](#_bookmark6)) is simply replaced by altering the order in which Player and Opponent appear in *A* −• *B*.

When *A* and *B* are negative, the situation is dual to the previous one.

When *A* is positive and *B* is negative, the strategy *σ* (modulo ) waits for a pair of an initial position in *A* and an initial position in *B*, then plays a move in *A* or in *B*.

When *A* is negative and *B* is positive, the strategy *σ* (modulo ) plays an initial position in *A* or an initial position in *B*, as long as this position is internal. Indeed, the strategy *σ* is forbidden to play an external position *x* in *A* or *y* in *B* since the resulting position (*x,* ∗) or (∗*, y*) would be of payoff −∞

⊥ **...**.**..**

in the negative game ↑(*A*)↑ *B*.

**.**

This ensures that every positive game *A* is isomorphic to the negative game

↑ *A*. From this follows that *Z* has all products, since the full subcategory of negative games in *Z* is easily shown to have products, in the same way as the categories *Y* − or Π*Y* +.

A model of linear logic. The category *Z* may be equipped with a linear exponential ! constructed according to the group theoretic ideas of [[11](#_bookmark26)] in order to establish that:

Proposition 5.2 *The category Z together with the linear exponential* ! *deﬁnes a model of full propositional linear logic, in the sense of [*[*14*](#_bookmark28)*,*[*5*](#_bookmark21)*,*[*10*](#_bookmark25)*].*

Besides, the category *Z* incorporates two well-known variants of the inno- cent arena game model: the well-bracketed and the non well-bracketed ones. More precisely, there are structure preserving functors *F* (resp. *G*) from the category of arena games and well-bracketed (resp. non well-bracketed) inno- cent strategies, to the category *Z*. The two functors *F* and *G* differ mainly

in the way they translate the boolean arena (noted bool). The linear game *F* (bool) is the positive game 1 ⊕ 1 with two external positions, equivalently the negative game ↑(1⊕1) obtained by lifting 1⊕1 with an internal position ∗. The linear game *G*(bool) is the negative game obtained by lifting 1 ⊕ 1 with an external position ∗. The functor *F* is full, but not faithful, whereas the functor *G* is full and faithful.

This may be illustrated as follows. The left and right implementations of the and operator of type *X* = bool bool bool are interpreted as different strategies *σ*1 and *σ*2 in the category of well-bracketed strategies. The functor *F* transports them to strategies *F* (*σ*1) and *F* (*σ*2) identified in the linear game *F* (*X*), because they hit the same set of external positions in *F* (*X*). On the other hand, the functor *G* transports *σ*1 and *σ*2 to strategies *G*(*σ*1) and *G*(*σ*2) not identified in the linear game *G*(*X*) because all the positions in *G*(*X*) are external. Intuitively, the external positions track the “terminal states” in the game *F* (*X*), and all the “intermediate states” in the game *G*(*X*).

# Conclusion and future work

The equality *A* =↓↑ *A* is reminiscent of the Reidemeister moves in tangle di- agrams. By imposing the equality, we identify enough strategies in order to obtain a model of propositional linear logic. We conjecture that the cate- gory *Z* (or a close variant) provides a fully complete model of propositional linear logic. In a related line of research, we would like to understand the re- lationship between the category *Z* and the free bicompletion of the singleton category, with respect to limits and colimits, as characterized and popularized by Andr´e Joyal.

Acknowledgements. The main part of Section [3](#_bookmark4) was developed in collab- oration with Peter Selinger, during the winter 2000-2001. We have learned recently that a similar line of research was developed by Robin Cockett and Robert Seely, on polarized categories and modules. There remains to under- stand how their work relates to the work presented here.

# References

1. Samson Abramsky. Sequentiality vs. concurrency in games and logic. Report Research Report RR-01-08, Oxford University, Programming Research Group, April 2001.
2. Samson Abramsky and Guy McCusker. Call-by-value games. In Mogens Nielsen and Wolfgang Thomas, editors, *Proceedings of CSL’97*, volume 1414 of *Lecture Notes in Computer Science*. Springer, 1998.
3. Andrea Blass. A games semantics for linear logic. *Annals of Pure and Applied Logic*, 56:183– 220, 1992.
4. Eric Goubault. Geometry and concurrency: A user’s guide. *Mathematical Structures in Computer Science*, 10(4), August 2000.
5. Martin Hyland. *Game Semantics*. Publications of the Newton Institute. Cambridge University Press, 1997.
6. Bart Jacobs. Semantics of weakening and contraction. *Annals of Pure and Applied Logic*, 69:73–106, 1994.
7. Andr´e Joyal. Remarques sur la th´eorie des jeux a` deux personnes. *Gazette des Sciences Math´ematiques du Qu´ebec*, 1(4):46–52, 1977.
8. Andr´e Joyal. Free lattices, communication and money games. In *Logic and scientiﬁc methods. Volume one of the proceedings of the tenth international congress of logic, methodology and philosophy of science, Florence, Italy, Synth. Libr. 259*, pages 29–68. Dordrecht: Kluwer Academic Publishers, 1997.
9. Olivier Laurent. Polarized games (extended abstract). In *Proceedings of the seventeenth annual symposium on Logic In Computer Science*, pages 265–274, Copenhagen, July 2002. IEEE Computer Society Press.
10. Paul-Andr´e Melli`es. Categorical models of linear logic revisited. Pr´epublication ´electronique PPS//03/09//n◦ 22 (pp), Laboratoire Preuves, Programmes et Syst`emes, March 2003. To appear in *Theoretical Computer Science*.
11. Paul-Andr´e Melli`es. Asynchronous games 1: a group-theoretic formulation of uniformity. Pr´epublication ´electronique PPS//04/06//n◦ 31 (pp), Laboratoire Preuves, Programmes et Syst`emes, 2004.
12. Paul-Andr´e Melli`es. Asynchronous games 2: the true concurrency of innocence. In Philippa Gardner and Nobuko Yoshida, editors, *Proceedings of CONCUR ’04*, Lecture Notes in Computer Science. Springer Verlag, September 2004.
13. John Power and Edmund Robinson. Premonoidal categories and notions of computation.

*Mathematical Structures in Computer Science*, 7(5):453–468, 1997.

1. Robert Seely. Linear logic, ∗-autonomous categories and cofree coalgebras. In *Applications of categories in logic and computer science*, number 92. Contemporary Mathematics, 1989.
2. Peter Selinger. Control categories and duality: on the categorical semantics of the lambda-mu calculus. *Mathematical Structures in Computer Science*, 11(2):207–260, 2001.