# A Logic of Sequentiality*

Martin Churchill and James Laird

University of Bath, United Kingdom
{m.d.churchill,j.d.laird}@bath.ac.uk

**Abstract.** Game semantics has been used to interpret both proofs and functional programs: an important further development on the programming side has been to model higher-order programs with state by allowing strategies with "history-sensitive" behaviour. In this paper, we develop a detailed analysis of the structure of these strategies from a logical perspective by showing that they correspond to proofs in a new kind of affine logic.
We describe the semantics of our logic formally by giving a notion of categorical model and an instance based on a simple category of games. Using further categorical properties of this model, we prove a full completeness result: each total strategy is the semantics of a unique cut-free *core* proof in the system. We then use this result to derive an explicit cut-elimination procedure.

**Key words:** Game semantics, sequentiality, full completeness

## 1 Introduction

In recent years, it has proved fruitful to give semantics of proofs and programs using game models. A proposition is represented as a game, corresponding to a dialogue between a Proponent asserting the proposition and an Opponent attempting to refute it. The (winning) strategies for Proponent provide us with a syntax-independent meaning for proofs for the corresponding formulas. Interestingly, there are typically winning strategies for dialogue games which do not correspond to any proof (see e.g [5]) whereas (viewed through the Curry-Howard correspondence) they are the denotations of programs in models of higher-order programming languages with imperative features. For such languages, a wealth of full abstraction and definability results have been established [4, 1, 12, 14], with applications in verification.

Our work attempts to give a logical description of such strategies. We develop a logic where derivations correspond both to (winning) strategies on a natural, basic notion of game, *and* to finitary stateful programs. There is a good reason we achieve both in the same system: the simple game model we use contains

* Proceedings of CSL 2010, Springer Verlag, Editors A. Dawar and H. Veith. The original publication is available at:
http://www.springerlink.com/content/r62nw77228k232q2/

fully abstract models of an object-oriented language [22] and a coroutine-based language [14].

**Related work.** Early game models give a computational meaning for formulas of first-order classical logic (Lorenzen) and Linear Logic [5]. One problem here is that there are winning strategies which do not correspond to any proof — and even for formulas which are not provable, such as the MIX rule [5]. In [3] a games model of multiplicative linear logic was given where every (history-free, uniform) winning strategy is the denotation of some proof. In [17], a fully complete games model was given for Polarised Linear Logic, with a similar full completeness result with respect to a class of strategies with limited access to history (innocent strategies). In our work, we provide a logic which provides a full completeness result with respect to arbitrary history-sensitive strategies — further each strategy is the denotation of a *unique* proof of a certain kind. This is done by taking the (very natural, simple) games model itself as a starting point. Similarly motivated work includes independence-friendly logic [8], computational logic [11] and ludics [7].

It was first noted in [4] that history-sensitive game models have a notion of state "built in", and so they are well-suited to modelling imperative programs. Various extensions to this model have been proposed, including languages with expressive control operators [14] and higher-order store [1]. In [19], a fully abstract game model is given for an object-oriented language using a very simple class of game — games are just trees and strategies subtrees satisfying a determinacy condition. It is this notion of game we will consider here.

**Contribution.** We develop a core proof system where proofs are in bijective correspondence with history-sensitive strategies. This core proof system is focused — the rules that can be used to conclude a sequent (a list of formula) are determined by the outermost connective in the head of that sequent. We can extend this core proof system with rules such as cut, tensor and weakening. By giving semantics to these rules, we can show that they are admissible: we can explicitly eliminate them by calculating semantics and then computing the corresponding core proof.

As the games model in question can be used to model (finitary) imperative objects, so can this logic: cut corresponds to composition; tensor to aggregation of methods into a single object; and so on. We will explore this informally, with a particular example. More generally, we have a logic where the computational content of a proof is a *stateful* program.

The semantics of the proof system are given using categorical axioms. The requirements are based on a sequoid operator, and are a subset of those required to model a coroutine calculus in [14]. Categorical axioms for full completeness in the style of [2] are identified, and our game model satisfies these axioms.

**Acknowledgements.** The authors would like to thank Guy McCusker for useful discussion and comments on earlier drafts; and Makoto Takeyama for the insights provided by jointly formalising this work in the theorem prover Agda.

## 2 Games and Strategies

### 2.1 Games

Our notion of game is essentially that introduced by [5], and similar to that of [3, 15].

**Definition 1.** *A* game *is a tuple* $(M_A, \lambda_A, b_A, P_A)$ *where*

- $M_A$ *is a set of moves*
- $\lambda_A : M_A \to \{O, P\}$
  - *We call* $m$ *an* O-move *if* $\lambda_A(m) = O$ *and a* P-move *if* $\lambda_A(m) = P$.
- $b_A \in \{O, P\}$ *specifies a starting player*
  - *We call* $s \in M_A^*$ alternating *if* $s$ *starts with a* $b_A$*-move and alternates between* O*-moves and* P*-moves. Write* $M_A^{\circledast}$ *for the set of such sequences.*
- $P_A \subseteq M_A^{\circledast}$ *is a nonempty prefix-closed set of valid plays.*

For example, the game $\mathbf{N}$ of natural numbers is $(q \cup \mathbb{N}, \{q \mapsto O, n \mapsto P\}, O, \{\epsilon, q\} \cup \{qn : n \in \mathbb{N}\})$ (where $\epsilon$ denotes the empty sequence). We write $o$, $p$ for the "single move" games $(\{q\}, \{q \mapsto O\}, O, \{\epsilon, q\}\})$ and $(\{q\}, \{q \mapsto P\}, P, \{\epsilon, q\}\})$ respectively.

We will call a game $A$ *negative* if $b_A = O$ and *positive* if $b_A = P$. We write $A, B, C, \ldots$ for arbitrary games; $L, M, N, \ldots$ for arbitrary negative games and $P, Q, R, \ldots$ for arbitrary positive games. Define $\neg : \{O, P\} \to \{O, P\}$ by $\neg(O) = P$ and $\neg(P) = O$.

### 2.2 Connectives

If $X$ and $Y$ are sets, let $X + Y = \{\mathsf{in}_1(x) : x \in X\} \cup \{\mathsf{in}_2 : y \in Y\}$. We use standard notation $[f, g]$ for copairing. If $s \in (X+Y)^*$ then $s|_i$ is the subsequence of $s$ consisting of elements of the form $\mathsf{in}_i(z)$. If $X_1 \subseteq X^*$, $Y_1 \subseteq Y^*$ define:

- $X_1 \| Y_1 = \{s \in (X + Y)^* : s|_1 \in X_1 \wedge s|_2 \in Y_1\}$
- $X_1 \| \!| Y_1 = \{s \in X_1 \| Y_1 : \forall t \sqsubseteq s.t|_1 = \varepsilon \Longrightarrow t|_2 = \varepsilon\}$
- $X_1 +^* Y_1 = \{s \in X_1 \| Y_1 : s|_1 = \epsilon \vee s|_2 = \epsilon\}$

We describe operators on games in Table 1 with abbreviations $M_{A+B} = M_A + M_B$, $\lambda_{A+B} = [\lambda_A, \lambda_B]$ and $P_A^{\circledast} = P \cap M_A^{\circledast}$. A play in $M \otimes N$ consists of an interleaving of a play in $M$ and a play in $N$. A play in $M \oslash N$ is also such an interleaving, but with a further restriction: the first move must be in $M$. A play in $M \& N$ consists of a play in $M$ or a play in $N$. The only play in $\mathbf{1}$ is the empty sequence. If $M$ is a negative game, the positive game $\downarrow M = p \oslash M$ prefixes all plays with a single P-move. Each of these operators has a dual acting on positive games.

We also have an operator $(-)^{\perp}$ inverting the role of Player and Opponent, with $A^{\perp} = (M_A, \neg \circ \lambda_A, \neg(b_A), P_A)$. Linear implication can then be derived, with $M \multimap N = N \lhd M^{\perp}$ (this agrees with the definition of $\multimap$ on negative games in [15, 3] etc.).

**Table 1.** Constructions on Games

| | | | |
|---|---|---|---|
| $N \otimes L = (M_{N+L}, \lambda_{N+L}, O, (P_N \| P_L)^{\circledast}_{N \otimes L})$ | | $Q \parr R = (M_{Q+R}, \lambda_{Q+R}, P, (P_Q \| P_R)^{\circledast}_{Q \parr R})$ | |
| $A \oslash N = (M_{A+N}, \lambda_{A+N}, b_A, (P_A \llbracket P_N)^{\circledast}_{A \oslash N})$ | | $A \lhd Q = (M_{A+Q}, \lambda_{A+Q}, b_A, (P_A \llbracket P_Q)^{\circledast}_{A \lhd Q})$ | |
| $N \& L = (M_{N+L}, \lambda_{N+L}, O, P_N +^* P_L)$ | | $Q \oplus R = (M_{Q+R}, \lambda_{Q+R}, P, P_Q +^* P_R)$ | |
| $\uparrow Q$ | $= o \lhd Q$ | $\downarrow N$ | $= p \oslash N$ |
| $\mathbf{1}$ | $= (\emptyset, \emptyset, O, \{\epsilon\})$ | $\mathbf{0}$ | $= (\emptyset, \emptyset, P, \{\epsilon\})$ |

### 2.3 Strategies

As usual we define the notion of strategy as a set of traces.

**Definition 2.** *A* strategy $\sigma$ *for a game* $(M_A, \lambda_A, b_A, P_A)$ *is a subset of* $P_A$ *satisfying:*

- *If* $\sigma = \varnothing$ *then* $b_A = P$*, and if* $\epsilon \in \sigma$ *then* $b_A = O$*.*
- *If* $sa \in \sigma$*, then* $\lambda_A(a) = P$*.*
- *If* $sab \in \sigma$*, then* $s \in \sigma$*.*
- *If* $sa, sb \in \sigma$*, then* $a = b$*.*

**Definition 3.** *A strategy on a game* $A$ *is* total *if it is nonempty and whenever* $s \in \sigma$ *and* $so \in P_A$*, there is some* $p \in M_A$ *such that* $sop \in \sigma$*.*

### 2.4 Imperative Objects as Strategies

Semantics of a full object-oriented language can be given by interpreting types as games and programs as strategies [22]. As an example, we describe the interpretation of a imperative object as a strategy on an appropriate game. We will later see how this object can be represented as a proof in our system.

We shall consider a simple counter object with two methods: a `void press()` method and a `nat read()` method, returning the number of times the `press` method has previously been invoked. For simplicity here, we will allow the `read` method to be called only once, and thus its type may be represented by the game $\mathbf{N}$. The type of `press` — a command that may be repeated indefinitely — may be represented as a negative game $\Sigma^*$ in which Opponent and Player alternately play $q$ and $a$ respectively. To combine these into an object, we use the operator $\otimes$.

The strategy $\mathsf{count} : \Sigma^* \otimes \mathbf{N}$ representing this counter is $\{s \in P_{\Sigma^* \otimes \mathbf{N}} : \beta(s)\}$ where $\beta(s)$ holds if $s = \epsilon$ or $s = tq_1a_1$ or $s = tq_2m_2$ where $s$ contains $m$ occurrences of $a_1$. An example play in $\mathsf{count}$ is

| $\Sigma^* \otimes \mathbf{N}$ | |
|---|---|
| $q$ | O |
| $a$ | P |
| $q$ | O |
| $a$ | P |
| $q$ | O |
| 2 | P |

By contrast with the *history-free* strategies which denote proofs of linear logic in the model of [3], this strategy is *history-sensitive* — the move prescribed by the strategy depends on the entire play so far. It is this property which allows the state of the object to described implicitly, as in e.g. [4].

## 3 The Logic WS

### 3.1 Proof system

We will now describe a proof system in which formulas represent (finite) games, and each proof of a formula represents a (total) strategy on the corresponding game. This logic is *polarised* — positive and negative formulas will represent positive and negative games, respectively.

The positive and negative formulas are defined as follows:

$$\begin{array}{l} P := \mathbf{0} \mid \downarrow N \mid P \parr Q \mid P \oplus Q \mid P \lhd Q \mid P \oslash N \\ N := \mathbf{1} \mid \uparrow P \mid N \otimes M \mid M \& N \mid N \oslash M \mid N \lhd P \end{array}$$

Define an operation $-^\perp$ on formulas (inverting polarity) as follows:

$$\begin{array}{llll} \mathbf{0}^\perp = \mathbf{1} & (P \parr Q)^\perp = P^\perp \otimes Q^\perp & (P \lhd Q)^\perp = P^\perp \oslash Q^\perp \\ \mathbf{1}^\perp = \mathbf{0} & (M \otimes N)^\perp = M^\perp \parr N^\perp & (M \oslash N)^\perp = M^\perp \lhd N^\perp \\ (\downarrow N)^\perp = \uparrow N^\perp & (P \oplus Q)^\perp = P^\perp \& Q^\perp & (P \oslash M)^\perp = P^\perp \lhd M^\perp \\ (\uparrow P)^\perp = \downarrow P^\perp & (M \& N)^\perp = M^\perp \oplus N^\perp & (M \lhd P)^\perp = M^\perp \oslash P^\perp \end{array}$$

Linear implication is defined $M \multimap N = N \lhd M^\perp$.

A *sequent* of WS is a non-empty sequence of formulas $\vdash A_1, \ldots, A_n$. Semantically, the comma $A, B$ will represent a left merge — $A \lhd B$ if $B$ is positive or $A \oslash B$ if $B$ is negative — and is therefore left-associative.

The proof rules for WS are defined in Table 2. Here $M, N$ range over negative formulas, $P, Q$ over positive formulas, $\Gamma, \Delta$ over lists of formulas, and $\Gamma^p, \Delta^p$ over lists of formulas with polarity $p$. The rules are partitioned into *core rules* and *admissible rules* — we will show that any proof is denotationally equivalent to one which uses only the core rules. Note that the core rules have a particular shape: none of them are multiplicative on sequents, all operate on the first (or head) formula of the sequent and the only connectives corresponding to a choice of introduction rule are $\parr$ and $\oplus$. Within this core, therefore, proof search is particularly simple.

The admissible rules include more familiar sequent-calculus style tensor and cut rules. These permit the embedding of multiplicative-additive linear logic by using the lifts to change polarities where necessary, as in [16, 17, 21]. Note that, whilst our notion of polarity differs from that of MALLP, the proof rules for the multiplicative connectives and lifts can be *derived* in $\mathsf{WS}$ (the roles of $\uparrow$ and $\downarrow$ are switched).

The non-core rules have also been chosen to facilitate representing finitary imperative objects in $\mathsf{WS}$. For example, the $\mathsf{P}_{\mathsf{cut}}$ corresponds to composition of functions, the $\mathsf{P}_{\mathsf{mix}}$ to aggregating objects, and $\mathsf{P}_{\mathsf{wk}}$ to hiding part of the object from the outside world.

**Table 2.** Proof rules for $\mathsf{WS}$

Core rules:

$$\mathsf{P1}\ \frac{}{\vdash \mathbf{1}, \Gamma} \qquad \mathsf{P}\oslash\ \frac{\vdash A, N, \Gamma}{\vdash A \oslash N, \Gamma} \qquad \mathsf{P}\triangleleft\ \frac{\vdash A, P, \Gamma}{\vdash A \triangleleft P, \Gamma}$$

$$\mathsf{P}\otimes\ \frac{\vdash M, N, \Gamma \qquad \vdash N, M, \Gamma}{\vdash M \otimes N, \Gamma} \qquad \mathsf{P}\parr_1\ \frac{\vdash P, Q, \Gamma}{\vdash P \parr Q, \Gamma} \qquad \mathsf{P}\parr_2\ \frac{\vdash Q, P, \Gamma}{\vdash P \parr Q, \Gamma}$$

$$\mathsf{P}\&\ \frac{\vdash M, \Gamma \qquad \vdash N, \Gamma}{\vdash M \& N, \Gamma} \qquad \mathsf{P}\oplus_1\ \frac{\vdash P, \Gamma}{\vdash P \oplus Q, \Gamma} \qquad \mathsf{P}\oplus_2\ \frac{\vdash Q, \Gamma}{\vdash P \oplus Q, \Gamma}$$

$$\mathsf{P}\uparrow^+\ \frac{\vdash\uparrow (P \parr Q), \Gamma}{\vdash\uparrow P, Q, \Gamma} \qquad \mathsf{P}\uparrow^-\ \frac{\vdash\uparrow (P \oslash N), \Gamma}{\vdash\uparrow P, N, \Gamma} \qquad \mathsf{P}\uparrow\ \frac{\vdash P}{\vdash\uparrow P}$$

$$\mathsf{P}\downarrow^-\ \frac{\vdash\downarrow (M \otimes N), \Gamma}{\vdash\downarrow M, N, \Gamma} \qquad \mathsf{P}\downarrow^+\ \frac{\vdash\downarrow (N \triangleleft P), \Gamma}{\vdash\downarrow N, P, \Gamma} \qquad \mathsf{P}\downarrow\ \frac{\vdash N}{\vdash\downarrow N}$$

Other rules:

$$\mathsf{P}^{\top}_{\mathbf{1}}\ \frac{\vdash A, \Gamma, \Delta}{\vdash A, \Gamma, \mathbf{1}, \Delta} \qquad \mathsf{P}^{\top}_{\mathbf{0}}\ \frac{\vdash A, \Gamma, \Delta}{\vdash A, \Gamma, \mathbf{0}, \Delta} \qquad \mathsf{P}^{\top}_{\otimes}\ \frac{\vdash A, \Gamma, M, N, \Delta}{\vdash A, \Gamma, M \otimes N, \Delta} \qquad \mathsf{P}^{\top}_{\parr}\ \frac{\vdash A, \Gamma, P, Q, \Delta}{\vdash A, \Gamma, P \parr Q, \Delta}$$

$$\mathsf{P}_{\bot}\ \frac{\vdash\uparrow P, \Gamma}{\vdash\uparrow 0, P, \Gamma} \qquad \mathsf{P}_{\top}\ \frac{\vdash\downarrow N, \Gamma}{\vdash\downarrow 1, N, \Gamma} \qquad \mathsf{P}^{+}_{\mathsf{sym}}\ \frac{\vdash A, \Gamma, P, Q, \Delta}{\vdash A, \Gamma, Q, P, \Delta} \qquad \mathsf{P}^{-}_{\mathsf{sym}}\ \frac{\vdash A, \Gamma, M, N, \Delta}{\vdash A, \Gamma, N, M, \Delta}$$

$$\mathsf{P}_{\mathsf{mix}}\ \frac{\vdash M, \Gamma, \Delta^+ \qquad \vdash N, \Delta^+_1}{\vdash M, \Gamma, N, \Delta^+, \Delta^+_1} \qquad \mathsf{P}_{\mathsf{wk}}\ \frac{\vdash A, \Gamma, M, \Delta}{\vdash A, \Gamma, \Delta} \qquad \mathsf{P}_{\mathsf{str}}\ \frac{\vdash A, \Gamma, \Delta}{\vdash A, \Gamma, P, \Delta}$$

$$\mathsf{P}_{\mathsf{cut}}\ \frac{\vdash A, \Gamma, N^{\bot}, \Gamma_1 \qquad \vdash N, \Delta^+}{\vdash A, \Gamma, \Delta^+, \Gamma_1} \qquad \mathsf{P}_{\mathsf{id}}\ \frac{}{\vdash N, N^{\bot}} \qquad \mathsf{P}^{\top}_{\oplus i}\ \frac{\vdash \Gamma, P_i, \Delta}{\vdash \Gamma, P_1 \oplus P_2, \Delta}$$

### 3.2 Imperative Objects as Proofs in WS

We now show how a bounded version of our counter can be represented in $\mathsf{WS}$.

Write $\top$ for $\downarrow \mathbf{0}$. If $\oplus_0 \top =_{df} \top$ and $\oplus_{n+1} =_{df} \top \oplus (\oplus_n \top)$ then $\mathbf{N}_n =\uparrow (\oplus_n \top)$ represents the type of numbers at most $n$. Similarly, if $!_0 A =_{df} \mathbf{1}$ and $!_{n+1} A =_{df} A \oslash !_n A$ then $!_n \Sigma$ represents a switch that can be pressed at most $n$ times.

We may derive a proof $\mathsf{count}_n \vdash !_n\Sigma \otimes \mathbf{N}_n$ for any $n$ by induction. The crux of the proof is the concluding application of the $\mathsf{P}\otimes$ rule: this partitions interactions in $!_n\Sigma \otimes \mathbf{N}_n$ into those that start in $!_n\Sigma$ (with a `press`) and those that start in $\mathbf{N}_n$ (with a `read`).

$$\mathsf{P}\otimes\ \frac{\overset{\mathsf{count}^1_{n,n}}{\vdots}\ \vdash\ !_n\Sigma, \mathbf{N}_n \qquad \overset{\mathsf{count}^2_{n,n}}{\vdots}\ \vdash\ \mathbf{N}_n, !_n\Sigma}{\mathsf{count}_n \vdash !_n\Sigma \otimes \mathbf{N}_n}$$

We first define $\mathsf{count}^1_{n,m}$ for $m \geq n$ by induction on $n$. The base case is simple. For $\mathsf{count}^1_{n+1,m}$, we have:

$$\mathsf{P}\otimes\ \frac{\mathsf{P}\mathbf{1}\ \dfrac{}{\vdash \mathbf{1}, !_n\Sigma, \mathbf{N}_m} \qquad \mathsf{P}^{\mathsf{T}}_{\mathbf{1}}\ \dfrac{\overset{\mathsf{count}^1_{n,m}}{\vdots}\ \vdash !_n\Sigma, \mathbf{N}_m}{\vdash !_n\Sigma, \mathbf{1}, \mathbf{N}_m}}{\vdash (\mathbf{1}\otimes !_n\Sigma), \mathbf{N}_m} \qquad \mathsf{P}^{\mathsf{T}}_{\otimes}\ \dfrac{\mathsf{P}^{\mathsf{T}}_{\mathbf{1}}\ \dfrac{\overset{\mathsf{count}^2_{n,m}}{\vdots}\ \vdash \mathbf{N}_m, !_n\Sigma}{\vdash \mathbf{N}_m, \mathbf{1}, !_n\Sigma}}{\vdash \mathbf{N}_m, (\mathbf{1}\otimes !_n\Sigma)}$$

$$\mathsf{P}\downarrow\ \frac{\vdash (\mathbf{1}\otimes !_n\Sigma) \otimes \mathbf{N}_m}{\mathsf{P}\downarrow^-\ \dfrac{\vdash\downarrow ((\mathbf{1}\otimes !_n\Sigma) \otimes \mathbf{N}_m)}{\mathsf{P}\downarrow^-\ \dfrac{\vdash\downarrow (\mathbf{1}\otimes !_n\Sigma), \mathbf{N}_m}{\mathsf{P}\oslash\ \dfrac{\vdash\downarrow \mathbf{1}, !_n\Sigma, \mathbf{N}_m}{\mathsf{P}\oslash\ \dfrac{\vdash (\downarrow \mathbf{1}\oslash !_n\Sigma), \mathbf{N}_m}{\mathsf{P}\uparrow\ \dfrac{\vdash (\downarrow \mathbf{1}\oslash !_n\Sigma) \oslash \mathbf{N}_m}{\mathsf{P}\uparrow^-\ \dfrac{\vdash\uparrow ((\downarrow \mathbf{1}\oslash !_n\Sigma) \oslash \mathbf{N}_m)}{\mathsf{P}\uparrow^-\ \dfrac{\vdash\uparrow (\downarrow \mathbf{1}\oslash !_n\Sigma), \mathbf{N}_m}{\mathsf{P}\oslash\ \dfrac{\vdash\uparrow\downarrow \mathbf{1}, !_n\Sigma, \mathbf{N}_m}{\mathsf{count}^1_{n+1,m} \vdash !_{n+1}\Sigma, \mathbf{N}_m}}}}}}}}}$$

For $\mathsf{count}^2_{n,m} \vdash\uparrow (\oplus_n \top), !_n\Sigma$, we first strip away the head $\uparrow$, requiring a proof $\mathsf{count}^3_{n,m} \vdash \oplus_n \top, !_n\Sigma$. This proof uses the difference between $m$ and $n$ to determine the result of `read`.

$$\mathsf{P}\oplus_2\ \frac{\overset{\mathsf{count}^3_{n,n+a}}{\vdots}\ \vdash \oplus_{n+a} \top, !_n\Sigma}{\mathsf{count}^3_{n,n+a+1} \vdash \top \oplus (\oplus_{n+a} \top), !_n\Sigma}$$

$$\mathsf{P}\oplus_1\ \frac{\mathsf{P}\mathbf{1}\ \dfrac{}{\vdash \mathbf{1}, !_{n+1}\Sigma} \qquad \mathsf{P}^{\mathsf{T}}_{\mathbf{1}}\ \dfrac{\overset{!\mathsf{com}_{n+1}}{\vdots}\ \vdash !_{n+1}\Sigma}{\vdash !_{n+1}\Sigma, \mathbf{1}}}{\mathsf{P}\otimes\ \dfrac{\vdash \mathbf{1}\otimes !_{n+1}\Sigma}{\mathsf{P}\downarrow\ \dfrac{\vdash\downarrow (\mathbf{1}\otimes !_{n+1}\Sigma)}{\mathsf{P}\downarrow^-\ \dfrac{\vdash\downarrow \mathbf{1}, !_{n+1}\Sigma}{\mathsf{count}^3_{n+1,n+1} \vdash \top \oplus (\oplus_n \top), !_{n+1}\Sigma}}}}$$

We omit the simple proofs $\mathsf{count}^3_{0,0}$ and $!\mathsf{com}_n$.

# 4 Categorical Semantics for WS

We now describe a categorical model of WS, together with a principal, motivating example based on games and strategies. It will be based on the notion of *sequoidal category* [14]. We use notation $\eta : F \Rightarrow G : \mathcal{C} \to \mathcal{D}$ to mean $\eta$ is a natural transformation from $F$ to $G$ with $F, G : \mathcal{C} \to \mathcal{D}$.

## 4.1 WS-categories

**Definition 4.** *A* sequoidal category *consists of:*

- *A symmetric monoidal category* $(\mathcal{C}, I, \otimes)$ *(we will call the relevant isomorphisms* $\mathsf{assoc}$*,* $\mathsf{li}$*,* $\mathsf{ri}$ *and* $\mathsf{sym}$*)*
- *A category* $\mathcal{C}_s$
- *A right-action* $\oslash$ *of* $\mathcal{C}$ *on* $\mathcal{C}_s$ *(that is, a functor* $\_ \oslash \_ : \mathcal{C}_s \times \mathcal{C} \to \mathcal{C}_s$ *with natural isomorphisms* $\mathsf{unit} : A \oslash I \cong A$ *and* $\mathsf{pasc} : A \oslash (B \otimes C) \cong (A \oslash B) \oslash C$ *satisfying some coherence conditions [10])*
- *A functor* $J : \mathcal{C}_s \to \mathcal{C}$
- *A natural transformation* $\mathsf{wk} : J(\_) \otimes \_ \Rightarrow J(\_ \oslash \_)$ *satisfying some coherence conditions [12]*

An inclusive sequoidal category is a sequoidal category in which $\mathcal{C}_s$ is a full-on-objects subcategory of $\mathcal{C}$ containing the monoidal isomorphisms and $\mathsf{wk}$; and $J$ is the inclusion functor.

We can identify this structure in a category of games, based on the constructions in Section 2. Let $\mathcal{G}$ be the category whose objects are negative games and an arrow $M \to N$ is a strategy on $M \multimap N$. Composition of strategies is by "parallel composition plus hiding", and identities are copycat strategies, as defined in [3]. In fact this category has been studied extensively in e.g. [15, 6, 19], and has equivalent presentations using *graph games* [9] and *locally Boolean domains* [13]. $\mathcal{G}$ can be enriched with symmetric monoidal structure based on the tensor $\otimes$ and unit $I$.

A strategy on $M \multimap N$ is *strict* if it responds to any first move in $N$ with some move in $M$ (if at all). Define $\mathcal{G}_s$ to be the subcategory of $\mathcal{G}$ consisting of only the strict strategies. Then we can extend the left-merge operator $\oslash$ to an action $\mathcal{G}_s \times \mathcal{G} \to \mathcal{G}_s$. There is a natural copycat strategy $\mathsf{wk} : M \otimes N \to M \oslash N$ in $\mathcal{G}_s$ satisfying the required axioms [14], and we have our sequoidal structure.

**Definition 5.** *An inclusive sequoidal category is* Cartesian *if* $\mathcal{C}$ *is Cartesian and* $\mathcal{C}_s$ *is a sub-Cartesian Category (we will write* $t_A$ *for the unique map* $A \to 1$*.) It is* decomposable *if the natural transformations* $\mathsf{dec} = \langle \mathsf{wk}, \mathsf{wk} \circ \mathsf{sym} \rangle : A \otimes B \Rightarrow (A \oslash B) \times (B \oslash A) : \mathcal{C}_s \times \mathcal{C}_s \to \mathcal{C}_s$ *and* $\mathsf{dec}^0 = \mathsf{t}_I : I \Rightarrow 1 : \mathcal{C}_s$ *are isomorphisms (so, in particular,* $(\mathcal{C}, \otimes, I)$ *is an affine SMC).*

*A Cartesian sequoidal category is* distributive *if the natural transformations* $\mathsf{dist} = \langle \pi_1 \oslash \mathsf{id}_C, \pi_2 \oslash \mathsf{id}_C \rangle : (A \times B) \oslash C \Rightarrow (A \oslash C) \times (B \oslash C) : \mathcal{C}_s \times \mathcal{C}_s \times \mathcal{C} \to \mathcal{C}_s$ *and* $\mathsf{dist}^0 = \mathsf{t}_{1 \oslash C} : 1 \oslash C \Rightarrow 1 : \mathcal{C} \to \mathcal{C}_s$ *are isomorphisms.*

$M\&N$ is a product of $M$ and $N$ in our category of games, and the empty game $I$ is a terminal object. In our sequoidal categories the decomposability and distributivity isomorphisms above exist as natural copycat morphisms [14].

**Definition 6.** *A* sequoidal closed category *is an inclusive sequoidal category where $\mathcal{C}$ is symmetric monoidal closed and the map $f \mapsto \Lambda(f \circ \mathsf{wk})$ defines a natural isomorphism $\Lambda_s : \mathcal{C}_s(B \oslash A, C) \Rightarrow \mathcal{C}_s(B, A \multimap C)$. Define $\mathsf{app}_s : (A \multimap B) \oslash A \to B$ as $\Lambda_s^{-1}(\mathsf{id})$.*

In a symmetric monoidal closed category, if $f : A \to B$ let $\Lambda_I(f) : I \to A \multimap B$ denote the name of $f$, and $\Lambda_I^{-1}$ the inverse operation. We can show that $\mathcal{G}$ is sequoidal closed, with the internal hom given by $\multimap$.

**Proposition 1.** *In any sequoidal closed category, $\multimap$ restricts to a functor $\mathcal{C}^{\mathsf{op}} \times \mathcal{C}_s \to \mathcal{C}_s$ with isomorphisms $\mathsf{unit}_\multimap : I \multimap A \cong A$ and $\mathsf{pasc}_\multimap : A \otimes B \multimap C \cong A \multimap (B \multimap C)$ in $\mathcal{C}_s$.*

*Proof.* We need to show that if $g$ is in $\mathcal{C}_s$ then $f \multimap g$ is in $\mathcal{C}_s$. This follows from the fact that $f \multimap g = \Lambda_s(g \circ \mathsf{app}_s \circ (\mathsf{id} \oslash f))$. We can define $\mathsf{unit}_\multimap : I \multimap A \cong A$ by $\mathsf{unit}_\multimap = \mathsf{app}_s \circ \mathsf{unit}^{-1}$ and $\mathsf{unit}_\multimap^{-1} = \Lambda_s(\mathsf{unit})$. We can define $\mathsf{pasc}_\multimap : A \otimes B \multimap C \cong A \multimap (B \multimap C)$ by $\Lambda_s\Lambda_s(\mathsf{app}_s \circ \mathsf{pasc}^{-1})$ and $\mathsf{pasc}_\multimap^{-1} = \Lambda_s(\mathsf{app}_s \circ (\mathsf{app}_s \oslash \mathsf{id}) \circ \mathsf{pasc})$. □

We have yet to discuss representation of positive games in our categorical model. We will exploit the fact that a strategy on the positive game $P$ is precisely a strategy on the negative game $P^\perp \multimap o$. The object $o$ satisfies an additional special property: an internalised version of *linear functional extensionality* [2]. This property differentiates our model from that of Conway games and models without local alternation [12].

**Definition 7.** *An object $o$ in a sequoidal closed category satisfies* linear functional extensionality *if the natural transformation $\mathsf{lfe} : (B \multimap o) \oslash A \Rightarrow (A \multimap B) \multimap o : \mathcal{C} \times \mathcal{C}^{\mathsf{op}} \to \mathcal{C}_s$ given by $\Lambda_s(\mathsf{app}_s \circ (\mathsf{id} \oslash \mathsf{app}) \circ \mathsf{pasc}^{-1})$ is an isomorphism.*

**Definition 8.** *A* WS-category *is a distributive, decomposable sequoidal closed category with an object $o$ satisfying linear functional extensionality.*

$\mathcal{G}$ is a WS-category. We can also consider the subcategory $\mathcal{G}_f$ of finite negative games and total strategies. The sequoidal structure on $\mathcal{G}$ and $\mathcal{G}_s$ restricts to $\mathcal{G}_f$ and its subcategory of strict and total strategies $\mathcal{G}_{f,s}$, providing another instance of a WS-category.

**Proposition 2.** *$\mathcal{G}$ and $\mathcal{G}_f$ both support the structure of a WS-category.*

Previous work of the second author [14] shows that if a WS-category has a well-behaved fixed point of $A \oslash \_$ for any $A$, then one can give a fully abstract model of a functional language with coroutines in that category.

### 4.2 Semantics of Formulas and Sequents

Let $\mathcal{C}$ be a WS-category. We give semantics of both positive and negative formulas as objects in $\mathcal{C}$ below. Note that in our semantics of formulas, $[\![A]\!] = [\![A^{\perp}]\!]$. However, the polarity of a formula will affect the type of proofs of that formula, as will be seen.

$$\begin{array}{ll|ll} [\![0]\!] & = I & [\![1]\!] & = I \\ [\![\uparrow N]\!] & = [\![N]\!] \multimap o & [\![\downarrow P]\!] & = [\![P]\!] \multimap o \\ [\![M \otimes N]\!] & = [\![M]\!] \otimes [\![N]\!] & [\![P \parr Q]\!] & = [\![P]\!] \otimes [\![Q]\!] \\ [\![M \& N]\!] & = [\![M]\!] \times [\![N]\!] & [\![P \oplus Q]\!] & = [\![P]\!] \times [\![Q]\!] \\ [\![M \oslash N]\!] & = [\![M]\!] \oslash [\![N]\!] & [\![P \lhd Q]\!] & = [\![P]\!] \oslash [\![Q]\!] \\ [\![M \lhd Q]\!] & = [\![Q]\!] \multimap [\![M]\!] & [\![P \oslash N]\!] & = [\![N]\!] \multimap [\![P]\!] \end{array}$$

We consider our list-connective comma to be a binary operator associating to the left. Then $[\![A, B]\!]$ is $[\![A]\!] \oslash [\![B]\!]$ if $A$ and $B$ are of the same polarity, and $[\![B]\!] \multimap [\![A]\!]$ otherwise.

### 4.3 Semantics of Contexts

A context is a nonempty list of formulas. If $\Gamma$ is a context, we give semantics $[\![\Gamma]\!]^b$ for $b \in \{+, -\}$ as endofunctors on $\mathcal{C}_s$ below.

$$\begin{array}{ll|ll} [\![\epsilon]\!]^+ & = \mathsf{id}_{\mathcal{C}_s} & [\![\epsilon]\!]^- & = \mathsf{id}_{\mathcal{C}_s} \\ [\![\Gamma, M]\!]^+ & = [\![M]\!] \multimap [\![\Gamma]\!]^+ & [\![\Gamma, P]\!]^- & = [\![P]\!] \multimap [\![\Gamma]\!]^- \\ [\![\Gamma, P]\!]^+ & = [\![\Gamma]\!]^+ \oslash [\![P]\!] & [\![\Gamma, M]\!]^- & = [\![\Gamma]\!]^- \oslash [\![M]\!] \end{array}$$

We have $[\![A, \Gamma]\!] = [\![\Gamma]\!]^b([\![A]\!])$ where $b$ is the polarity of $A$. We can construct isomorphisms $\mathsf{dist}_{\mathsf{b},\Gamma} : [\![\Gamma]\!]^b(A \times B) \cong [\![\Gamma]\!]^b(A) \times [\![\Gamma]\!]^b(B)$ and $\mathsf{dist}^{\mathsf{0}}_{\mathsf{b},\Gamma} : [\![\Gamma]\!]^b(\mathbf{1}) \cong \mathbf{1}$ by induction on $\Gamma$, with $[\![\Gamma]\!]^b(\pi_1) \circ \mathsf{dist}^{-1}_{b,\Gamma} = \pi_1$.

### 4.4 Semantics of Proofs

While the semantics of formulas are independent of polarity, semantics of proofs are not. If $p \vdash A, \Gamma$ is a proof, we define $[\![p \vdash A, \Gamma]\!]$ as an arrow $\mathcal{C}(I, [\![A, \Gamma]\!])$ in the case that $A$ is negative, and as an arrow in $\mathcal{C}([\![A, \Gamma]\!], o)$ in the case that $A$ is positive. Semantics of the core rules and $\mathsf{P_{cut}}$ are given in Table 3 (interpretation of the remaining admissible rules in a WS-category is omitted for the sake of brevity).

In the semantics of $\mathsf{P_{cut}}$ we use an additional construction. If $\tau : I \to [\![N, \Delta]\!]$ define (strict) $\tau^{\circ -}_{M,\Gamma} : [\![M, \Gamma, N^{\perp}]\!] \to [\![M, \Gamma, \Delta]\!]$ to be $\mathsf{unit}_{\multimap} \circ (\tau \multimap \mathsf{id}_{[\![M,\Gamma]\!]})$ if $|\Delta| = 0$ and $\mathsf{pasc}^n_{\multimap} \circ (\Lambda^{-n}\Lambda_I^{-1}\tau \multimap \mathsf{id}_{[\![M,\Gamma]\!]})$ if $|\Delta| = n + 1$. Define (strict) $\tau^{\circ +}_{P,\Gamma} : [\![P, \Gamma, \Delta]\!] \to [\![P, \Gamma, N^{\perp}]\!]$ to be $(\mathsf{id}_{[\![P,\Gamma]\!]} \oslash \tau) \circ \mathsf{unit}^{-1}$ if $|\Delta| = 0$ and $(\mathsf{id} \oslash \Lambda^{-n}\Lambda_I^{-1}\tau) \circ ((\mathsf{id}_{[\![P,\Gamma]\!]} \oslash \mathsf{sym}) \circ \mathsf{pasc}^{-1})^n$ if $|\Delta| = n + 1$.

**Table 3.** Categorical Semantics for WS (core rules, plus cut)

$$\mathsf{P1}\ \frac{}{[\![\Gamma]\!]^-(\mathsf{dec}^0)^{-1}\circ(\mathsf{dist}^0_{-,\Gamma})^{-1}\circ\mathsf{t} : [\![\vdash \mathbf{1},\Gamma]\!]}$$

$$\mathsf{P}\otimes\ \frac{\sigma : [\![\vdash M,N,\Gamma]\!] \qquad \tau : [\![\vdash N,M,\Gamma]\!]}{[\![\Gamma]\!]^-(\mathsf{dec}^{-1})\circ\mathsf{dist}^{-1}_{-,\Gamma}\circ\langle\sigma,\tau\rangle : [\![\vdash M\otimes N,\Gamma]\!]} \qquad \mathsf{P}\&\ \frac{\sigma : [\![\vdash M,\Gamma]\!] \qquad \tau : [\![\vdash N,\Gamma]\!]}{\mathsf{dist}^{-1}_{-,\Gamma}\circ\langle\sigma,\tau\rangle : [\![\vdash M\&N,\Gamma]\!]}$$

$$\mathsf{P}\parr_2\ \frac{\sigma : [\![\vdash Q,P,\Gamma]\!]}{\sigma\circ[\![\Gamma]\!]^+(\mathsf{wk}\circ\mathsf{sym}) : [\![\vdash P\parr Q,\Gamma]\!]} \qquad \mathsf{P}\parr_1\ \frac{\sigma : [\![\vdash P,Q,\Gamma]\!]}{\sigma\circ[\![\Gamma]\!]^+(\mathsf{wk}) : [\![\vdash P\parr Q,\Gamma]\!]}$$

$$\mathsf{P}\oplus_1\ \frac{\sigma : [\![\vdash P,\Delta]\!]}{\sigma\circ[\![\Delta]\!]^+(\pi_1) : [\![\vdash P\oplus Q,\Delta]\!]} \qquad \mathsf{P}\oplus_2\ \frac{\sigma : [\![\vdash Q,\Delta]\!]}{\sigma\circ[\![\Delta]\!]^+(\pi_2) : [\![\vdash P\oplus Q,\Delta]\!]}$$

$$\mathsf{P}\uparrow^+\ \frac{\sigma : [\![\vdash\uparrow(P\parr Q),\Gamma]\!]}{[\![\Gamma]\!]^-(\mathsf{pasc}_{\multimap}\circ(\mathsf{sym}\multimap\mathsf{id}))\circ\sigma : [\![\vdash\uparrow P,Q,\Gamma]\!]} \qquad \mathsf{P}\uparrow\ \frac{\sigma : [\![\vdash P]\!]}{\Lambda_I(\sigma) : [\![\vdash\uparrow P]\!]}$$

$$\mathsf{P}\uparrow^-\ \frac{\sigma : [\![\vdash\uparrow(P\oslash N),\Gamma]\!]}{[\![\Gamma]\!]^-(\mathsf{Ife}^{-1})\circ\sigma : [\![\vdash\uparrow P,N,\Gamma]\!]} \qquad \mathsf{P}\downarrow\ \frac{\sigma : [\![\vdash N]\!]}{\mathsf{unit}_{\multimap}\circ(\sigma\multimap\mathsf{id}) : [\![\vdash\downarrow N]\!]}$$

$$\mathsf{P}\downarrow^-\ \frac{\sigma : [\![\vdash\downarrow(M\otimes N),\Gamma]\!]}{\sigma\circ[\![\Gamma]\!]^+((\mathsf{sym}\multimap\mathsf{id})\circ\mathsf{pasc}^{-1}_{\multimap}) : [\![\vdash\downarrow M,N,\Gamma]\!]} \qquad \mathsf{P}\downarrow^+\ \frac{\sigma : [\![\vdash\downarrow(N\triangleleft P),\Gamma]\!]}{\sigma\circ[\![\Gamma]\!]^+(\mathsf{Ife}) : [\![\vdash\downarrow N,P,\Gamma]\!]}$$

$$\mathsf{P}_{\mathsf{cut}}\ \frac{\sigma : [\![\vdash M,\Gamma,N^\perp,\Gamma_1]\!] \qquad \tau : [\![\vdash N,\Delta^+]\!]}{[\![\Gamma_1]\!]^-(\tau^{\circ-}_{M,\Gamma})\circ\sigma : [\![\vdash M,\Gamma,\Delta^+,\Gamma_1]\!]} \qquad \mathsf{P}\triangleleft\ \frac{\sigma : [\![\vdash A,P,\Gamma]\!]}{\sigma : [\![\vdash A\triangleleft P,\Gamma]\!]}$$

$$\mathsf{P}_{\mathsf{cut}}\ \frac{\sigma : [\![\vdash P,\Gamma,N^\perp,\Gamma_1]\!] \qquad \tau : [\![\vdash N,\Delta^+]\!]}{\sigma\circ[\![\Gamma_1]\!]^+(\tau^{\circ+}_{P,\Gamma}) : [\![\vdash P,\Gamma,\Delta^+,\Gamma_1]\!]} \qquad \mathsf{P}\oslash\ \frac{\sigma : [\![\vdash A,N,\Gamma]\!]}{\sigma : [\![\vdash A\oslash N,\Gamma]\!]}$$

## 5 Full completeness

We now prove a *full completeness* result for the model of WS in $\mathcal{G}_f$: every total strategy on a game denoted by a formula is the denotation of a unique *core* proof of that formula (i.e. one which only uses the core rules). This exhibits a strong correspondence between syntax and semantics, and establishes admissibility of all non-core rules.

We first give categorical axioms in the style of [2], capturing the properties of a WS-category which enable us to prove full completeness and observe that they are satisfied by $\mathcal{G}_f$. Rather than identifying a class of categorical models with many or varied examples (precluded by the strength of the result itself), these axioms allow us to give a rigorous and abstract proof of full completeness using the structure of a WS-category.

**Definition 9.** *A* complete *WS-category is a WS-category such that:*

**1a** *The unique map* $i : \varnothing \Rightarrow \mathcal{C}(I,o)$ *is a bijection.*
**1b** *The map* $\mathsf{d} = [\lambda f.f\circ\pi_1, \lambda g.f\circ\pi_2] : \mathcal{C}(M,o)+\mathcal{C}(N,o) \Rightarrow \mathcal{C}(M\times N,o)$ *is a bijection. (*$\pi$-atomicity *[2]).*
**2** *The map* $\_\multimap o : \mathcal{C}(I,M) \Rightarrow \mathcal{C}(M\multimap o, I\multimap o)$ *is a bijection.*

These axioms capture the properties of determinacy, totality and the object $o$ — it is straightforward to show:

**Proposition 3.** *$\mathcal{G}_f$ is a complete WS-category.*

We can then prove the following full completeness result.

**Theorem 1.** *In any complete WS-category, if $\sigma : [\![\vdash \Gamma]\!]$ then $\sigma$ is the denotation of a unique core proof $\mathsf{reify}_\Gamma(\sigma) \vdash \Gamma$.*

Informally, $\mathsf{reify}$ may be seen as a kind of semantics-guided proof search procedure: given a strategy $\sigma$, $\mathsf{reify}$ finds a proof which denotes it. It is defined by considering the outermost connective in the head formula of $\Gamma$, to which the core rule is always applied:

- If the head formula is a negative unit, then $\sigma$ is the unique strategy on the terminal object. By axiom $(1a)$ it cannot be a positive unit.
- If the outermost connective is $\otimes$ or $\&$, then we may reverse the associated rule (which corresponds to a bijection which holds for *any* WS-category) to decompose the head formula.
- If the outermost connective is $\oplus$ — i.e. $\Gamma = P_1 \oplus P_2, \Gamma'$ – then by axiom $(1b)$, $\sigma$ corresponds to a projection $\pi_i$ (representing a unique choice of introduction rule for $\oplus$) composed with a strategy on $[\![\vdash P_i, \Gamma']\!]$. If the outer connective in $\parr$, then similar reasoning applies.
- If the outermost connective is a lift ($\uparrow$ or $\downarrow$) then we may force it outwards through $\Gamma$ using the rules $\uparrow^-, \uparrow^+, \downarrow^-, \downarrow^+$ (which correspond to isomorphisms in any WS-category) until we obtain a sequent consisting of a single, lifted formula. This lift may be "reversed" either by applying axiom (2) (for $\downarrow$) or by definition of the semantics ($\uparrow$).

Formally, $\mathsf{reify}_\Gamma$ is defined inductively in Table 4.

**Proposition 4.** *$\mathsf{reify}_\Gamma$ is a well-defined, terminating procedure.*

*Proof.* We define a measure $\mathsf{fd}$ on formulas by

- $\mathsf{fd}(\mathbf{0}) = \mathsf{fd}(\mathbf{1}) = 2$
- $\mathsf{fd}(\uparrow A) = \mathsf{fd}(\downarrow A) = 2 + \mathsf{fd}(A)$
- $\mathsf{fd}(A \odot B) = (\mathsf{fd}(A) \times \mathsf{fd}(B)) + 1$ for $\odot \in \{\oplus, \&, \oslash, \triangleleft, \otimes, \parr\}$

We extend this to sequents inductively with $\mathsf{fd}(\Gamma, A) = (\mathsf{fd}(\Gamma) \times \mathsf{fd}(A)) + 1$. We then define $\ell(A, \Gamma) = \langle \mathsf{fd}(A, \Gamma), |A| \rangle$ where $|A|$ is the size of $A$, and consider the well-founded lexicographical ordering on $\mathbb{N}^2$. Informally, reducing $\ell$ consists of (first priority) moving lifts outwards, and (second priority) simplifying the head formula. Then $\mathsf{reify}_\Gamma$ is defined by induction on $\ell(\Gamma)$. It is routine to see in each case the call to the inductive hypothesis decreases the $\ell$-measure (note that if $M$ is a proper subformula of $N$ then $\mathsf{fd}(M) < \mathsf{fd}(N)$ and $\_, \Gamma$ is monotonic with respect to $\mathsf{fd}$). ◻

**Table 4.** Reification of Strategies as Core Proofs

$$
\begin{array}{ll}
\mathsf{reify}_{\mathbf{1},\Gamma}(\sigma) & = \mathsf{P}\mathbf{1} \\
\mathsf{reify}_{\uparrow P}(\sigma) & = \mathsf{P}\uparrow(\mathsf{reify}_P(\Lambda_I^{-1}(\sigma))) \\
\mathsf{reify}_{\uparrow P,Q,\Gamma}(\sigma) & = \mathsf{P}\uparrow^+(\mathsf{reify}_{\uparrow(P\parr Q),\Gamma}([\![\Gamma]\!]^-((\mathsf{sym}\multimap\mathsf{id})\circ\mathsf{pasc}_{\multimap}^{-1}\circ\sigma))) \\
\mathsf{reify}_{\uparrow P,N,\Gamma}(\sigma) & = \mathsf{P}\uparrow^-(\mathsf{reify}_{\uparrow(P\oslash N),\Gamma}([\![\Gamma]\!]^-(\mathsf{lfe}))) \\
\mathsf{reify}_{M\&N,\Gamma}(\sigma) & = \mathsf{P}\&(\mathsf{reify}_{M,\Gamma}(\pi_1\circ\mathsf{dist}_{-,\Gamma}\circ\sigma))(\mathsf{reify}_{N,\Gamma}((\pi_2\circ\mathsf{dist}_{-,\Gamma}\circ\sigma)) \\
\mathsf{reify}_{M\otimes N,\Gamma}(\sigma) & = \mathsf{P}\otimes(\mathsf{reify}_{M,N,\Gamma}(\pi_1\circ\sigma'),\mathsf{reify}_{N,M,\Gamma}(\pi_2\circ\sigma')) \\
 & \quad \text{where } \sigma' = \mathsf{dist}_{-,\Gamma}\circ[\![\Gamma]\!]^-(\mathsf{dec})\circ\sigma \\
\mathsf{reify}_{A\oslash N,\Gamma}(\sigma) & = \mathsf{P}\oslash(\mathsf{reify}_{A,N,\Gamma}(\sigma)) \\
\mathsf{reify}_{A\triangleleft P,\Gamma}(\sigma) & = \mathsf{P}\triangleleft(\mathsf{reify}_{A,P,\Gamma}(\sigma)) \\
\mathsf{reify}_{\downarrow N}(\sigma) & = \mathsf{P}\downarrow(\mathsf{reify}_N((\_\multimap o)^{-1}(\mathsf{unit}_{\multimap}^{-1}\circ\sigma))) \\
\mathsf{reify}_{\downarrow N,M,\Gamma}(\sigma) & = \mathsf{P}\downarrow^-(\mathsf{reify}_{\downarrow(N\otimes M),\Gamma}(\sigma\circ[\![\Gamma]\!]^+(\mathsf{pasc}_{\multimap}\circ(\mathsf{sym}\multimap\mathsf{id})))) \\
\mathsf{reify}_{\downarrow N,P,\Gamma}(\sigma) & = \mathsf{P}\downarrow^+(\mathsf{reify}_{\downarrow(N\triangleleft P),\Gamma}(\sigma\circ[\![\Gamma]\!]^+(\mathsf{lfe}^{-1}))) \\
\mathsf{reify}_{P\oplus Q,\Gamma}(\sigma) & = [\mathsf{P}\oplus_1\circ\mathsf{reify}_{P,\Gamma},\mathsf{P}\oplus_2\circ\mathsf{reify}_{Q,\Gamma}]\circ\mathsf{d}^{-1}(\sigma\circ\mathsf{dist}_{+,\Gamma}^{-1}) \\
\mathsf{reify}_{P\parr Q,\Gamma}(\sigma) & = [\mathsf{P}\parr_1\circ\mathsf{reify}_{P,Q,\Gamma},\mathsf{P}\parr_2\circ\mathsf{reify}_{Q,P,\Gamma}]\circ\mathsf{d}^{-1}(\sigma\circ[\![\Gamma]\!]^+(\mathsf{dec})\circ\mathsf{dist}_{+,\Gamma}^{-1})
\end{array}
$$

We can complete the proof of Theorem 1 by showing that $\mathsf{reify}_\Gamma$ gives an inverse to $[\![-]\!]_\Gamma$:

**Lemma 1.** *1. For all $\sigma : [\![\vdash \Gamma]\!]$ we have $[\![\mathsf{reify}_\Gamma(\sigma)]\!] = \sigma$.*
*2. For any core proof $p \vdash \Gamma$ we have $\mathsf{reify}_\Gamma([\![p]\!]) = p$.*

**Corollary 1.** *In any complete WS-category, morphisms $\mathcal{C}([\![M]\!],[\![N]\!])$ correspond (bijectively) to core proofs of $\vdash N, M^\perp$.*

*Proof.* Such morphisms correspond to $\mathcal{C}(I,[\![M]\!]\multimap[\![N]\!]) = \mathcal{C}(I,[\![M,N^\perp]\!])$. □

**Corollary 2.** *All non-core proof rules are admissible.*

*Proof.* Let $p \vdash \Gamma$ be a proof. We can construct the proof $p' = \mathsf{reify}([\![p]\!]) \vdash \Gamma$ using only the core rules. By Lemma 1, we have $[\![p']\!] = [\![p]\!]$ (and $p'$ is the unique such core proof). □

This yields reduction-free normalisation from proofs to core proofs.

## 6 Cut Elimination

We have shown that the non-core rules are admissible via a reduction-free evaluation with respect to a particular complete WS-category. However we do not know that such a procedure is sound with respect to any other WS-category. We will address this here in the case of (a restricted form of) cut elimination, by defining a corresponding syntactic procedure.

We describe a procedure to transform a core proof of $\vdash A, \Gamma, N^\perp$ and a core proof of $\vdash N, P$ into a core proof of $\vdash A, \Gamma, P$. We proceed by induction. The

interesting cases are the lifts: if $A = \uparrow Q$ and $\Gamma = \epsilon$ then $\vdash \uparrow Q, N^\perp$ must have been concluded from $\vdash Q \parr N^\perp$, i.e. $\vdash Q, N^\perp$ or $\vdash N^\perp, Q$. In the first case we can apply the inductive hypothesis, but in the second case we cannot. We need an auxiliary procedure $\mathsf{cut}_2$ which turns core proofs of $\vdash N^\perp, Q$ and $\vdash N, P$ into a core proof of $\vdash Q \parr P$ (from which we can deduce $\vdash \uparrow Q, P$ as required). If we think of this procedure as a representation of strategy composition, this corresponds to the situation when some player is set to play in $N$ next and so the next observable move could be in $Q$ or $P$.

Some cases of $\mathsf{cut}$ and $\mathsf{cut}_2$ are given in Table 5 using a term notation based on the names of the core proof rules. All other cases just use the inductive hypothesis in an obvious way. We use a third trivial procedure $\mathsf{symP}\parr$ mapping core proofs of $\vdash P \parr Q, \Gamma$ to core proofs of $\vdash Q \parr P, \Gamma$.

**Table 5.** Cut Elimination for Core Rules (key cases)

| | | |
|---|---|---|
| $A$ | $\Gamma\ \mathsf{cut} :\vdash A, \Gamma, N^\perp \times \vdash N, P$ | $\to \vdash A, \Gamma, P$ |
| $\uparrow$ | $\_ \epsilon\ \mathsf{cut}(\mathsf{P} \uparrow^+ (\mathsf{P} \uparrow (\mathsf{P} \parr_1 (y))), g)$ | $= \mathsf{P} \uparrow^+ (\mathsf{P} \uparrow (\mathsf{P} \parr_1 (\mathsf{cut}(y, g))))$ |
| $\uparrow$ | $\_ \epsilon\ \mathsf{cut}(\mathsf{P} \uparrow^+ (\mathsf{P} \uparrow (\mathsf{P} \parr_2 (y))), g)$ | $= \mathsf{P} \uparrow^+ (\mathsf{P} \uparrow (\mathsf{cut}_2(y, g)))$ |
| $\downarrow$ | $\_ \epsilon\ \mathsf{cut}(\mathsf{P} \downarrow^+ (\mathsf{P} \downarrow (\mathsf{P} \lhd (y))), g)$ | $= \mathsf{P} \downarrow^+ (\mathsf{P} \downarrow (\mathsf{P} \lhd (\mathsf{cut}(y, g))))$ |
| $Q$ | $\Gamma\ \mathsf{cut}_2 :\vdash Q, \Gamma, N^\perp \times \vdash Q^\perp, \Gamma^\perp, P$ | $\to \vdash N^\perp \parr P$ |
| $\downarrow$ | $\_ \epsilon\ \mathsf{cut}_2(\mathsf{P} \downarrow^+ (\mathsf{P} \downarrow (\mathsf{P} \lhd y)))(\mathsf{P} \uparrow^+ (\mathsf{P} \uparrow (\mathsf{P} \parr_1 y')))$ | $= \mathsf{symP}\parr(\mathsf{cut}_2(y', y))$ |
| $\downarrow$ | $\_ \epsilon\ \mathsf{cut}_2(\mathsf{P} \downarrow^+ (\mathsf{P} \downarrow (\mathsf{P} \lhd y)))(\mathsf{P} \uparrow^+ (\mathsf{P} \uparrow (\mathsf{P} \parr_2 y')))$ | $= \mathsf{P} \parr_2 (\mathsf{cut}(y', y))$ |

Despite the fact that we are emulating the mechanics of strategy composition in $\mathsf{WS}$, we can show that the procedure is sound with respect to any WS-category.

**Proposition 5.** *In any WS-category,* $[\![\mathsf{cut}(p_1, p_2)]\!] = [\![\mathsf{P}_{\mathsf{cut}}(p_1, p_2)]\!]$.

## 7 Further Directions

We first consider how exponentials can be introduced into $\mathsf{WS}$. There are several different exponentials on games which one might wish to represent. As presented here all formulas represent finite games (total strategies on unbounded games do not compose, in general). The non-repetitive backtracking exponential of Lamarche [15, 6] preserves finiteness, and the categorical properties characterising it can be used to give a variant of $\mathsf{WS}$ with exponentials, with a corresponding full completeness result and contraction as a non-core rule. It is possible to then embed full polarised linear logic [17] in such a system.

Other exponentials do not preserve finiteness of games, for example the least fixed point $!A \cong A \oslash !A$, which can be used to model stateful computation. In this case, totality can be retained by formulating a notion of an *exponential with variable bounds* including formulas such as $!_x A$ representing a family of (finite) games. In this system contraction would be represented by $\vdash !_x A \otimes !_y A \multimap !_{x+y} A$. Alternatively, we may abandon totality, and develop a type theory with fixed

points, defining $!A = \mu X.A \oslash X$. Extending WS to infinite games via such fixed points (cf. [20]) is a future direction of this project. This will allow representation of imperative objects over infinite datatypes in WS, as well as methods that can be used arbitrarily many times. Thus, such an extension would be a natural setting for formal embedding of a programming language such as Lingay [18].

It is possible to extend WS with propositional variables, representing arbitrary games. In such an extension proofs represent total uniform history-sensitive strategies. Finally, we have formalised some of this work in the theorem prover Agda — including WS, its game semantics, and reification of strategies as proofs.

## References


1. S. Abramsky, K. Honda, and G. McCusker. A fully abstract game semantics for general references. In *LICS '98: Proceedings of the 13th Annual IEEE Symposium on Logic in Computer Science*, page 334, Washington, DC, USA, 1998. IEEE Computer Society.
2. Samson Abramsky. Axioms for definability and full completeness. *Proof, language, and interaction: essays in honour of Robin Milner*, pages 55–75, 2000.
3. Samson Abramsky and Radha Jagadeesan. Games and full completeness for multiplicative linear logic. *J. Symb. Logic*, 59(2):543–574, 1994.
4. Samson Abramsky and Guy McCusker. Linearity, Sharing and State: a fully abstract game semantics for Idealized Algol with active expressions: Extended Abstract. *Electronic Notes in Theoretical Computer Science*, 3:2 – 14, 1996. Linear Logic 96 Tokyo Meeting.
5. Andreas Blass. A game semantics for linear logic. *Annals of Pure and Applied Logic*, 56(1-3):183 – 220, 1992.
6. Pierre-Louis Curien. On the symmetry of sequentiality. In *Proceedings of the 9th International Conference on Mathematical Foundations of Programming Semantics*, pages 29–71, London, UK, 1994. Springer-Verlag.
7. J.-Y. Girard. Locus solum: From the rules of logic to the logic of rules. *Mathematical Structures in Computer Science*, 11(03):301–506, 2001.
8. Jaakko Hintikka. Hyperclassical logic (a.k.a. if logic) and its implications for logical theory. *The Bulletin of Symbolic Logic*, 8(3):404–423, 2002.
9. Martin Hyland and Andrea Schalk. Games on graphs and sequentially realizable functionals (extended abstract). *Logic in Computer Science, Symposium on*, 0:257, 2002.
10. G. Janelidze and G. M. Kelly. A note on actions of a monoidal category, 2001.
11. Giorgi Japaridze. Introduction to computability logic. *Annals of Pure and Applied Logic*, 123(1-3):1 – 99, 2003.
12. J. Laird. A categorical semantics of higher order store. In *Proceedings, 9th Conference on Category Theory and Computer Science, CTCS 2002, Electronic Notes in Theoretical Computer Science*. Elsevier, 2002.
13. J. Laird. Locally Boolean domains. *Theoretical Computer Science*, 342(1):132 – 148, 2005. Applied Semantics: Selected Topics.
14. J. Laird. Functional programs as coroutines : A semantic analysis. *Logical Methods in Computer Science*, to appear.
15. F. Lamarche. Sequentiality, games and linear logic. In *Proceedings, CLICS workshop, Aarhus University*. DAIMI-397–II, 1992.

16. François Lamarche. Games semantics for full propositional linear logic. In *LICS*, pages 464–473, 1995.
17. O. Laurent. Polarized games. In *Logic in Computer Science, 2002. Proceedings. 17th Annual IEEE Symposium on*, pages 265–274, 2002.
18. J. Longley and N. Wolverson. Eriskay: a programming language based on game semantics, 2008. Games for Logic and Programming Languages III Workshop.
19. John Longley. Some programming languages suggested by game models (extended abstract). *Electronic Notes in Theoretical Computer Science*, 249:117 – 134, 2009. Proceedings of the 25th Conference on Mathematical Foundations of Programming Semantics (MFPS 2009).
20. G. McCusker. Games and full abstraction for FPC. In *Logic in Computer Science, 1996. LICS '96. Proceedings., Eleventh Annual IEEE Symposium on*, pages 174–183, Jul 1996.
21. Paul-André Melliès and Nicolas Tabareau. Resource modalities in tensor logic. *Annals of Pure and Applied Logic*, 161(5):632 – 653, 2010. The Third workshop on Games for Logic and Programming Languages (GaLoP), Galop 2008.
22. Nicholas Wolverson. *Game Semantics for an object-orientated language*. PhD thesis, University of Edinburgh, 2008.