Imperative Programs as Proofs (via Game Semantics)

Martin Churchill, Jim Laird, Guy McCusker
University of Bath

University of Birmingham, 8th July 2011
Motivation 1: Curry-Howard Correspondence

The Curry-Howard isomorphism notes a striking correspondence between *proofs* and *functional programs*:

<table>
<thead>
<tr>
<th>Types</th>
<th>Propositions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programs</td>
<td>Proofs</td>
</tr>
<tr>
<td>Evaluation</td>
<td>Proof normalisation</td>
</tr>
</tbody>
</table>

- We can extend our notion of programs to include those with imperative effects...
- What are the corresponding proofs?
Motivation 2: A Simple Games Model

- Modelling programs/proofs as *strategies* is a compelling metaphor and has yielded strong technical results.
- ⇒ the games themselves are important mathematical entities.
- Curien-Lamarche sequential games are a strikingly simple formulation
  - Rich mathematical structure, can model many languages and logics
- Can we find a logic where each strategy interprets a proof?
Overview

- We will develop a logic WS1 where formulas correspond to games and proofs to history-sensitive strategies
  - Proofs with imperative computational content
- The system is expressive:
  - This logic contains first-order intuitionistic linear logic
  - We can embed a total imperative programming language
  - We can use it to reason about imperative programs
- This logic admits a strong full completeness result with respect to the game model
Formulas of WS1

- Fix a first-order language $\mathcal{L}$ with pairs of predicates $(\phi, \overline{\phi})$ and a variable set $\mathcal{V} (= \in \phi)$

- For formulas of the logic are as follows:

| $M, N :=$ | 1 | $\bot$ | $\phi($ $\overline{\phi}$ $)$ |
| $M \otimes N$ | $M \ominus N$ | $N \triangleleft P$ |
| $\forall x. P$ | $M \& N$ | $!N$ |

| $P, Q :=$ | 0 | $\top$ | $\overline{\phi}(\overline{\phi})$ |
| $P \otimes Q$ | $P \triangleleft Q$ | $P \ominus N$ |
| $\exists x. P$ | $P \oplus Q$ | $?P$ |

- We have an involutive ($-\bot$) operation switching polarity
- We can encode implication $M \rightarrow N = N \triangleleft M^{\bot}$
Formulas as Games

- Formulas denote (families of) two-player games
  - (indexed over $\mathcal{L}$-structures and valuations)
  - Opponent and Player alternately play moves according to a tree of valid plays
  - In negative formulas Opponent starts, in positive formulas Player starts

- Proofs of a formula denote (families of) winning P-strategies on the interpretation of that formula.
  - Player must always respond to an Opponent-move
  - There is a winning condition for infinite plays
Units and Atoms

\[ M, N := \begin{cases} 1 & | & \bot & | & \phi(\overrightarrow{x}) & | & \ldots \end{cases} \]
\[ P, Q := \begin{cases} 0 & | & \top & | & \phi(\overrightarrow{x}) & | & \ldots \end{cases} \]

- \( 1 \) represents the empty negative game (no moves) (\( \vdash 1 \))
- \( \bot \) represents the game with a single Opponent move (\( \not\vdash \bot \))
- \( \phi(\overrightarrow{x}) \) is interpreted as \( 1 \) if the model validates \( \phi(\overrightarrow{x}) \), \( \bot \) if it does not
Units and Atoms

\[ M, N := 1 | \bot | \phi(\overrightarrow{x}) | \ldots \]

\[ P, Q := 0 | \top | \phi(\overrightarrow{x}) | \ldots \]

- 0 represents the empty positive game (no moves) (\( \not\models 0 \))
- \( \top \) represents the game with a single Player move (\( \models \top \))
- \( \phi(\overrightarrow{x}) \) is interpreted as 0 if the model validates \( \phi(\overrightarrow{x}) \), \( \top \) if it does not
Additives and Quantifiers

\[ M, N := \quad M \& N \quad | \quad \forall x. P \quad | \quad \ldots \]
\[ P, Q := \quad P \oplus Q \quad | \quad \exists x. P \quad | \quad \ldots \]

- In \( M \& N \), Opponent may chose to start a play in \( M \) or in \( N \)
  - So a strategy \( \vdash M \& N \) is a pair \( (\vdash M, \vdash N) \)
- In \( \forall x. M(x) \), Opponent may chose a value \( v \) for \( x \) in the model and start a play in \( M(v) \)
Additives and Quantifiers

\[ M, N := M \& N \mid \forall x. P \mid \ldots \]

\[ P, Q := P \oplus Q \mid \exists x. P \mid \ldots \]

- In \( P \oplus Q \), Player may chose to start a play in \( P \) or in \( Q \)
  - So a strategy \( \vdash P \oplus Q \) is a strategy \( \vdash P \) or a strategy \( \vdash Q \)
- In \( \exists x. P(x) \), Player may chose a value \( v \) for \( x \) in the model and start a play in \( P(v) \)
Multiplicatives

\[
M, N := \quad M \otimes N \quad | \quad M \oslash N \quad | \quad N \triangleleft P \quad | \quad \ldots
\]
\[
P, Q := \quad P \otimes Q \quad | \quad P \triangleleft Q \quad | \quad P \oslash N \quad | \quad \ldots
\]

- A play in \( M \otimes N \) is an **interleaving** of a play in \( M \) with a play in \( N \)
  - Opponent may start in either component, and then switch between components
- A play in \( M \oslash N \) is a play in \( M \otimes N \) that must start in \( M \)
  - So we have \( M \otimes N \cong M \oslash N \triangleleft N \oslash M \)
- In the game \( M \triangleleft P \), it is Player that can switch between the two components.
Multiplicatives

\[ M, N := M \otimes N \quad | \quad M \ominus N \quad | \quad N \triangleleft P \quad | \quad \ldots \]

\[ P, Q := P \oslash Q \quad | \quad P \triangleleft Q \quad | \quad P \oslash N \quad | \quad \ldots \]

▶ A play in \( P \oslash Q \) is an **interleaving** of a play in \( P \) with a play in \( Q \)
  ▶ Player may start in either component, and then switch between components

▶ A play in \( P \triangleleft Q \) is a play in \( P \oslash Q \) that must start in \( P \)
  ▶ So we have \( P \oslash Q \cong P \triangleleft Q \oplus Q \triangleleft P \)

▶ In the game \( P \oslash M \), it is Opponent that can switch between the two components.
Exponentials

\[ M, N ::= !M \ldots \]
\[ P, Q ::= ?P \ldots \]

- \( !M \) denotes an (ordered) interleaving of infinitely many copies of \( M \)
  - Opponent may spawn new copies of \( M \) and switch between copies he has opened (\( !M \simeq M \circ !M \) )

- \( ?P \) denotes an (ordered) interleaving of infinitely many copies of \( P \)
  - Player may spawn new copies of \( P \) and switch between copies he has opened (\( ?P \simeq P \triangleleft ?P \) )

- These are the only operators yielding infinite games
Example

- The game of Booleans can be given by $B = \bot \triangleleft (\top \oplus \top)$
  - A play consists of an O-move ($q$) followed by one of two P-moves ($t$ or $f$)
  - Two winning strategies correspond to True or False values

- We can represent 'functions' Bool → Bool by
  $B \rightarrow B = B \triangleleft B_{\bot} = (\bot \triangleleft (\top \oplus \top)) \triangleleft (\top \oslash (\bot \& \bot))$
Example

- The game of Booleans can be given by $B = \bot \triangleleft (\top \oplus \top)$
  - A play consists of an $O$-move ($q$) followed by one of two $P$-moves ($t$ or $f$)
  - Two winning strategies correspond to True or False values

- We can represent 'functions' $\text{Bool} \rightarrow \text{Bool}$ by
  $B \rightarrow B = B \triangleleft B' = (\bot \triangleleft (\top \oplus \top)) \triangleleft (\top \otimes (\bot \& \bot))$

*Opponent asks for output*
Example

- The game of Booleans can be given by $B = \bot \triangleleft (\top \oplus \top)$
  - A play consists of an O-move ($q$) followed by one of two P-moves ($t$ or $f$)
  - Two winning strategies correspond to True or False values
- We can represent 'functions' $\text{Bool} \to \text{Bool}$ by
  \[
  B \rightarrow B = B \triangleleft B^\bot = (\bot \triangleleft (\top \oplus \top)) \triangleleft (\top \oslash (\bot \& \bot))
  \]

*Player gives output*
Example

- The game of Booleans can be given by $\mathcal{B} = \bot \triangleleft (\top \oplus \top)$
  - A play consists of an O-move ($q$) followed by one of two P-moves ($t$ or $f$)
  - Two winning strategies correspond to True or False values
- We can represent 'functions' $\text{Bool} \to \text{Bool}$ by

$$\mathcal{B} \rightarrow \mathcal{B} = \mathcal{B} \triangleleft \mathcal{B}^\bot = (\bot \triangleleft (\top \oplus \top)) \triangleleft (\top \oslash (\bot \& \bot))$$

or Player asks for input
Example

- The game of Booleans can be given by $B = \bot \triangleleft (\top \oplus \top)$
  - A play consists of an O-move ($q$) followed by one of two P-moves ($t$ or $f$)
  - Two winning strategies correspond to True or False values
- We can represent 'functions' Bool $\rightarrow$ Bool by
  \[
  B \rightarrow B = B \triangleleft B^\bot = (\bot \triangleleft (\top \oplus \top)) \triangleleft (\top \otimes (\bot \& \bot))
  \]

Opponent gives input
Example

- The game of Booleans can be given by $\mathbb{B} = \bot \triangleleft (\top \oplus \top)$
  - A play consists of an O-move ($q$) followed by one of two P-moves ($t$ or $f$)
  - Two winning strategies correspond to True or False values
- We can represent 'functions' $\text{Bool} \rightarrow \text{Bool}$ by
  \[
  \mathbb{B} \rightarrow \mathbb{B} = \mathbb{B} \triangleleft \mathbb{B}^\bot = (\bot \triangleleft (\top \oplus \top)) \triangleleft (\top \oslash (\bot \& \bot))
  \]

*Player gives output*
Sequents

A *sequent* of WS1 is $\Phi \vdash \Gamma$ where:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ $\Phi = X; \Theta$ where $X$ is variables in scope, $\Theta$ is atomic assumptions on those variables.</td>
<td></td>
</tr>
<tr>
<td>▶ $\Gamma$ is a nonempty list of formulas, of either polarity.</td>
<td></td>
</tr>
</tbody>
</table>

$\Phi \vdash M, P, Q, N$

Comma is to be read as a left-associative $\ominus$ or $\ltimes$:

$\Phi \vdash ((M \ltimes P) \ltimes Q) \ominus N$

$\Rightarrow$ First move must occur in first formula.
Core Rules

\[
\begin{align*}
\Phi &\vdash 1, \Gamma \\
\Phi &\vdash M, N, \Gamma \\
\Phi &\vdash M \otimes N, \Gamma \\
\Phi &\vdash M & N, \Gamma \\
\Phi &\vdash P, \Gamma \\
\Phi &\vdash \bot, P, \Gamma \\
\Phi &\vdash \bot, N, \Gamma \\
\Phi &\vdash \top, N \triangleleft P, \Gamma \\
\Phi &\vdash \top, N, P, \Gamma \\
\Phi &\vdash A, P, \Gamma \\
\Phi &\vdash A \triangleleft P, \Gamma
\end{align*}
\]

\[
\begin{align*}
\Phi &\vdash \top \\
\Phi &\vdash Q, P, \Gamma \\
\Phi &\vdash P \otimes Q, \Gamma \\
\Phi &\vdash P \oplus Q, \Gamma \\
\Phi &\vdash \bot, P \otimes Q, \Gamma \\
\Phi &\vdash \bot, P, Q, \Gamma \\
\Phi &\vdash N, \Gamma \\
\Phi &\vdash \top, N \\
\Phi &\vdash \top, \Gamma \\
\Phi &\vdash A, N, \Gamma \\
\Phi &\vdash A \otimes N, \Gamma \\
\Phi &\vdash P, ?P, \Gamma
\end{align*}
\]
Core Rules (atoms and quantifiers)

A proof of $X; \Theta \vdash \Gamma$ is interpreted as a strategy on $\llbracket \Gamma \rrbracket (L)$ for each $\Theta$-satisfying $L$-model over $X$

\[
\frac{\Theta, \bar{\phi}(\vec{x}) \vdash \bot, \Gamma}{\Theta \vdash \phi(\vec{x}), \Gamma} \\
\frac{X \uplus \{x\}; \Theta \vdash N, \Gamma \quad x \not\in FV(\Theta, \Gamma)}{X; \Theta \vdash \forall x. N, \Gamma} \\
\frac{(X; \Theta \vdash \Gamma)[\frac{z}{x}, \frac{z}{y}]}{X; \Theta, x \neq y \vdash \Gamma} \\
\frac{\Theta, \bar{\phi}(\vec{x}) \vdash \top, \Gamma}{\Theta, \bar{\phi}(\vec{x}) \vdash \bar{\phi}(\vec{x}), \Gamma} \\
\frac{X \uplus \{y\}; \Theta \vdash P[y/x], \Gamma}{X \uplus \{y\}; \Theta \vdash \exists x. P, \Gamma} \\
\frac{\Theta, x \neq x \vdash \Gamma}{\Theta, x \neq y \vdash \Gamma}
\]
Other Rules

<table>
<thead>
<tr>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\vdash \Gamma^*, \Delta$</td>
</tr>
<tr>
<td>$\vdash \Gamma^*, 1, \Delta$</td>
</tr>
<tr>
<td>$\vdash \Gamma^*, \Delta$</td>
</tr>
<tr>
<td>$\vdash \Gamma^*, 0, \Delta$</td>
</tr>
<tr>
<td>$\vdash M, \Gamma, \Delta^+$</td>
</tr>
<tr>
<td>$\vdash N, \Delta_1^+$</td>
</tr>
<tr>
<td>$\vdash \Gamma^*, N^\perp, \Gamma_1$</td>
</tr>
<tr>
<td>$\vdash \Gamma^*, \Delta^+, \Gamma_1$</td>
</tr>
<tr>
<td>$\vdash \Gamma, !M, P$</td>
</tr>
</tbody>
</table>
Interpretation of Proofs

- We can interpret proofs as (families of) strategies using the ideas described above.
- Semantics of the ‘other’ rules use the categorical structure of the games model:
  - One may compose strategies $M \rightarrow N$ and $N \rightarrow L$, take the tensor of maps $M \otimes N \rightarrow M' \otimes N'$ and so on.
- We distinguish them from the ‘core’ rules due to full completeness result...
Full Completeness

- We can show that any uniform family of winning finitary strategies is the denotation of a unique analytic proof
- We define a **semantics-guided proof search procedure**:  
  - Choice of rule to prove $\vdash A, \Gamma$ determined by $A$ in most cases  
  - There is a choice in $\&$, $\ominus$, $\exists$ cases; determined by move played by the strategy  
    - But what if there is a different choice in different components?
The interpretations of proofs are \textit{uniform} families of strategies.

- If \((L, \nu) \models \overline{\phi}(\overline{x})\) whenever \((L', \nu') \models \overline{\phi}(\overline{x})\) then \([\Gamma](L', \nu')\) is a subgame of \([\Gamma](L, \nu)\).

- Uniformity means that the strategy on \((L', \nu')\) is the restriction of the strategy on \((L, \nu)\).
Non-example

Consider \( \bot \triangleleft (\overline{\phi} \oplus (\top \otimes \phi)) \) ("excluded middle")

There is a family of winning strategies, but it is not uniform.
Uniformity is formalised using tools from category theory...

- A sequent $X; \Theta \vdash \Gamma$ is interpreted as a functor $M^\Theta_X \to G$
  - $M^\Theta_X$ is the category where objects are $\Theta$-satisfying $\mathcal{L}$-structures and $X$-valuations, and morphisms are functions that preserve positive predicates and valuation ($\Rightarrow$ injective)
  - $G$ is the category of games and strategies

- A proof of $X; \Theta \vdash \Gamma$ is interpreted as a uniform winning strategy on $[X; \Theta \vdash \Gamma]$
  - A *lax-natural transformation* $I \Rightarrow [X; \Theta \vdash \Gamma]$ that is *pointwise winning*
Proposition

Provided Θ is “lean” (contains $x \neq y$ for all distinct $x, y \in X$)

- A uniform winning strategy on $P \oplus Q$ is a uniform winning strategy on $P$ or a uniform strategy on $Q$
- A uniform winning strategy on $P \otimes Q$ is a uniform winning strategy on $P \triangleleft Q$ or a uniform winning strategy on $Q \triangleleft P$
- A uniform winning strategy on $\exists x. P(x)$ corresponds to a choice of a unique variable $y$ (in scope) and uniform winning strategy on $P(y)$.
We can hence define our proof search procedure for bounded strategies:

- Apply match rule to ensure $\Theta$ is lean
- Decompose the head formula using core introduction rules until it is a unit
  - Choices for $\otimes, \oplus, \exists$ determined by strategy
- Consolidate the tail into a single formula using the elimination rules
- Strictly decrease the size of the strategy using the rules that remove the head unit
Some Core Rules (reminder)

\[
\begin{align*}
(\Phi \vdash \Gamma)[\frac{x}{\bar{x}}, \frac{z}{\bar{z}}] & \quad \Phi, x \neq y \vdash \Gamma \\
\hline
\Phi \vdash \Gamma & \quad \Phi \vdash 1, \Gamma \\
\Phi \vdash M, N, \Gamma & \quad \Phi \vdash N, M, \Gamma \\
\hline
\Phi \vdash M \otimes N, \Gamma & \quad \Phi \vdash P \oslash Q, \Gamma \\
\Phi \vdash M, \Gamma & \quad \Phi \vdash P, \Gamma \\
\hline
\Phi \vdash M \& N, \Gamma & \quad \Phi \vdash P \oplus Q, \Gamma \\
\Phi \vdash P & \quad \Phi \vdash P \oslash Q, \Gamma \\
\hline
\Phi \vdash \bot, P, \Gamma & \quad \Phi \vdash \bot, P \oslash Q, \Gamma \\
\Phi \vdash \bot, \Gamma & \quad \Phi \vdash \bot, P, Q, \Gamma \\
\hline
\Phi \vdash N & \quad \Phi \vdash \bot, \Gamma \\
\Phi \vdash \top, N & \quad \Phi \vdash \top, P, \Gamma \\
\hline
\Phi \vdash \top, N \triangleleft P, \Gamma & \quad \Phi \vdash \top, P, \Gamma \\
\Phi \vdash \top, N, P, \Gamma & \quad \Phi \vdash \top, \Gamma \\
\hline
\Phi \vdash A, N, \Gamma & \quad \Phi \vdash A \circ N, \Gamma \\
\Phi \vdash N, !N, \Gamma & \quad \Phi \vdash P, ?P, \Gamma \\
\end{align*}
\]
Full Completeness

Thus, each finitary winning uniform strategy is the denotation of a unique analytic proof.

- In the exponential-free subsystem, the interpretation of any proof is finitary.
- We can ‘normalise’ proofs to analytic proofs via the semantics
  - (unique analytic proof with same semantics)
  - ⇒ all of the ‘other’ rules (e.g. cut) are admissible

This also works for the full system, if we allow normal forms to be infinitary analytic proofs.
Analytic Theorems

- In e.g. ILL, we can reduce any proof to an analytic (cut-free) finite proof, even in the presence of exponentials
- In WS1, the analytic proof may be infinite. Why the weaker situation?
  - In ILL proofs \( \cong \) innocent strategies — a strategy on \(!N\) must act the same way in each thread.
  - In WS1 proofs are history-sensitive — so \(!\) really introduces infinite (possibly non-computable) behaviour
- But we can write proofs which denote infinite (computable, total) behaviour...
Non-core rules for Exponential

To generate a finite proof on a type involving the exponentials, we can use the following proof rule:

\[
\begin{align*}
\vdash & M, P \perp, P \\
\vdash & !M, P
\end{align*}
\]

This represents the fact that:

**Proposition**

In \(G\), \(!M\) is the final coalgebra of the functor \(M \otimes \_\).
Intuitionistic Linear Logic

- We can use this (with contraction) to derive promotion

  ⇒ Embedding of Intuitionistic Linear Logic in WS1

- There are formulas that are not provable in ILL but are provable in WS1 e.g. medial:

  $\vdash ((\alpha \otimes \beta \to \bot) \otimes (\gamma \otimes \delta \to \bot) \to \bot) \to $

  $((\alpha \to \bot) \otimes (\gamma \to \bot) \to \bot) \otimes ((\beta \to \bot) \otimes (\delta \to \bot) \to \bot)$
Boolean Variables

- Let $\mathbf{Bi} = (\bot \& \bot) \triangleleft \top$ (input Boolean)
- $\texttt{!var} = ! (\mathbf{B} \& \mathbf{Bi})$ is a type of reusable Boolean variables (read method and write method)
- We can define a reusable Boolean cell $\vdash \mathbf{B} \rightsquigarrow \texttt{!var}$ using the anamorphism rule and a proof $p \vdash \texttt{var}, \mathbf{B}, \mathbf{B}^\perp$
Boolean Cell — $p$

\[
\begin{align*}
B & \rightarrow (B \land Bi) \otimes B \\
& \quad r \\
& \quad r \\
& \quad b \\
& \quad b \\
& \quad r \\
& \quad b \\
\end{align*}
\]

\[
\begin{align*}
& \quad wb \\
& \quad ok \\
& \quad r \\
& \quad b \\
\end{align*}
\]
Boolean Cell — $\text{ana}(p)$

$\Box \rightarrow (B \land Bi) \uplus B \rightarrow (B \land Bi) \uplus (B \land Bi) \uplus (B \land Bi)$

$\vdots$

$wb$

$ok$

$ok$

$r$

$b$

$\vdots$

We can extend this example to define a Boolean Stack in WS1 ($B \cong \text{pop}$, $Bi \cong \text{push}$). For the “state” we use $!B$.
We can embed a total programming language (TotLang).

- Simply typed lambda calculus
- Ground types: com, nat, var
- Constants: skip, sequencing, ifzero, repeat, 0, suc, assignment, deref, newvar, coroutines, encaps, mkvar

```
add  =  \ m \ n . \ newvar \ x := n in
       repeat \ m \ (x := suc !x) ; !x

newstack  =  encaps (\ g . \ newvar \ x := 0 in \ g \ a \ b) 0
where
  a  =  \ n . \ mkvar \ n (\ m . \ x := suc m)
  b  =  \ n . \ ifzero !x then n else
        (let \ z = !x - 1 in \ x := 0 ; z)
```
Naturals in WS1

- To embed TotLang into WS1 we must add natural numbers to WS1...

\[ N := \omega \mid \ldots \quad P := \bar{\omega} \mid \ldots \]

- \(\omega\) (resp. \(\bar{\omega}\)) denotes the game \(\bot^\omega\) (resp. \(\top^\omega\))

- Proof rules:

\[
\begin{align*}
0 & \vdash \bar{\omega} \\
\text{suc} & \vdash \omega, \bar{\omega} \\
\text{ind} & \vdash P \\
\text{ind} & \vdash P_{\perp}, P
\end{align*}
\]

- Full completeness, normalisation etc extends to this setting
Language Embedding

- We can map types to negative formulas: $\text{com} \mapsto \bot \triangleleft \top$, $\text{nat} \mapsto \bot \triangleleft \omega$, $\text{var} \mapsto B \& Bi$, $A \rightarrow B \mapsto B \triangleleft ? A \bot$, ...

- The lambda calculus part uses the structure of the ILL embedding

- Constants can be mapped to proofs in WS1

The games model of TotLang is fully abstract, resultantly:

- Two programs $M$ and $N$ are observationally equivalent if and only if their representations in WS1 have the same (infinitary) normal form

(we can also embed a call-by-value language with these features)
Formulas as Specifications

Formulas of WS1 are much finer than the programming language types, we can use them to represent specifications on programs.

- Evalation order of arguments
- Number of times an argument is interrogated
- Predicates on ground values

Example:

- Identity specification on nat → nat given by
  \[ \bot \land \top \land \forall x. \bot \land \exists y. y = x \]

Adding function symbols increases expressivity further.
We can use uniformity of the underlying strategies to give refinements on imperative behaviour. E.g...

- Define $B' = ⊥ ⊲ (\bar{\alpha} \oplus \bar{\beta})$, $Bi' = (\alpha \& \beta) ⊲ T$.
- If $\alpha$ and $\beta$ are false, $B' = B$, $Bi' = Bi$
- ... in which case $\textbf{worm} = Bi' \oslash !B'$ represents the type of a “write-once-read-many” Boolean cell.
- But since any proof must be a uniform strategy on all models, any proof of $\textbf{worm}$ must act as a well-behaved Boolean cell.
Data-independence

We can use the first-order structure in a different way to model a data-independent language:

- Interpretation of ground type depends on model
  
  \[ \text{val} = \bot \triangleleft \exists x. \top \]

- Cells of ground type, only operation is equality

- Example program: data-independent set
Further Directions

- Enhancing the logic to be able to specify more interesting properties of more interesting programs
- Introducing propositional variables
  - Ranging over arbitrary games — polymorphism
- Recursive types
  - à la Clairambault — e.g.
    \[ \text{list}(B) = \mu X. \bot \llhd (\top \oplus (\top \otimes (B \otimes X))) \]
- Universality results
  - !N is a universal type in the games model... are the embeddings/retractions definable in the logic?
Thank You

Any questions?