Programming Languages in String Diagrams

Asynchronous Games

Paul-André Melliès

Oregon Summer School in Programming Languages
June 2011
Starting point: game semantics

Every proof of formula $A$ initiates a dialogue where

Proponent tries to convince Opponent

Opponent tries to refute Proponent

An interactive approach to logic and programming languages
A formal proof of the drinker formula

\[
\begin{align*}
A(x_0) & \vdash A(x_0) \quad \text{Axiom} \\
A(x_0) & \vdash A(x_0), \forall x. A(x) \\
\vdash A(x_0), A(x_0) & \Rightarrow \forall x. A(x) \quad \text{Right } \Rightarrow \\
\vdash A(x_0), \exists y. \{ A(y) \Rightarrow \forall x. A(x) \} & \quad \text{Right } \exists, \forall \\
\vdash \forall x. A(x), \exists y. \{ A(y) \Rightarrow \forall x. A(x) \} & \quad \text{Left Weakening } \Rightarrow \\
A(y_0) & \vdash \forall x. A(x), \exists y. \{ A(y) \Rightarrow \forall x. A(x) \} \\
\vdash A(y_0) & \Rightarrow \forall x. A(x), \exists y. \{ A(y) \Rightarrow \forall x. A(x) \} \quad \text{Right } \Rightarrow \\
\vdash \exists y. \{ A(y) \Rightarrow \forall x. A(x) \}, \exists y. \{ A(y) \Rightarrow \forall x. A(x) \} & \quad \text{Right } \exists, \text{Contraction}
\end{align*}
\]
Duality

Proponent Program plays the game \( A \)

Opponent Environment plays the game \( \neg A \)

Negation permutes the rôles of Proponent and Opponent
Duality

Opponent
Environment
plays the game
\neg A

Proponent
Program
plays the game
A

Negation permutes the rôles of Opponent and Proponent
Tensor product

Play the two games in parallel
Sum

Proponent selects one component
Opponent opens as many copies as necessary to beat Proponent
Sequential games

An idea dating back to André Joyal
Traditional game semantics

A proof $\pi$

alternating sequences of moves

A proof $\pi$

Game semantics: an **interleaving** semantics of proofs.
Sequential games

A sequential game \((M, P, \lambda)\) consists of

- \(M\) a set of **moves**,  
- \(P \subseteq M^*\) a set of **plays**,  
- \(\lambda : M \to \{-1, +1\}\) a **polarity** function on moves

such that every play is **alternating** and **starts by Opponent**.

Alternatively, a sequential game is an alternating decision tree.
Sequential games

The boolean game $\mathbb{B}$:

- Player in red
- Opponent in blue

- true
- false
- question
Deterministic strategies

A strategy $\sigma$ is a set of alternating plays of even-length

$$s = m_1 \cdots m_{2k}$$

such that:

- $\sigma$ contains the empty play,
- $\sigma$ is closed by even-length prefix:

$$\forall s, \forall m, n \in M, \quad s \cdot m \cdot n \in \sigma \implies s \in \sigma$$

- $\sigma$ is deterministic:

$$\forall s \in \sigma, \forall m, n_1, n_2 \in M, \quad s \cdot m \cdot n_1 \in \sigma \text{ and } s \cdot m \cdot n_2 \in \sigma \implies n_1 = n_2.$$
Three strategies on the boolean game $B$

Player in red
Opponent in blue

true

false

question
Total strategies

A strategy $\sigma$ is total when

– for every play $s$ of the strategy $\sigma$

– for every Opponent move $m$ such that $s \cdot m$ is a play

there exists a Proponent move $n$ such that $s \cdot m \cdot n$ is a play of $\sigma$. 
Two total strategies on the boolean game $B$

Player in red
Opponent in blue
Tensor product

Given two sequential games $A$ and $B$, define

$$ A \otimes B $$

as the sequential game

$$ M_{A \otimes B} = M_A + M_B $$

$$ \lambda_{A \otimes B} = [ \lambda_A, \lambda_B ] $$

$$ P_{A \otimes B} = P_A \otimes P_B $$

where $P_A \otimes P_B$ denotes the set of alternating plays in $M^*_{A \otimes B}$ obtained by interleaving a play $s \in P_A$ and a play $t \in P_B$. 
Linear implication

Given two sequential games $A$ and $B$, define

$$A \leadsto B$$

as the sequential game

$$M_{A \leadsto B} = M_A + M_B$$

$$\lambda_{A \leadsto B} = \left[ -\lambda_A, \lambda_B \right]$$

$$P_{A \leadsto B} = P_A \leadsto P_B$$

where $P_A \leadsto P_B$ denotes the set of alternating plays in $M_{A \leadsto B}^*$ obtained by interleaving a play $s \in P_A$ and a play $t \in P_B$. 
A category of sequential games

The category \textbf{Games} has

– the sequential games as objects

– the strategies of the sequential game

\begin{align*}
\sigma & \in A \to B \\
\sigma & : A \rightarrow B
\end{align*}
The copycat strategy

The identity map

\[ id_A : A \rightarrow A \]

is the \textit{copycat} strategy

\[ id_A : A \circ A \]

defined as

\[ id_A = \{ s \in P_{A \circ A} \mid s = m_1 \cdot m_1 \cdot m_2 \cdot m_2 \cdots m_k \cdot m_k \} \]
The copycat strategy

\[ A \xrightarrow{id} A \]

\[ m_1 \]
\[ m_2 \]
\[ m_3 \]
\[ \vdots \]
\[ m_k \]
Composition

Given two strategies

\[ A \overset{\sigma}{\rightarrow} B \overset{\tau}{\rightarrow} C \]

the composite strategy

\[ A \overset{\sigma;\tau}{\rightarrow} C \]

is defined as

\[ \sigma;\tau = \left\{ u \in P_{A \rightarrow C} \mid \exists s \in \sigma, \exists t \in \tau \quad u \upharpoonright A = s \upharpoonright A, \quad s \upharpoonright B = t \upharpoonright B, \quad u \upharpoonright C = t \upharpoonright C \right\} \]

The definition of composition is associative.
Illustration

1 \xrightarrow{\text{true}} \begin{array}{c} B \\ \text{question} \\ \text{true} \end{array} \xrightarrow{id} \begin{array}{c} B \\ \text{question} \\ \text{true} \end{array}
Illustration

1 \xrightarrow{\text{false}} \mathbb{B} \xrightarrow{id} \mathbb{B}

question
false

question
false
Illustration

1 \rightarrow \neg B \rightarrow \neg B

question
true

question
false
An important isomorphism

The sequential game

\[( A \otimes B ) \rightarrow C \]

is isomorphic to the sequential game

\[ A \rightarrow ( B \rightarrow C ) \]

for all sequential games \( A, B, C \).

Here, isomorphism means tree isomorphism.
The category of sequential games

Theorem.

The category \textbf{Games} is symmetric monoidal closed.

As such, it defines a model of linear \(\lambda\)-calculus.
Illustration

\[(\text{true} \rightarrow \text{true}) \otimes \text{true} \xrightarrow{\text{eval}} \text{true}\]

\[f : \text{true} \rightarrow \text{true}, \quad x : \text{true} \vdash f(x) : \text{false}\]
Illustration

\[(\mathbb{B} \rightarrow \mathbb{B}) \otimes \mathbb{B} \xrightarrow{\text{eval}} \mathbb{B}\]

\[f : \mathbb{B} \rightarrow \mathbb{B}, \quad x : \mathbb{B} \vdash f(x) : \mathbb{B}\]
Illustration

\[(\mathbb{B} \to \mathbb{B}) \to \mathbb{B} \to \mathbb{B}\]

\[\text{question}\]

\[\text{question}\]

\[\text{question}\]

\[\text{question}\]

\[\text{bool}\]

\[f(\text{bool})\]

\[\text{question}\]

\[\text{question}\]

\[\text{question}\]

\[\text{question}\]

\[f(\text{bool})\]

\[f : \mathbb{B} \to \mathbb{B} \vdash \lambda x. f(x) : \mathbb{B} \to \mathbb{B}\]
Currification

More generally, the transformation of the term

\[\Gamma, x : A \vdash f : B\]

into the term

\[\Gamma \vdash \lambda x. f : A \rightarrow^\circ B\]

does not alter the associated strategy, simply reorganizes it.
Cartesian product

Given two sequential games $A$ and $B$, define

$$A \& B$$

as the sequential game

$$M_{A\&B} = M_A + M_B$$

$$\lambda_{A\&B} = \lambda_A + \lambda_B$$

$$P_{A\&B} = P_A \oplus P_B$$

where $P_A \oplus P_B$ is the coalesced sum of the pointed sets $P_A$ and $P_B$.

This means that every nonempty play in $A\&B$ is either in $A$ or in $B$. 
The cartesian product

For every sequential game \( X \), there exists an isomorphism

\[
X \leadsto A \& B \cong (X \leadsto A) \& (X \leadsto B)
\]

This means that the game \( A \& B \) is the **cartesian product** of \( A \) and \( B \).
The cartesian product

For every game $X$, there exists a bijection between the strategies

$$X \rightarrow A&B$$

and the pair of strategies

$$X \rightarrow A \quad \quad X \rightarrow B.$$
The cartesian product

The cartesian product $A \times B$ of two objects $A$ and $B$ is an object equipped with two maps

$$\pi_1 : A \times B \rightarrow A \quad \pi_2 : A \times B \rightarrow B$$

such that for all diagrams

there exists a unique map $h : X \rightarrow A \times B$ making the diagram commute.
The exponential game

Given a sequential game \( A \), define

\[
!A = \left( \mathbb{N} \times M_A \right)
\]

\[
\lambda !A : (i, m) \mapsto \lambda_A(m)
\]

\[
P !A = !P_A
\]

where \( !P_A \) is the set of alternating plays in \( M !A \) such that for all \( i \in \mathbb{N} \)

1. the restriction \( s \upharpoonright i \) of \( s \) to the \( i \)-th component is a play of \( A \)
2. for every prefix \( u \) of \( s \) : \( u \upharpoonright i = \varepsilon \Rightarrow u \upharpoonright i + 1 = \varepsilon. \)
Two important isomorphisms

Define

\[ A \Rightarrow B = !A \rightarrow B \]

Observe the isomorphism

\[ ! (A \& B) \cong !A \otimes !B \]

and the isomorphism

\[
\begin{align*}
(A \& B) \Rightarrow C &= !(A \& B) \rightarrow C \\
\cong ( !A \otimes !B ) \rightarrow C \\
\cong !A \rightarrow ( !B \rightarrow C ) \\
= A \Rightarrow ( B \Rightarrow C )
\end{align*}
\]
A model of the simply-typed $\lambda$-calculus

Fact.

Every simply-typed $\lambda$-term

\[
x_1:A_1 \ , \ \cdots \ , \ x_k:A_k \vdash M : B
\]

may be interpreted as a strategy in the game

\[
!A_1 \otimes \cdots \otimes !A_k \rightarrow B
\]

Here, the modality $!$ enables to backtrack in the sequential games
\[
f : ! (\mathbb{N} \rightarrow \mathbb{N}) , \quad x : \mathbb{N} \vdash ff(x) : \mathbb{N}
\]
Asynchronous games

An asynchronous reconstruction of the $\lambda$-calculus
Geometry of rewriting

Can we do the same thing for open systems?
The traditional interleaving semantics

The boolean game \( \text{IB} \):
The traditional interleaving semantics

The tensor product of two boolean games $B_1$ et $B_2$: 
A step towards true concurrency: bend the branches!
True concurrency: tile the diagram!
Asynchronous game semantics

A proof $\pi$

trajectories in an asynchronous graph

A proof $\pi$

The phenomenon refined: a truly concurrent semantics of proofs.
Event structures

An event structure \(( M, \leq, \#)\) consists of:

— a partially ordered set \(( M, \leq)\) where

\[
m \downarrow = \{ n \in M \mid n \leq m \}\]

is finite for every event \(m \in M\),

— a binary symmetric irreflexive relation \(\#\) satisfying

\[
\forall m, n, p \in M, \quad m \# n \leq p \Rightarrow m \# p.
\]
The asynchronous graph

A position is a compatible downward-closed subset of $M$.

The asynchronous graph has:

— the finite positions as vertices

— the triples $(x, m, y)$ such that $y = x \cup \{m\}$ as edges $x \rightarrow y$.

The asynchronous graph is pointed with the empty position $*$ as root.
Event structures
Event structures
Asynchronous games

An asynchronous game is an event structure

\((M, \leq, \#)\)

equipped with a polarity function

\[\lambda : M \rightarrow \{-1, +1\}\]

indicating whether a move is Player (+1) or Opponent (−1).
Legal plays

A legal play is a path

\[ * \xrightarrow{m_1} x_1 \xrightarrow{m_2} x_2 \xrightarrow{m_3} \cdots x_{k-1} \xrightarrow{m_k} x_k \]

starting from the empty position \(*\) and satisfying:

\[ \forall i \in [1, \ldots, k] \quad \lambda(m_i) = (-1)^i \]

So, a legal play is alternating and starts by an Opponent move.
Strategies

A strategy is a set of legal plays of even length, such that:

— $\sigma$ contains the empty play,

— $\sigma$ is closed under even-length prefix

$$s \cdot m \cdot n \in \sigma \Rightarrow s \in \sigma,$$

— $\sigma$ is deterministic

$$s \cdot m \cdot n_1 \in \sigma \text{ and } s \cdot m \cdot n_2 \in \sigma \Rightarrow n_1 = n_2.$$

A strategy plays according to the current play.
Innocence: strategies with partial information

**Full definability result** (Hyland, Ong, Nickau 1994)

Innocence characterizes the interactive behavior of \( \lambda \)-terms.

An innocent strategy plays according to the current view.
A positionality theorem for innocent strategies

Starting from a local criterion of innocence
Backward innocence
Forward innocence
Innocent strategies are positional

Definition. A strategy $\sigma$ is **positional** when for every two plays $s_1$ and $s_2$ with same target $x$:

$$s_1 \in \sigma \quad \text{and} \quad s_2 \in \sigma \quad \text{and} \quad s_1 \cdot t \in \sigma \Rightarrow s_2 \cdot t \in \sigma$$

**Theorem** (by an easy diagrammatic proof)
Every innocent strategy $\sigma$ is positional

**More**: An innocent strategy is characterized by the positions it reaches.
An illustration: the strategy \((\text{true } \otimes \text{false})\)
Hence, the schism between games and linear logic

Imagine that a discussion is defined by the set of its final states
Tensorial logic

A primitive logic of tensor and negation
Five primitive components of logic

[1] the negation \( \neg \)

[2] the linear conjunction \( \otimes \)

[3] the repetition modality \( ! \)

[4] the existential quantification \( \exists \)

[5] the least fixpoint \( \mu \)

Logic = Data Structure + Duality
Tensorial logic

\[
\frac{A}{\Gamma, \vdash A} \quad \text{axiom}
\]

\[
\frac{\Gamma \vdash A}{\Gamma, \neg A \vdash} \quad \text{left } \neg
\]

\[
\frac{\Gamma, A, B \vdash C}{\Gamma, A \otimes B \vdash C} \quad \text{left } \otimes
\]

\[
\frac{\Gamma \vdash A}{\Gamma, \text{true} \vdash A} \quad \text{left } \text{true}
\]

\[
\frac{\Gamma \vdash A}{\Gamma, \Delta \vdash B} \quad \text{cut}
\]

\[
\frac{\Gamma, A \vdash A, \Delta \vdash B}{\Gamma, \Delta \vdash B} \quad \text{cut}
\]

\[
\frac{\Gamma, A \vdash}{\Gamma \vdash \neg A} \quad \text{right } \neg
\]

\[
\frac{\Gamma \vdash A}{\Gamma, \Delta \vdash A \otimes B} \quad \text{right } \otimes
\]

\[
\frac{\Gamma \vdash A, \Delta \vdash B}{\vdash \text{true}} \quad \text{right } \text{true}
\]
Tensorial logic

\[
\frac{\Gamma \vdash B}{\Gamma, !A \vdash B} \quad \text{weakening}
\]

\[
\frac{\Gamma \vdash B}{\Gamma, !A \vdash B} \quad \text{dereliction}
\]

\[
\frac{\Gamma, !A \vdash B}{\Gamma \vdash B} \quad \text{contraction}
\]

\[
\frac{\Gamma \vdash A}{!\Gamma \vdash !A} \quad \text{promotion}
\]

\[
\frac{\Gamma, A(x) \vdash B}{\Gamma, \exists x. A \vdash B} \quad \text{left } \exists
\]

\[
\frac{\Gamma \vdash A(t)}{\Gamma \vdash \exists x. A} \quad \text{right } \exists
\]
Tensorial logic

The boolean in linear logic:

\[ B := (1 \oplus 1) \]

The boolean in tensorial logic:

\[ B := \neg \neg (1 \oplus 1) \]

The classical boolean in tensorial logic:

\[ C := \neg ! \neg (1 \oplus 1) \]
The left-to-right implementation of conjunction

\[ B \otimes B \rightarrow B \]

question

answer_1

question

answer_2

answer_1 \land answer_2
The right-to-left implementation of conjunction

\[ B \otimes B \rightarrow B \]

question

answer_1 \land answer_2
A formal proof of the drinker formula

\[
\begin{align*}
A(x_0) \vdash A(x_0) & \quad \text{Axiom} \\
A(x_0) \vdash A(x_0), \forall x. A(x) & \\
\vdash A(x_0), A(x_0) \Rightarrow \forall x. A(x) & \\
\vdash A(x_0), \exists y. \{ A(y) \Rightarrow \forall x. A(x) \} & \\
\vdash \forall x. A(x), \exists y. \{ A(y) \Rightarrow \forall x. A(x) \} & \\
\vdash \forall x. A(x), \exists y. \{ A(y) \Rightarrow \forall x. A(x) \} & \\
\vdash \exists y. \{ A(y) \Rightarrow \forall x. A(x) \} & \\
\end{align*}
\]
The drinker formula in tensorial logic

\[ \forall x \, A(x) \iff \exists y \, \Rightarrow \Rightarrow \]
A proof of the drinker formula in tensorial logic

\[
\begin{array}{c}
A(x_0) \vdash A(x_0) \\
\neg A(x_0), A(x_0) \vdash \\
\neg A(x_0), \ !w A(x_0) \vdash \\
\neg A(x_0), \ !w A(x_0), \ !w \exists x \neg A(x) \vdash \\
\neg A(x_0) \vdash \neg \{ \ !w A(x_0) \otimes \ !w \exists x \neg A(x) \} \\
\neg A(x_0) \vdash \exists y \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} \\
\neg \exists y \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} , \neg A(x_0) \vdash \\
\neg \exists y \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} , \exists x \neg A(x) \vdash \\
\neg \exists y \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} , \ !w \exists x \neg A(x) \vdash \\
\neg \exists y \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} , \ !w A(y_0) , \ !w \exists x . \neg A(x) \vdash \\
\neg \exists y . \{ \ !w A(y) \otimes \ !w \exists x . \neg A(x) \} \vdash \neg \{ \ !w A(y_0) \otimes \ !w \exists x \neg A(x) \} \\
\neg \exists y . \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} \vdash \exists y . \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} \\
\neg \exists y . \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} , \neg \exists y . \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} \vdash \\
! \neg \exists y . \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} , \neg \exists y . \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} \vdash \\
! \neg \exists y . \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} \vdash \neg ! \neg \exists y . \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} \\
\neg \neg \exists y . \{ \ !w A(y) \otimes \ !w \exists x \neg A(x) \} \vdash
\end{array}
\]
A proof of the drinker formula in tensorial logic

\[
\begin{align*}
\frac{A(x_0)}{\neg A(x_0), A(x_0)} & \quad \text{Axiom} \quad \neg\text{Negation} \\
\frac{\neg A(x_0), A(x_0)}{\neg A(x_0), \neg A(x_0)} & \quad \neg\text{Negation} \\
\frac{\neg A(x_0), \neg A(x_0)}{\neg A(x_0), \neg A(x_0)} & \quad \neg\text{Negation}_2 \\
\frac{\neg A(x_0)}{\neg A(x_0), \neg A(x_0)} & \quad \text{Right } \exists \\
\frac{\neg A(x_0)}{\neg A(x_0), \neg A(x_0)} & \quad \text{Negation}_1 \\
\frac{\neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}, \neg A(x_0)}{\neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}, \neg A(x_0)} & \quad \text{Left } \exists \\
\frac{\neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}, \neg A(x_0)}{\neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}, \neg A(x_0)} & \quad \text{Negation}_2 \\
\frac{\neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}, \neg A(x_0)}{\neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}, \neg A(x_0)} & \quad \text{Negation}_1 \\
\frac{\neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}, \neg A(x_0)}{\neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}, \neg A(x_0)} & \quad \text{Contraction} \\
\frac{! \neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}}{! \neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}} & \quad \neg\text{Negation}_0 \\
\frac{! \neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}}{! \neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}} & \quad \neg\text{Negation}_0 \\
\frac{! \neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}}{! \neg \exists y \{ !w A(y) \otimes !w \exists x \neg A(x) \}} & \quad \neg\text{Negation}_0
\end{align*}
\]