An Asynchronous Game Semantics for Linear Logic

Samuel Mimram

CIE’10

July 2nd 2010
A program is a text in a programming language which will evolve during time.

We have to give a meaning to this language!
A **model** interprets

- a type $A$ as a *computation space* $[A]$
- a program $f : A \Rightarrow B$ as a *transformation* $[f] : [A] \rightarrow [B]$
A denotational model interprets

- a type $A$ as a *computation space* $[A]$
- a program $f : A \Rightarrow B$ as a *transformation* $[f] : [A] \rightarrow [B]$
- in a way such that the interpretation of programs is invariant under reduction

\[ \text{denotational semantics} = \text{program invariants} \]
Interactive semantics

Here, a program will be modeled by its interactive behavior i.e. by the way it reacts to information provided by its environment.

\[
(fun \ x \ -> \ not \ x)false \ \Rightarrow \ true
\]
\[
(fun \ x \ -> \ not \ x)true \ \Rightarrow \ false
\]

⇒ Game Semantics!
How can we extend game semantics to concurrent languages?
Game semantics

An *interactive trace semantics*:

- types are interpreted by **games**

- programs are interpreted by **strategies**
An *interactive trace semantics*:

- types are interpreted by **games**
  - a poset \((M, \leq)\) of *moves*
  - a *polarization function* \(\lambda : M \to \{O, P\}\)

- programs are interpreted by **strategies**
Game semantics

An *interactive trace semantics*:

- types are interpreted by **games**
  - a poset \((M, \leq)\) of moves
  - a *polarization function* \(\lambda : M \rightarrow \{O, P\}\)

- a **play** is a sequence \(m_1 \cdot m_2 \cdots m_k\) of moves which is
  - respecting order: all the moves below a given move \(m_i\) occur before \(m_i\)
  - alternating: \(m_1 \cdot m_2 \cdot m_3 \cdot m_4 \cdots\)

- programs are interpreted by **strategies**
Game semantics

An *interactive trace semantics*:

- types are interpreted by **games**
  - a poset \((M, \leq)\) of **moves**
  - a *polarization function* \(\lambda : M \to \{O, P\}\)

- a **play** is a sequence \(m_1 \cdot m_2 \cdots m_k\) of moves which is
  - respecting order: all the moves below a given move \(m_i\) occur before \(m_i\)
  - alternating: \(m_1 \cdot m_2 \cdot m_3 \cdot m_4 \cdots\)

- programs are interpreted by **strategies**
  
  \[
  \text{strategy} = \text{set of plays closed under prefix}
  \]
Booleans

\[ B \]

\[ q \]

\[ T \quad F \]
Booleans
The strategy interpreting negation \( \text{not} : \mathbb{B} \Rightarrow \mathbb{B} \) is

\[
[\text{not}] = \{ q \cdot q \cdot T \cdot F, \ q \cdot q \cdot F \cdot T, \ldots \}
\]

\[ \mathbb{B} \Rightarrow \mathbb{B} \]
The strategy interpreting negation \( \text{not} : \mathbb{B} \Rightarrow \mathbb{B} \) is

\[
[\text{not}] = \{ q \cdot q \cdot T \cdot F, \; q \cdot q \cdot F \cdot T, \; \ldots \}
\]

\[
\mathbb{B} \; \Rightarrow \; \mathbb{B}
\]

\[
q
\]
The negation

The strategy interpreting negation $\text{not} : \mathbb{B} \Rightarrow \mathbb{B}$ is

$$[\text{not}] = \{ q \cdot q \cdot T \cdot F, \ q \cdot q \cdot F \cdot T, \ldots \}$$

$$\mathbb{B} \Rightarrow \mathbb{B}$$
The strategy interpreting negation $\text{not} : \mathbb{B} \rightarrow \mathbb{B}$ is

$$[\text{not}] = \{ q \cdot q \cdot T \cdot F, q \cdot q \cdot F \cdot T, \ldots \}$$
The strategy interpreting negation \( \text{not} : \mathbb{B} \Rightarrow \mathbb{B} \) is

\[
[\text{not}] = \{ q \cdot q \cdot T \cdot F, \ q \cdot q \cdot F \cdot T, \ldots \}
\]

\[
\mathbb{B} \Rightarrow \mathbb{B}
\]

\[
q
\]

\[
q
\]

\[
T
\]

\[
F
\]
The negation

The strategy interpreting negation \(\text{not} : \mathbb{B} \Rightarrow \mathbb{B}\) is

\[
[\text{not}] = \{ \ q \cdot q \cdot T \cdot F, \ q \cdot q \cdot F \cdot T, \ \ldots \ \}
\]

\[
\mathbb{B} \Rightarrow \mathbb{B}
\]

\[
q
\]

\[
q
\]

\[
F
\]

\[
T
\]
A category of games and strategies

We can thus build a category whose

- objects $A$ are *games*
- morphisms $\sigma : A \to B$ are *strategies*
A category of games and strategies

For example, the composite \([\text{not}] \circ [\text{not}] : \mathcal{B} \to \mathcal{B}\) is

\[
\begin{array}{c}
\mathcal{B} \xrightarrow{[\text{not}]} \mathcal{B} \\
q \quad q \\
q \quad q \\
q \quad q \\
F \quad T \\
T \quad T \\
T \quad F
\end{array}
\]
A category of games and strategies

For example, the composite \([\text{not}] \circ [\text{not}] : \mathbb{B} \to \mathbb{B}\) is
A category of games and strategies

For example, the composite \([\text{not}] \circ [\text{not}] : \mathbb{B} \to \mathbb{B}\) is

\[
\begin{array}{ccc}
\mathbb{B} & \xrightarrow{[\text{not}] \circ [\text{not}]} & \mathbb{B} \\
q & \downarrow & q \\
T & \downarrow & T
\end{array}
\]
A category of games and strategies

For example, the composite $\lnot \circ \lnot : \mathbb{B} \to \mathbb{B}$ is

$$\begin{array}{cccc}
\mathbb{B} & \xrightarrow{\lnot \circ \lnot} & \mathbb{B} \\
q & \quad & q \\
T & \quad & T
\end{array}$$

$$[\lnot] \circ [\lnot] = \{ q \cdot q \cdot T \cdot T, q \cdot q \cdot F \cdot F, \ldots \} = [\text{id}_\mathbb{B}]$$
Definable strategies

We have to characterize *definable* strategies
(= strategies which are the interpretation of a program)
Definable strategies

We have to characterize **definable** strategies
(= strategies which are the interpretation of a program)
We have to characterize **definable** strategies
(= strategies which are the interpretation of a program)
Definable strategies

We have to characterize **definable** strategies
(= strategies which are the interpretation of a program)

Two series of work laid the foundations of game semantics:

- fully abstract models of PCF [HON,AJM]
  definable strategies: bracketing and innocence conditions
  extended later on: references, control, non-determinism, ...

- fully complete models of MLL [AJ,HO]
Purposes of game semantics

- Better understanding the core features of programming languages and logics
Purposes of game semantics

- Better understanding the core features of programming languages and logics
- Compositional model checking
Purposes of game semantics

• Better understanding the core features of programming languages and logics
• Compositional model checking
• Synthesis of electronic circuits

\[
\begin{array}{c}
q \\
\text{not} \\
T \\
F
\end{array} \quad \begin{array}{c}
\rightarrow \\
\rightarrow \\
\rightarrow \\
\rightarrow
\end{array} \\
\begin{array}{c}
q \\
T \\
F
\end{array}
\]
How do we extend those results to *concurrent programming languages*?
Three flavors of conjunction

\[ B \times B \Rightarrow B \]

- \( q \)
- \( q_L \)
- \( T_L \)
- \( q_R \)
- \( F_R \)
- \( F \)
Three flavors of conjunction

$$B \times B \Rightarrow B$$

right conjunction

$q_R$

$q_L$

$q$

$F_R$

$T_L$

$F$
Three flavors of conjunction

\[ B \times B \Rightarrow B \]

- Parallel conjunction
  - \( q_L \)
  - \( q_R \)
  - \( q \)

- \( F_R \)
  - \( T_L \)
  - \( F \)
Three flavors of conjunction

\[ B \times B \Rightarrow B \]

parallel conjunction

\[ q_L \]

\[ q_R \]

\[ T_L \]

\[ F_R \]

\[ F \]
Towards asynchronous game semantics

In order to represent such strategies we have to

- take in account **non-alternating plays**
Towards asynchronous game semantics

In order to represent such strategies we have to

- take in account **non-alternating plays**
- represent concurrency by interleavings modulo an equivalence relation, in the spirit of Mazurkiewicz traces:

  asynchronous game semantics
Towards asynchronous game semantics

In order to represent such strategies we have to

- take in account **non-alternating plays**
- represent concurrency by interleavings modulo an equivalence relation, in the spirit of Mazurkiewicz traces:

  asynchronous game semantics

- more generally try to bring closer game semantics and **concurrency theory**
The multiplicative-additive linear logic

We consider here MALL formulas (without units):

\[ \Gamma, A, B \vdash \Gamma, A \& B \]

\[ \Gamma, A \& B \vdash \Gamma, A, B \]

\[ \Gamma_1, A \vdash \Gamma_2, B \]

\[ \Gamma_1, \Gamma_2, A \otimes B \vdash \Gamma_1, \Gamma_2, A \otimes B \]

\[ \Gamma, A \vdash \Gamma, A \oplus B \]

\[ \Gamma, A \oplus B \vdash \Gamma, A \oplus B \]

\[ \Gamma, B \vdash \Gamma, B \]

\[ \Gamma, A \oplus B \vdash \Gamma, A \oplus B \]

\[ \Gamma, B \vdash \Gamma, B \]

\[ \Gamma, A \oplus B \vdash \Gamma, A \oplus B \]

\[ \Gamma, A \oplus B \vdash \Gamma, A \oplus B \]

\[ \Gamma, B \vdash \Gamma, B \]

\[ \Gamma, A \oplus B \vdash \Gamma, A \oplus B \]

\[ \Gamma, B \vdash \Gamma, B \]

\[ \Gamma, A \oplus B \vdash \Gamma, A \oplus B \]
The multiplicative-additive linear logic

We consider here MALL formulas (without units):

- \( \Gamma, A, B \vdash \Gamma, A \& B \) (\&)
- \( \Gamma, A \vdash \Gamma, A \& B \)
- \( \Gamma, A \vdash \Gamma, A \oplus B \) (\oplus_L)
- \( \Gamma, B \vdash \Gamma, A \oplus B \) (\oplus_R)
- \( \Gamma, A \oplus B \vdash \Gamma, A \) (\oplus)
- \( \Gamma, A \oplus B \vdash \Gamma, B \)

- multiplicatives: concurrency / additives: non-determinism
- negative: Opponent / positive: Player
Unifying semantics of linear logic

Sequential games
Hyland, Ong 1994
Abramsky, Jagadeesan, Malacaria 1994

Asynchronous games
Melliès 2004

Concurrent games
Abramsky, Melliès 1999

Event structures
Curien, Faggian 2005
Varraca, Yoshida 2006

Coherence spaces
Relational model
Girard 1987
Proofs explore formulas

\[(A \& B) \& (C \& D)\]
Proofs explore formulas

\[(A \& B) \& (C \& D)\]
Proofs explore formulas

\((A \& B) \& (C \& D)\)
Proofs explore formulas

\[ \vdash A, B, C, D \]
\[ \vdash A, B, C \not\vdash D \]
\[ \vdash A \not\vdash B, C \not\vdash D \]
\[ \vdash (A \not\vdash B) \not\vdash (C \not\vdash D) \]
Proofs explore formulas

\[
\begin{align*}
\vdash A, B, C, D \\
\vdash A, B, C \vdash D \Rightarrow (\forall) \\
\vdash A \vdash B, C \vdash D \Rightarrow (\forall) \\
\vdash (A \vdash B) \vdash (C \vdash D) \Rightarrow (\forall)
\end{align*}
\]
Proofs explore formulas

\[
\begin{align*}
\vdash A, B, C, D \\
\vdash A, B, C \land D \\
\vdash A \land B, C \land D \\
\vdash (A \land B) \land (C \land D)
\end{align*}
\]
Proofs explore formulas

\[
\vdash A, B, C, D \\
\vdash A, B, C \not\vdash D \quad (\not\vdash) \\
\vdash A \not\vdash B, C \not\vdash D \quad (\not\vdash) \\
\vdash (A \not\vdash B) \not\vdash (C \not\vdash D) \quad (\not\vdash)
\]
Proofs explore formulas

\[
\begin{align*}
\text{play} & \quad = \quad \text{exploration of the formula} \\
\text{proof} & \quad = \quad \text{exploration strategy}
\end{align*}
\]
1. Associating an asynchronous game semantics to linear logic
2. Characterizing definable strategies in this semantics
3. Recovering preexisting models
From plays to Mazurkiewicz traces

Partial order vs transition graph

(position = downward-closed set of moves)
From plays to Mazurkiewicz traces

partial order vs transition graph

position = downward-closed set of moves
Asynchronous graphs: homotopy

\[ \begin{array}{c}
\text{plays} \\
\begin{array}{cc}
\hspace{1em} m & n \\
\downarrow & \downarrow \\
\sim & \\
\downarrow & \downarrow \\
n & m
\end{array}
\end{array} \sim
\begin{array}{c}
\text{vs} \\
\begin{array}{cc}
\hspace{1em} m & n \\
\downarrow & \downarrow \\
\downarrow & \downarrow \\
n & m
\end{array}
\end{array} \]
Asynchronous graphs: homotopy

plays

processes

\[ m \parallel n \quad \text{vs} \quad m \cdot n + n \cdot m \]
Asynchronous graphs: homotopy

plays

\[
\begin{array}{c}
  \quad \sim \quad \\
  n \quad \sim \quad m \\
  m \quad \sim \quad n \\
\end{array}
\]

 Processes

\[
m \parallel n
\]

Linear logic

Multiplicatives

Additives

\[
m \cdot n + n \cdot m
\]
Asynchronous graphs: homotopy

plays

\[ m \sim n \]

\[ n \sim m \]

vs

\[ m \leftrightarrow n \]

\[ n \leftrightarrow m \]

processes

\[ m \parallel n \]

\[ m \cdot n + n \cdot m \]

linear logic

multiplicatives

additives

gometry

possible deformation

hole
Asynchronous games

**Definition**
An asynchronous game is an asynchronous graph together with an initial position.

**Definition**
A play is a path in a game starting from the initial position.

**Definition**
A strategy $\sigma : A$ is a prefix closed set of plays on the asynchronous graph $A$. 
Asynchronous game semantics: conjunction

The game $\mathbb{B} \times \mathbb{B} \Rightarrow \mathbb{B}$ contains eight subgraphs:
Asynchronous game semantics: conjunction

Left implementation of conjunction:

```
* × * ⇒ *
q
↓
* × * ⇒ q

q × * ⇒ q

q × * ⇒ q

T × * ⇒ q

T × q ⇒ q

q × q ⇒ q

q × F ⇒ q

T × F ⇒ q
```
Asynchronous game semantics: conjunction

Right implementation of conjunction:
Asynchronous game semantics: conjunction

Parallel implementation of conjunction:

\[
\begin{align*}
* \times * & \Rightarrow * \\
\downarrow & \\
* \times * & \Rightarrow q \\
\downarrow & \\
q \times q & \Rightarrow q \\
\sim & \\
q \times * & \Rightarrow q \\
\downarrow & \\
q \times q & \Rightarrow q \\
\sim & \\
T \times * & \Rightarrow q \\
\sim & \\
T \times q & \Rightarrow q \\
\sim & \\
T \times F & \Rightarrow q \\
\sim & \\
q \times F & \Rightarrow q \\
\sim & \\
T \times q & \Rightarrow q \\
\sim & \\
T \times * & \Rightarrow q \\
\sim & \\
T \times F & \Rightarrow q \\
\sim & \\
q \times q & \Rightarrow q \\
\sim & \\
q \times F & \Rightarrow q \\
\sim & \\
T \times F & \Rightarrow q
\end{align*}
\]
Interpreting formulas and proofs

By an easy inductive definition we associate

- an asynchronous game to every formula

\[
\begin{align*}
&\vdash A, B, C, D
\end{align*}
\]

\[
(\vdash A \triangleright B) \triangleright (C \triangleright D)
\]
Interpreting formulas and proofs

By an easy inductive definition we associate

- an asynchronous game to every formula
- a strategy to every proof

\[ \vdash A, B, C, D \]
\[ \vdash A, B, C \not\!\not\!\not\!\!\not\vdash D \]
\[ \vdash A \not\!\!\not\!\not\!\!\not\vdash B, C \not\!\!\not\!\not\!\!\not\vdash D \]
\[ \vdash (A \not\!\!\not\!\not\!\!\not\vdash B) \not\!\!\not\!\not\!\!\not\vdash (C \not\!\!\not\!\not\!\!\not\vdash D) \]
In order to characterize definable strategies, we will impose conditions on our strategies.
In order to characterize definable strategies, we will impose conditions on our strategies.

We will begin by some technical conditions which are necessary to regulate the strategies...
From sequentiality to causality

A game induces an asynchronous graph:

```
(\emptyset) ← a → \{a\} ← b → \{b\}
\sim
\{a\} ← b → \{a, b\} ← \{b\}
\sim
\{a, b\} ← c → \{a, b, c\} ← \{a, b\}
\sim
\{a, b, c\} ← d → \{a, b, c, d\} ← \{a, b, c, d\}
```

```
\{a\} ← a → \emptyset ← b → \{b\}
\sim
\{a\} ← b → \{a, b\} ← \{b\}
\sim
\{a, b\} ← c → \{a, b, c\} ← \{a, b\}
\sim
\{a, b, c\} ← d → \{a, b, c, d\} ← \{a, b, c, d\}
```

```
\{a\} ← a → \emptyset ← b → \{b\}
\sim
\{a\} ← b → \{a, b\} ← \{b\}
\sim
\{a, b\} ← c → \{a, b, c\} ← \{a, b\}
\sim
\{a, b, c\} ← d → \{a, b, c, d\} ← \{a, b, c, d\}
```
From sequentiality to causality

Conversely, one needs the Cube Property
The Cube Property

Theorem

Homotopy classes of paths are generated by a partial order on moves.

Proof: essentially Birkhoff duality theorem for finite posets.
Definition
An **asynchronous game** is a pointed asynchronous graph satisfying the Cube Property.

Definition
A **strategy** $\sigma : A$ is a prefix closed set of plays on the asynchronous graph $A$. 
Positional strategies

Definition
A strategy $\sigma$ is **positional** when its plays form a subgraph of the game:

$$
\sigma \ni \begin{array}{c}
\ast \\
s \\
u \\
y
\end{array}
\quad \text{and} \quad
\begin{array}{c}
\ast \\
s \sim t \\
t \\
x
\end{array}
\quad \text{and} \quad
x \in \sigma 
\implies 
\begin{array}{c}
\ast \\
t \\
u \\
y
\end{array}
\ni 
x \in \sigma
$$
We consider strategies which

1. are *position*al,
Ingenuous strategies

We consider strategies which

1. are *positional*,
2. satisfy the **Cube Property**,
Ingenuous strategies

We consider strategies which

1. are **positional**,
2. satisfy the **Cube Property**,
3. satisfy

\[
\begin{align*}
\sigma \ni m & \quad n \in \sigma \\
\sigma \ni m & \quad n \in \sigma \\
\sigma \ni n & \quad m \in \sigma \\
\sigma \ni n & \quad m \in \sigma
\end{align*}
\]

implies

\[
\begin{align*}
\sigma \ni n & \quad m \in \sigma \\
\sigma \ni n & \quad m \in \sigma
\end{align*}
\]

where \( m \) is a Proponent move.
Ingenuous strategies

We consider strategies which

1. are **positional**, 
2. satisfy the **Cube Property**, 
3. satisfy . . . 
4. are **deterministic**:

\[
\begin{array}{ccc}
\sigma \ni m & \Rightarrow & y_1 \\
\ni n & \Rightarrow & y_2 \\
\end{array}
\]

implies

\[
\begin{array}{ccc}
\sigma \ni m & \Rightarrow & y_1 \\
\sim & \Rightarrow & y_2 \\
\ni n & \Rightarrow & m \in \sigma \\
\end{array}
\]

where \( m \) is a Proponent move.
Property

Asynchronous games and strategies form a $\ast$-autonomous category (which is compact closed).
This category still has “too many” strategies!

\[ A \otimes B = A \supset B \]
Halting positions

In the spirit of the relational model, a strategy $\sigma$ should be characterized by its set $\sigma^\circ$ of halting positions.

Definition
A **halting position** of a strategy $\sigma$ is a position $x$ such that there is no Player move $m : x \rightarrow y$ that $\sigma$ can play.
The game \( \mathbb{B} \otimes \mathbb{B} \) contains the subgraph:
The pair $\text{true} \otimes \text{false}$:
The left biased pair true $\otimes$ false:
Courteous strategies

Definition
An ingenuous strategy $\sigma$ is **courteous** when it satisfies

where $m$ is a Player move.

Theorem
A courteous ingenuous strategy $\sigma$ is characterized by its set $\sigma^\circ$ of halting positions.
Concurrent strategies

The halting positions of such a strategy $\sigma : A$ are precisely the fixpoints of a closure operator on the positions of $A$.

- We thus recover the model of concurrent strategies.
- A semantical counterpart of the focusing property: strategies can play all their Player moves in one “cluster” of moves.
Some introduction rules can be permuted:

\[
\begin{align*}
\vdash A & \quad \vdash B, C, D \quad (\supset) \\
\vdash A \quad \vdash B, C \supset D & \quad (\supset) \\
\vdash A \otimes B, C \supset D & \quad (\supset) \\
\vdash (A \otimes B) \supset (C \supset D) & \quad (\supset)
\end{align*}
\]
Focusing

Some introduction rules can be *permuted*:

\[ \vdash A \vdash B, C, D \vdash B, C, D \quad (\text{\^\$}) \]
\[ \vdash A \vdash B, C, D \vdash A \otimes B, C, D \quad (\otimes) \]
\[ \vdash A \vdash B, C, D \vdash (A \otimes B) \otimes (C \otimes D) \quad (\otimes) \]
\[ \vdash A \vdash B, C, D \vdash (A \otimes B) \otimes (C \otimes D) \quad (\text{\^\$}) \]

Every proof can be reorganized into a *focusing* proof:

- **negative phase**: if the sequent contains a negative formula then a negative formula should be decomposed,
- **positive phase**: otherwise a positive formula should be chosen and decomposed repeatedly until a (necessarily unique) formula is produced
Towards a functorial correspondence

The operation \((-)^\circ\) from the category of games and courteous ingenuous strategies to the category of relations is not functorial!
This mismatch is essentially due to **deadlock** situations occurring during the interaction.
The scheduling criterion

the left conjunction:

\[
\begin{array}{c}
\mathbb{B} \otimes \mathbb{B} \rightarrow \mathbb{B} \\
q \\
\end{array}
\]

\[
\begin{array}{c}
q \\
T \\
\end{array}
\]

\[
\begin{array}{c}
q \\
F \\
\end{array}
\]

\[
\begin{array}{c}
F \\
\end{array}
\]
The scheduling criterion

The right boolean composed with the left conjunction:

\[ B \otimes B \rightarrow B \]

\[
\begin{array}{ccc}
q & q & q \\
F & T & F \\
T & F & F \\
\end{array}
\]
The scheduling criterion

Two kinds of tensors: $\otimes$ and $\otimes$. 
The scheduling criterion

Two kinds of tensors: $\otimes$ and $\otimes$. 

\[
\begin{array}{ccc}
B & \otimes & B \\
& q & \\
& F & \\
& q & \\
& T & \\
\end{array}
\]
The scheduling criterion

Two kinds of tensors: $\otimes$ and $\oslash$.

\[
\begin{array}{c}
B \otimes B \\
q \\
F \\
q \\
T
\end{array}
\]
The scheduling criterion

Two kinds of tensors: $\otimes$ and $\bowtie$.

\begin{center}
\begin{tikzpicture}
    \node (A) at (0,0) {B \otimes B};
    \node (B) at (1,1) {q};
    \node (C) at (1,0) {T};
    \node (D) at (0,1) {F};
\end{tikzpicture}
\end{center}
The scheduling criterion

Two kinds of tensors: $\otimes$ and $\bowtie$.

The role of the correctness criterion is to avoid deadlocks!
Functoriality

Theorem

Strategies which are
- ingenuous
- courteous
- and satisfy the scheduling criterion

compose and satisfy

\[(\sigma; \tau)^\circ = \sigma^\circ; \tau^\circ\]
Functoriality

Theorem

Strategies which are

- ingenuous
- courteous
- and satisfy the scheduling criterion

compose and satisfy

\[(\sigma; \tau)^\circ = \sigma^\circ; \tau^\circ\]

Theorem

The model we thus get is fully complete for MLL+MIX.
Conclusion

We have:

• a game semantics adapted to concurrency
• an unifying framework in which we recover
  • innocent strategies
  • game semantics
  • concurrent games
  • the relational model
  • event structure semantics

In the future:

• extend this model (exponentials in particular)
• typing of concurrent processes (CCS without deadlocks)
• links with geometrical models for concurrency