Determinisme, Structures d’événements et le $\pi$-Calcul

Daniele Varacca

with Nobuko Yoshida

PPS

Sophia, Parsec - 2/2/2007
Determinism

What is determinism

- For functions: only one result
- For reactive systems: confluence

Only one maximal execution, up to order
Determinism

What is determinism

- For functions: only one result
- For reactive systems: confluence

Only one maximal execution, up to order
Some fairness assumptions may be necessary
What is probabilistic determinism

- For functions: only one probability distribution
- For reactive systems?
  Only one maximal execution, up to order??
Road Map

1. Typed $\pi$
   - Syntax

2. Event Structures
   - Conflict Freeness
   - Semantics
   - Correspondence

3. Probabilistic case
   - Syntax
   - Probabilistic event structures
Road Map

1. Typed π
   - Syntax

2. Event Structures
   - Conflict Freeness
   - Semantics
   - Correspondence

3. Probabilistic case
   - Syntax
   - Probabilistic event structures
We all know what the $\pi$-calculus is

$$x(\tilde{y}).P \mid \bar{x}\langle\tilde{z}\rangle.Q \longrightarrow P\{\tilde{z}/\tilde{y}\} \mid Q$$
We all know what the $\pi$-calculus is

$$x(\tilde{y}).P \mid \overline{x}\langle \tilde{z} \rangle .Q \rightarrow P\{\tilde{z}/\tilde{y}\} \mid Q$$

We consider a restricted version:
bound output only (“internal” mobility)
We all know what the $\pi$-calculus is

$$x(\tilde{y}).P \mid \overline{x}(\tilde{y}).Q \rightarrow (\nu \tilde{y})(P \mid Q)$$

We consider a restricted version:
bound output only (“internal” mobility)
The syntax

\( \pi \) processes

\[
P ::= x \&_{i \in I} \inp_i(\tilde{y}_i).P_i \quad \text{branching}
\]
\[
| \overline{x}\inp_j(\tilde{y}).P \quad \text{selection}
\]
\[
| !x(\tilde{y}).P \quad \text{server}
\]
\[
| \overline{x}(\tilde{y}).P \quad \text{client}
\]
\[
| P | Q \quad \text{parallel}
\]
\[
| (\nu x)P \quad \text{restriction}
\]
\[
| 0 \quad \text{inaction}
\]
Typed $\pi$-calculus

A linear type discipline:

(A) for each linear name there are a unique input and a unique output

(B) for each replicated name there is a unique stateless replicated input with zero or more dual outputs

This discipline guarantees confluence (determinism)
Examples

\overline{a}.b \mid \overline{a}.c \mid a

This is not typable as \( a \) appears twice as output
Examples

\[ b.\bar{a} \mid c.\bar{b} \mid a.(\bar{c} \mid \bar{e}) \]

This is typable since each channel appears at most once as input and output
Examples

\[ \mathbf{!} \ b \overline{a} \mid \mathbf{!} \ b \overline{c} \]

This is **not** typable as there are two different servers associated with \( b \).
Examples

\[ !b.\overline{a} | \overline{b} | !c.\overline{b} \]

This is typable: the two clients on \( b \) are associated to a unique server
$P = \overline{a} \mathit{in}_1.b \mid a[\mathit{in}_1 \overline{d} \& \mathit{in}_2 \overline{e}]$

This process is typable, and performs a choice:

$P \rightarrow (b \mid \overline{d})$
Road Map

1. Typed $\pi$
   - Syntax

2. Event Structures
   - Conflict Freeness
   - Semantics
   - Correspondence

3. Probabilistic case
   - Syntax
   - Probabilistic event structures
True concurrency

Standard “interleaving” semantics
  - reduces parallelism to nondeterministic interleaving (“expansion law”)
  - Labelled transition systems, reduction semantics
True concurrency

Standard “interleaving” semantics
- reduces parallelism to nondeterministic interleaving (“expansion law”)
- Labelled transition systems, reduction semantics

“True concurrent” models
- Represent explicitly causality, conflict, independence
- Petri nets, Mazurkiewicz traces, event structures
Event structures

An event structure is a partial order \( \langle E, \leq \rangle \) together with a conflict relation \( \sim \)

- order represents causal dependency
- conflict is irreflexive and symmetric
- conflict is “hereditary”:

\[
e_1 \sim e \text{ and } e_1 \leq e_2 \text{ implies } e_2 \sim e
\]

A conflict is immediate if it is not inherited from another conflict
A notion of run

A **configuration** is a set $x$ of events

- justified: $e \in x$, $e' \leq e \implies e' \in x$
- conflict-free: $e, e' \in x \implies \neg e \vdash e'$

Example:

$$[e] := \{e' \mid e' \leq e\}$$
Event structures

Example

\[ \begin{array}{ccc}
  d & \sim & e \\
  b & \sim & c \\
  a & & \\
\end{array} \]
Event structures

Example

Events can also be labelled: $\lambda : E \rightarrow L$
Event structures

Example

Events can also be labelled: $\lambda : E \rightarrow L$
Operators on event structures

Prefixing $\alpha.\varepsilon$

\begin{align*}
\gamma_1 & \quad \gamma_2 \\
\big| & \quad \big| \\
\beta_1 \sim \cdots \sim & \quad \beta_2
\end{align*}
Operators on event structures

Prefixing $\alpha.\mathcal{E}$
Operators on event structures

Prefix sum $\sum_{i \in I} \alpha_i.\mathcal{E}_i$

\[
\begin{array}{c}
\gamma_1 \\
\beta_1 \\
\gamma_2 \\
\beta_2
\end{array}
\]
Operators on event structures

Prefix sum $\sum_{i \in I} \alpha_i.\mathcal{E}_i$

$\gamma_1 \rightarrow \beta_1 \rightarrow \alpha_1 \sim \alpha_2 \sim \beta_2 \rightarrow \gamma_2$
Operators on event structures

Parallel composition $\mathcal{E}_1 \parallel \mathcal{E}_2$

\[
\begin{array}{c}
\gamma_1 \\
\gamma_2 \\
\beta \\
\bar{\beta}
\end{array}
\]
Operators on event structures

Parallel composition $\mathcal{E}_1 \parallel \mathcal{E}_2$

A complex construction involving synchronisation
Consider

- $E = \langle E, \leq, \prec, \lambda \rangle$, a labelled event structure
- $e$, one of its minimal events

We define $E \setminus e$ as $E$ minus event $e$, and minus all events that are in conflict with $e$

We can then generate a labelled transition system as follows: if $\lambda(e) = \beta$, then

$$E \xrightarrow{\beta} E \setminus e$$
Event structures and transition systems

Example

\[
\begin{array}{c}
\gamma_1 \\
\beta_1 \\
\end{array}
\quad
\begin{array}{c}
\gamma_2 \sim \gamma_3 \\
\beta_2 \\
\end{array}
\]

An event structure \( \mathcal{E} \)
Event structures and transition systems

Example

\[ \gamma_1 \quad \gamma_2 \sim \sim \sim \gamma_3 \]
\[ \beta_1 \sim \sim \sim \beta_2 \]

Eliminate a minimal event \( e \) (labelled by \( \beta_2 \))
Event structures and transition systems

Example

\[ \gamma_1 \quad \gamma_2 \quad \gamma_3 \]

\[ \beta_1 \quad \gamma_2 \sim \gamma_3 \]

 Eliminate a minimal event \( e \) (labelled by \( \beta_2 \))
Event structures and transition systems

Example

And every event in conflict with it
Event structures and transition systems

Example

\[ \gamma_2 \sim \sim \sim \sim \gamma_3 \]

And every event in conflict with it
Event structures and transition systems

Example

\[ \gamma_1 \xrightarrow{\beta_1} \gamma_2 \xrightarrow{\beta_2} \gamma_3 \]

\[ \mathcal{E} \xrightarrow{\beta_2} \mathcal{E}|_e \]

\[ \gamma_2 \xrightarrow{\sim} \gamma_3 \]
Conflict freeness

When the conflict relation is empty, the corresponding transition system is confluent
Conflict freeness

When the conflict relation is empty, the corresponding transition system is confluent.

Idea: give a conflict free event structure semantics to the linear $\pi$-calculus.
Conflict freeness

When the conflict relation is empty, the corresponding transition system is confluent

Idea: give a conflict free event structure semantics to the linear $\pi$-calculus

Issues:
- perform synchronisation without introducing conflict
- difficult to handle name generation
- hidden conflicts appear
Example:

- Stateless replicated resource: post office $!a.P$
- Clients: customers $\bar{a}.C$

Every customer wants to send a letter $a$
The process $\overline{a}.D | \overline{a}.N | !a.P$ is confluent

\[
\text{The post office}
\]

\[
\begin{align*}
D & | P | \overline{a}.N | !a.P \\
N & | P | \overline{a}.D | !a.P \\
N & | P | D | P | !a.P \\
\end{align*}
\]
The post office

Situation 1: two customers, one till
A conflict to resolve: who goes first?
Eventually, it does not matter, but the two events are not independent
The post office

Situation 1: two customers, one till
A conflict to resolve: who goes first?
Eventually, it does not matter, but the two events are not independent

Situation 2: two customers, infinitely many identical tills
if the two customers want to go to the same till, there is a conflict
The post office

Situation 1: two customers, one till
A conflict to resolve: who goes first?
Eventually, it does not matter, but the two events are not independent

Situation 2: two customers, infinitely many identical tills
if the two customers want to go to the same till, there is a conflict

Situation 3: one customer, infinitely many identical tills
the customer has to choose which till to go to
### The post office

Solution: no conflict arises if every possible customer is assigned a specific till *in advance*
Event structure semantics of $\pi$

The semantics has the form $[P]^\Delta$, where $\Delta$ assigns each client a specific instance of its server.
Event structure semantics of $\pi$

The semantics has the form $[P]^\Delta$, where $\Delta$ assigns each client a specific instance of its server.

The interpretation functions are partial functions: for the wrong choice of $\Delta$, a post office customer would not find her till.

It is always possible to find suitable $\Delta$: we perform $\alpha$-conversion “at compile time”.
The semantics has the form $\llbracket P \rrbracket^\Delta$, where $\Delta$ assigns each client a specific instance of its server.

The interpretation functions are partial functions: for the wrong choice of $\Delta$, a post office customer would not find her till.

It is always possible to find suitable $\Delta$: we perform $\alpha$-conversion “at compile time.”

**Theorem:**
For every process $P$, there exists a choice $\Delta$ such that $\llbracket P \rrbracket^\Delta$ is defined.
Correspondence between transition system and event structure:

Theorem: [Operational correspondence]

If $P \xrightarrow{\beta} P'$, then $\llbracket P \rrbracket_\Delta \xrightarrow{\beta} \simeq \llbracket P' \rrbracket_\Delta'$
Correspondence between transition system and event structure:

**Theorem: [Operational correspondence]**

If \( P \xrightarrow{\beta} P' \), then \( [P]^\Delta \xrightarrow{\beta} \cong [P']^\Delta' \)

If \( [P]^\Delta \xrightarrow{\beta} \mathcal{E}' \), then there exists \( P' \) such that \( P \xrightarrow{\beta} P' \) and \( [P']^\Delta' \cong \mathcal{E}' \)
Road Map

1. Typed $\pi$
   - Syntax

2. Event Structures
   - Conflict Freeness
   - Semantics
   - Correspondence

3. Probabilistic case
   - Syntax
   - Probabilistic event structures
The syntax

\[ P ::= \begin{align*} & x \land_{i \in I} \text{in}_i(\tilde{y}_i).P_i \\
| & \overline{x}\text{in}_j(\tilde{y}).P \\
| & !x(\tilde{y}).P \\
| & \overline{x}(\tilde{y}).P \\
| & P \parallel Q \\
| & (\nu x)P \\
| & 0 \end{align*} \]

branching

selection

server

client

parallel

restriction

inaction
### The syntax

**π** processes

\[
P ::= \begin{align*}
x \&\! \sum_{i \in I} \text{in}_i(\tilde{y}_i).P_i & \quad \text{branching} \\
\overline{x} \bigoplus_{i \in I} p_i \text{in}_i(\tilde{y}_i).P_i & \quad \text{selection} \\
!x(\tilde{y}).P & \quad \text{server} \\
\overline{x}(\tilde{y}).P & \quad \text{client} \\
| & \quad \text{parallel} \\
| & P \mid Q \\
| & (\nu x)P \\
| & 0 \quad \text{inaction}
\end{align*}
\]
Typed $\pi$-calculus

The same linear type discipline:

(A) for each linear name there are a unique input and a unique output

(B) for each replicated name there is a unique stateless replicated input with zero or more dual outputs

This discipline guarantees probabilistic confluence?
Example

\[ P = \overline{a}[in_1.b \oplus_p in_2.c] | a[in_1.d \& in_2.e] \]

This process is typable, and performs a choice:

\[ P \rightarrow_p (b \mid \overline{d}) \]

\[ P \rightarrow_{1-p} (c \mid \overline{e}) \]
How to add probabilities to event structures?
Idea: resolve the immediate conflict by flipping a coin

Coins resolve local choices
What does \textit{local} mean?
Locality: Example

- take car
- take train
- wife comes home

Non local!
Locality: Example

\begin{itemize}
  \item \textbf{take car} \quad \cdots \quad \textbf{take train}
  \item \textit{wife comes home}
\end{itemize}

Local!
An event structure is **confusion-free** when

- “reflexive” immediate conflict is an equivalence
- any two events in immediate conflict have the same predecessors

The equivalence classes are the **cells**

Cells represent local choices
Examples

Confusion Free
Examples

Confusion!
Examples

Confusion!
Valuations on event structures

A local valuation on $E$ associates to every cell a coin/die. It is a function $p : E \rightarrow [0, 1]$ such that for every cell $c$:

$$\sum_{e \in c} p(e) = 1$$

The weight $\nu_p(x)$ of a configuration $x$ is the product of the probabilities of the events in $x$. 
Probabilistic runs

An conflict free event structure has only one maximal configurations (only one maximal run up to order)

**Theorem:** [Varacca-Völzer-Winskel]

For every local valuation $p$ there exists a unique probability measure $m_p$ on the set of maximal configurations such that

$$m_p(\uparrow x) = v_p(x)$$

“A probabilistic event structure has only one maximal run up to order”

probabilistic determinism
Confusion arises from synchronisation
Consider $(\overline{a} \mid a)$
The event structure for this is

\[
\begin{array}{c}
\overline{a} \\
\tau \\
a
\end{array}
\]

Confusion - the choice is not local
Confusion arises from synchronisation
Consider \((\overline{a} \mid a)\)
The event structure for this is

\[
\overline{a} \sim \tau \sim a
\]

Confusion - the choice is not local

Issue: how to perform synchronisation without introducing confusion
Confusion arises from synchronisation

Consider \((\overline{a} \mid a)\)

The event structure for this is

\[
\overline{a} \sim \tau \sim a
\]

Confusion - the choice is not local

Issue: how to perform synchronisation without introducing confusion

Same machinery as for the conflict free case
The semantics of $\pi$ extends to the probabilistic case.

Only one probability distributions over maximal runs: probabilistic determinism.

Relations with interleaving semantics (Segala automata)
Related Work

- Concurrent games (Melliès, Faggian, Curien)
- Untyped \( \pi \)-calculus (with Silvia Crafa)
- Termination
- Encodings
Dessert
Historical perspective

An unfair and myopic view of the last 40 years
An unfair and myopic view of the last 40 years

Petri ['60]

Petri nets
Historical perspective

An unfair and myopic view of the last 40 years

Petri [’60]               Scott and Strachey [’70]

Denotational semantics - Domain theory
Historical perspective

An unfair and myopic view of the last 40 years

Petri ['60] → Scott and Strachey ['70] ↓ Nielsen, Plotkin and Winskel ['80]

Event structures
An unfair and myopic view of the last 40 years

Petri ['60] \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \righta
An unfair and myopic view of the last 40 years

- Petri ['60]
- Scott and Strachey ['70]
- Park and Milner ['80]
- Nielsen, Plotkin and Winskel ['80]
- Berry and Boudol ['90]

Reduction semantics
An unfair and myopic view of the last 40 years

- Petri ['60]
- Scott and Strachey ['70]
- Park and Milner ['80]
- Nielsen, Plotkin and Winskel ['80]
- Girard ['80]
- Berry and Boudol ['90]

Linear logic
An unfair and myopic view of the last 40 years

Petri ['60] → Scott and Strachey ['70]

Scott and Strachey ['70] ↓

Park and Milner ['80] ↓

Park and Milner ['80] ↓

Nielsen, Plotkin and Winskel ['80] ↓

Berry and Boudol ['90] ↓

Berry and Boudol ['90] ↓

Girard ['80] ↓

Girard ['80] ↓

Blass et al. ['90]

Game semantics
Historical perspective

An unfair and myopic view of the last 40 years


Linearly typed $\pi$ calculus
Historical perspective

An unfair and myopic view of the last 40 years

- Petri ['60]
- Scott and Strachey ['70]
- Park and Milner ['80]
- Nielsen, Plotkin and Winskel ['80]
- Girard ['80]
- Berry and Boudol ['90]
- Blass et al. ['90]
- Honda, Berger and Yoshida ['00]
- Faggian, Curien, Mellèses ['00]

True concurrent games
Historical perspective

An unfair and myopic view of the last 40 years

Petri ['60]

Scott and Strachey ['70]

Nielsen, Plotkin and Winskel ['80]

Scott and Strachey ['70]

Park and Milner ['80]

Berry and Boudol ['90]

Honda, Berger and Yoshida ['00]

Park and Milner ['80]

Berry and Boudol ['90]

Girard ['80]

Blass et al. ['90]

Event structures for $\pi$

Varacca and Yoshida [Now]

Varacca and Yoshida [Now]

Varacca and Yoshida [Now]

Faggian, Curien, Mellès ['00]

Faggian, Curien, Mellès ['00]