If you are in a hurry

SPOILER WARNING: Plot and/or ending details follow.

\[
\begin{align*}
\frac{}{\emptyset \vdash 0} \quad & (\text{NIL}) \\
\frac{A \vdash \pi P \quad A \leq A'}{A' \vdash \pi P} \quad & (\text{WEAK}) \\
\frac{}{A' \vdash_p P} \quad & (\text{REP}) \\
\frac{i = 1, 2 : A_i \vdash \pi P_i}{A_1 \odot A_2 \vdash \pi P_1 | P_2} \quad & (\text{PAR}) \\
\frac{A \vdash \pi P}{\forall l : \text{md}(\Sigma_A(l)) \notin \{\downarrow_1, \uparrow_1, \downarrow_{\omega_0}\}} \quad & (\text{RES}) \\
\frac{p.(\nu \tilde{x}) \left(p : ((\tilde{\sigma})^p, \rho) + \rho(\tilde{x} : \tilde{\sigma}) \odot A\right) \vdash_p p(\tilde{x}).P}{A \vdash \pi P} \quad & (\text{INP}_p) \\
\frac{(\nu \tilde{x}) \left(l : ((\tilde{\sigma})^{\downarrow_1}, \rho, \emptyset, (\tilde{x})) + l.\hat{\rho}(\tilde{x} : \tilde{\sigma}) \odot l.A\right) \vdash \pi l(\tilde{x}).P}{A \vdash \pi P} \quad & (\text{INP}_1) \\
\frac{(\nu \tilde{x}) \left(u : ((\tilde{\sigma})^{\downarrow_{\omega_0}}, \rho, \emptyset, (\tilde{x})) + u.\rho(\tilde{x} : \tilde{\sigma}) \odot u.A\right) \vdash u(\tilde{x}).P}{A \vdash \pi P} \quad & (\text{INP}_\omega) \\
\frac{\uparrow_{\omega} \not\in \text{md}(\tilde{\sigma}) \quad \forall l : \text{md}(\Sigma_A(l)) \notin \{\downarrow_1, \uparrow_1, \downarrow_{\omega_0}\}}{p.(p : ((\tilde{\sigma})^p, \rho) + \overline{\rho}(\tilde{x} : \tilde{\sigma}) \odot A) \vdash_p \overline{p}(\tilde{x}).P} \quad & (\text{OUT}_p) \\
\frac{l : ((\tilde{\sigma})^{\uparrow_1}, \rho, \emptyset, \emptyset, (\tilde{x})) + l.\check{\rho}(\tilde{x} : \tilde{\sigma}) \odot l.A \vdash \pi l(\tilde{x}).P}{A \vdash \pi P} \quad & (\text{OUT}_1) \\
\frac{u : ((\tilde{\sigma})^{\uparrow_{\omega}}, \rho) + u.\check{\rho}(\tilde{x} : \tilde{\sigma}) \odot u.A \vdash \pi u(\tilde{x}).P}{A \vdash \pi P} \quad & (\text{OUT}_\omega)
\end{align*}
\]
Deciding Deterministic Responsiveness and Closeness in $\pi$-calculus

Maxime Gamboni

Insituto Superior Técnico

June 27, 2006
Teach Yourself Polyadic $\pi$-Calculus in 4 Minutes (I)

- Model for Communication & Concurrency
- Based around Named Channels
Teach Yourself Polyadic $\pi$-Calculus in 4 Minutes (I)

- Model for Communication & Concurrency
- Based around Named Channels

Two kinds of things are done in $\pi$.

- Sending something ($\xi$) over a channel ($a$): $\overline{a}(\xi).P$
- Receiving something on a channel ($a$), and referring to it as $x$ afterwards: $a(x).P$
Some other constructs: $P_1 | P_2$, $(\nu x) P$, $! P$, $0$

E.g. $a(s) | a(x).\bar{x} \rightarrow 0 | \bar{s}$
Teach Yourself Polyadic $\pi$-Calculus in 4 Minutes (II)

- Some other constructs: $P_1|P_2$, $(\nu x) P$, $! P$, $0$

E.g. $\bar{a}(s) | a(x).\bar{x} \rightarrow 0 | \bar{s}$

- *POLY*-adic: More than one name can be moved around at a time

E.g. $\bar{a}(x, y, z).P$
Higher level languages can be encoded into $\pi$:

\[
\llbracket \bar{a}\langle \xi \rangle \rrbracket \overset{\text{def}}{=} \bar{a}\langle u \rangle . \! u(\bar{r}). \cdots \]

server for $\xi$
Higher level languages can be encoded into $\pi$:

$$\llbracket \overline{a}(\xi) \rrbracket \overset{\text{def}}{=} \overline{a}\langle u \rangle . \overline{u(\overline{r})}. \cdots$$

server for $\xi$

We want Full Abstraction:

$$(P \approx Q) \iff (\llbracket P \rrbracket \approx_r \llbracket Q \rrbracket)$$
These two (high level) processes are \textit{bisimilar}

\[
P = a(b).\text{if}(b)(\text{if}(\neg b)\text{ print } \text{OOPS}; \text{ else print } \text{OK};) \\
Q = a(b).\text{print } \text{OK};
\]
The two (high level) processes are bisimilar:

\[
P = a(b).\text{if}(b)(\text{if}(\neg b) \text{print } OOPS; \text{else print } OK;)
\]

\[
Q = a(b).\text{print } OK;
\]

Yet their encoded forms are not.

We also need to enforce:
$\approx_R$ is not a Regular Bisimulation

- These two (high level) processes are *bisimilar*
  
  $P = a(b).\text{if}(b)(\text{if}(\neg b) \text{ print } OOPS; \text{ else print } OK;)$
  
  $Q = a(b).\text{print } OK;$

- Yet their encoded forms are not.

- We also need to enforce:

  *Determinism,*
These two (high level) processes are \textit{bisimilar}

\[
P = a(b).\text{if}(b)(\text{if}(\neg b)\text{ print } OOPS; \text{ else }\text{ print } OK;)\]
\[
Q = a(b).\text{print } OK;
\]

Yet their encoded forms are not.

We also need to enforce:
\begin{itemize}
\item \textit{Determinism},
\item \textit{Closeness},
\end{itemize}
These two (high level) processes are *bisimilar*

\[ P = a(b).\text{if}(b)(\text{if}(\neg b)\text{ print } OOPS;\text{ else print } OK;) \]
\[ Q = a(b).\text{print } OK; \]

Yet their encoded forms are not.

We also need to enforce:

*Determinism, Closeness, Responsiveness*
\( \approx_R \) is not a Regular Bisimulation

- These two (high level) processes are bisimilar
  
  \[ P = a(b).\text{if}(b)(\text{if}(!b) \text{ print } OOPS; \text{ else } \text{ print } OK;) \]
  
  \[ Q = a(b).\text{print } OK; \]

- Yet their encoded forms are not.

- We also need to enforce:

  *Determinism, Closeness, Responsiveness* and *Uniformity.*
Name Classes

Names in an encoded process (and its environment) are separated in three groups.

- For encoded data:
  
  $\omega$-names
Name Classes

Names in an encoded process (and its environment) are separated in three groups.

- For encoded data:
  - $\omega$-names
- For responsiveness:
  - Linear names
Names in an encoded process (and its environment) are separated in three groups.

- For encoded data:
  \( \omega \)-names

- For responsiveness:
  linear names

- For the rest:
  plain names
Two constructs are needed for defining bisimilarity:

**Definition**

*Template Processes* $L_\sigma(a)$: Models $\omega$-servers in the environment.

$$L((\mathcal{P}^1)_{\omega}(a) = !a(x).\bar{x}\langle a_1 \rangle$$
Two constructs are needed for defining bisimilarity:

**Definition**

*Template Processes* $L_\sigma(a)$: Models $\omega$-servers in the environment.

$L_{(\downarrow(a))} = !a(x).\bar{x}\langle a_1 \rangle$

**Definition**

*Observable Data* $\Omega_\Sigma^P(a)$: Tests $\omega$-servers in the process.

If $P = !a(x).\bar{x}\langle z \rangle$ then $\Omega_\Sigma^P(a) = \langle z \rangle$
Symmetric $\mathcal{R}$ is a *discreet* bisimulation if $P \mathcal{R} Q$ implies:
Discrete Bisimulation

Symmetric $\mathcal{R}$ is a discrete bisimulation if $P \mathcal{R} Q$ implies:

1. If $P \xrightarrow{\mu} P'$ where $\mu$ is silent or on a plain/linear channel:
   - $Q \xrightarrow{\hat{\mu}} Q'$ and $P' \mathcal{R} Q'$.
Symmetric \( R \) is a discreet bisimulation if \( P R Q \) implies:

1. If \( P \xrightarrow{\mu} P' \) where \( \mu \) is silent or on a plain/linear channel:
   - \( Q \xrightarrow{\mu} Q' \) and \( P'RQ' \).

2. \( \forall u \omega \)-input in \( P \) (say \( P \xrightarrow{u(\bar{x})} P' \))
   - \( \Omega_P^\Sigma (u) = \Omega_Q^\Sigma (u) \),
Symmetric $\mathcal{R}$ is a *discrete* bisimulation if $P \mathcal{R} Q$ implies:

1. If $P \xrightarrow{\mu} P'$ where $\mu$ is silent or on a plain/linear channel:
   - $Q \xrightarrow{\hat{\mu}} Q'$ and $P' \mathcal{R} Q'$.

2. $\forall u \ \omega$-input in $P$ (say $P \xrightarrow{u(\check{x})} P'$)
   - $\Omega_P^\Sigma (u) = \Omega_Q^\Sigma (u)$,
   - Safety: $P' \mathcal{R} P'$. 
Symmetric \( R \) is a \textit{discrete} bisimulation if \( P R Q \) implies:

1. If \( P \xrightarrow{\mu} P' \) where \( \mu \) is silent or on a plain/linear channel:
   - \( Q \xrightarrow{\hat{\mu}} Q' \) and \( P' R Q' \).

2. \( \forall u \) \( \omega \)-input in \( P \) (say \( P \xrightarrow{u(x)} P' \))
   - \( \Omega_P^\Sigma (u) = \Omega_Q^\Sigma (u) \),
   - Safety: \( P' R P' \),
   - Determinism: \( \exists! \xi \) s.t. \( \Omega_P^\Sigma (u) = \xi \),
Discrete Bisimulation

Symmetric $R$ is a \textit{discrete} bisimulation if $P \mathcal{R} Q$ implies:

1. If $P \xrightarrow{\mu} P'$ where $\mu$ is silent or on a plain/linear channel:
   - $Q \xrightarrow{\hat{\mu}} Q'$ and $P' \mathcal{R} Q'$.

2. $\forall u \omega$-input in $P$ (say $P \xrightarrow{u(\bar{x})} P'$)
   - $\Omega^\Sigma_P(u) = \Omega^\Sigma_Q(u)$,
   - Safety: $P' \mathcal{R} P'$,
   - Determinism: $\exists! \xi$ s.t. $\Omega^\Sigma_P(u) = \xi$,
   - Closeness: $P \mathcal{R} (\nu \bar{x}) P'$.
Discrete Bisimulation

Symmetric $\mathcal{R}$ is a *discrete* bisimulation if $P \mathcal{R} Q$ implies:

1. If $P \xrightarrow{\mu} P'$ where $\mu$ is silent or on a plain/linear channel:
   - $Q \xrightarrow{\hat{\mu}} Q'$ and $P' \mathcal{R} Q'$.

2. $\forall u$ $\omega$-input in $P$ (say $P \xrightarrow{u(\tilde{x})} P'$)
   - $\Omega_P^\Sigma (u) = \Omega_Q^\Sigma (u)$,
   - Safety: $P' \mathcal{R} P'$,
   - Determinism: $\exists! \xi$ s.t. $\Omega_P^\Sigma (u) = \xi$,
   - Closeness: $P \mathcal{R} (\nu \tilde{x}) P'$.

3. $\forall u$ $\omega$-output in $P$:
   - $(L_\sigma (u) \mid P) \mathcal{R} Q$.
Channel Types

Definition

A Channel Type is a structure of the form:

\[ a : ((\tilde{\sigma})^m, \rho, \tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}) \]

- \( \tilde{\sigma} \): Parameters
- \( m \): Action Mode
- \( \rho \): Protocol
- \( \tilde{\alpha} \): Receptiveness
- \( \tilde{\beta} \): Input Responsiveness
- \( \tilde{\gamma} \): Output Responsiveness
Inter-Class Interactions

Highly constrained $\omega$ and unreliable plain names can interact.
Inter-Class Interactions

Highly constrained $\omega$ and unreliable plain names can interact.

$\omega$ over $p$:

$$(\nu p) \ (\overline{p\langle u \rangle}.! u \ \cdots \ | \ \overline{p\langle v \rangle}.! v \ \cdots \ | \ p(x). \ \cdots \ | \ p(y). \ \cdots )$$
Highly constrained $\omega$ and unreliable plain names can interact.

- $\omega$ over $p$:
  $$(\nu p) \left( \overline{p\langle u \rangle}.! u \cdots \mid \overline{p\langle v \rangle}.! v \cdots \mid p(x).\cdots \mid p(y).\cdots \right)$$

- $p$ over $\omega$:
  $$\overline{u\langle l, p, q \rangle} \mid ! u(x, y, z).\overline{x\langle y \rangle}$$
Inter-Class Interactions

Highly constrained $\omega$ and unreliable plain names can interact.

- $\omega$ over $p$:

$$\langle \nu p \rangle \ (\langle p \rangle u \! u \cdots \mid \langle p \rangle v \! v \cdots \mid p(x) \cdots \mid p(y) \cdots)$$

- $p$ over $\omega$:

$$\langle l, p, q \rangle \mid ! u(x, y, z).x$$

- Still, $\omega$’s discreetness guarantees are preserved.
Anatomy of one Rule

\[
\begin{align*}
A \vdash_{\pi} P \\
(\nu\tilde{x}) \left( l : (\downarrow_1)((\tilde{\sigma})^{\uparrow_1}, \rho, \emptyset, (\tilde{x})) + l.\hat{l}.\rho(\tilde{x} : \tilde{\sigma}) \odot l.A \right) \vdash_{\pi} l(\tilde{x}).P \quad (\text{INP}_1)
\end{align*}
\]
Anatomy of one Rule

\[
\frac{A \vdash_{\pi} P}{(\nu \tilde{x}) \left( l : (\downarrow_1)(\tilde{\sigma})^{\downarrow_1}, \rho, \emptyset, (\tilde{x}) \right) + l.\hat{l}.\rho(\tilde{x} : \tilde{\sigma}) \odot l.A} \vdash_{\pi} l(\tilde{x}).P} \quad (\text{INP}_1)
\]

- \(l\) receptive now; responsive when parameters are ready
Anatomy of one Rule

\[
\frac{A \vdash_\pi P}{(\nu \tilde{x}) \left( l : (\downarrow_1)((\tilde{\sigma})^{\downarrow_1}, \rho, \emptyset, (\tilde{x})) + l.\hat{l}.\rho(\tilde{x} : \tilde{\sigma}) \odot l.A \right) \vdash_\pi l(\tilde{x}).P} \quad \text{(INP}_1\text{)}
\]

- \( l \) receptive now; responsive when parameters are ready
- Remote parameters
Anatomy of one Rule

\[
A \vdash_{\pi} P
\]

\[(\nu \tilde{x}) \left( l : (\downarrow_1)(\langle \tilde{\sigma}\rangle_1, \rho, \emptyset, (\tilde{x})) + l.\hat{l}.\rho(\tilde{x} : \tilde{\sigma}) \circ I.A \right) \vdash_{\pi} l(\tilde{x}).P \quad \text{(INP} \_1)\]

- \(l\) receptive now; responsive when parameters are ready
- Remote parameters
- Continuation
Anatomy of one Rule

\[
\frac{A \vdash_\pi P}{(\nu \tilde{x})(l : (\downarrow_1)((\tilde{\sigma})^\downarrow^1, \rho, \emptyset, (\tilde{x})) + l.\hat{l}.\rho(\tilde{x} : \tilde{\sigma}) \odot l.A) \vdash_\pi l(\tilde{x}).P} \quad (\text{INP}_1)
\]

- \(l\) receptive now; responsive when parameters are ready
- Remote parameters
- Continuation
- \(P\) must provide resources specified in \(\tilde{\sigma}\)
(Expected) Results

Discreetness:

**Theorem**

\[(A \vdash_\pi P) \Rightarrow (P \approx_R P)\]
(Expected) Results

Discreetness:

**Theorem**

\[(A \vdash_{\pi} P) \Rightarrow (P \approx_{R} P)\]

Soundness:

**Theorem**

\[(A \vdash_{\pi} P) \land \Sigma_{A}(a) = (\cdots)^{1} \Rightarrow (P \xrightarrow{(\nu z)\ a}\overrightarrow{x})\]

\[(A \vdash_{\pi} P) \land \Sigma_{A}(a) = (\cdots)^{1} \Rightarrow (P \xrightarrow{a}\overleftarrow{x})\]
(Expected) Results

Discreetness:

**Theorem**

\[(A \vdash_{\pi} P) \Rightarrow (P \approx_{R} P)\]

Soundness:

**Theorem**

\[
(A \vdash_{\pi} P) \land \Sigma_{A}(a) = (\cdots)^{\uparrow_{1}} \Rightarrow (P \xrightarrow{\nu z \overline{a}(\overline{x})})
\]

\[
(A \vdash_{\pi} P) \land \Sigma_{A}(a) = (\cdots)^{\downarrow_{1}} \Rightarrow (P \xrightarrow{a(\overline{x})})
\]

Safety:

**Theorem**

\[
(A \vdash_{\pi} P) \land (P \rightarrow P') \Rightarrow (A \vdash_{\pi} P')
\]
Thank You (Obrigado, Shukria, Kiitos, Merci)! 

The paper can be found at http://gamboni.org/maxime/