ρ-calculus modulo

Clément HOUTMANN, ENS Cachan & LORIA

31st May 2005

Definition

Terms :

$$
\begin{array}{rll}
T ::= & & \\
& | x & \text{Variable} \\
& | a & \text{Constant} \\
& | T.T & \text{Application} \\
& | T \to T & \text{Abstraction} \\
& | T \mid T & \text{Structure (non-determinism)}
\end{array}
$$

Definition

The classical rules of the ρ-calculus are :

$$(\rho) \quad (M_1 \to M_2).M_3 \quad \mapsto \quad \sigma_1 M_2 \mid \ldots \mid \sigma_n M_2 \quad \text{with } \{\sigma_i\} = \mathcal{S}ol(M_1 \ll M_3)$$

$$(\delta) \quad (M_1 \mid M_2).M_3 \quad \mapsto \quad M_1.M_3 \mid M_2.M_3$$

Theories

Theories can be plugged inside the calculus through the matching definition :

$$\mathcal{S}ol(P \ll M) = \{\sigma / \sigma P =_T M\}$$

where T can be any theory.

About the patterns

- Is it safe to use terms such as $(x \mid y \longrightarrow x)$?
  What is the meaning of $(x \mid y \longrightarrow x).(a \mid (a \mid b))$ anyway ?
  Is $\mid$ ACI ? AC ? A ? syntactic ?

About the patterns

- Is it safe to use terms such as $(x \mid y \longrightarrow x)$?
  What is the meaning of $(x \mid y \longrightarrow x).(a \mid (a \mid b))$ anyway ?
  Is $\mid$ ACI ? AC ? A ? syntactic ?
- Should we introduce higher order matching ?

About the patterns

Patterns :

$$
\begin{array}{rll}
P ::= & & \\
& |x & \text{Variable} \\
& |a & \text{Constant} \\
& |(\ldots(f.P).P\ldots) & \text{First Order terms}
\end{array}
$$

The non-linear terms problem

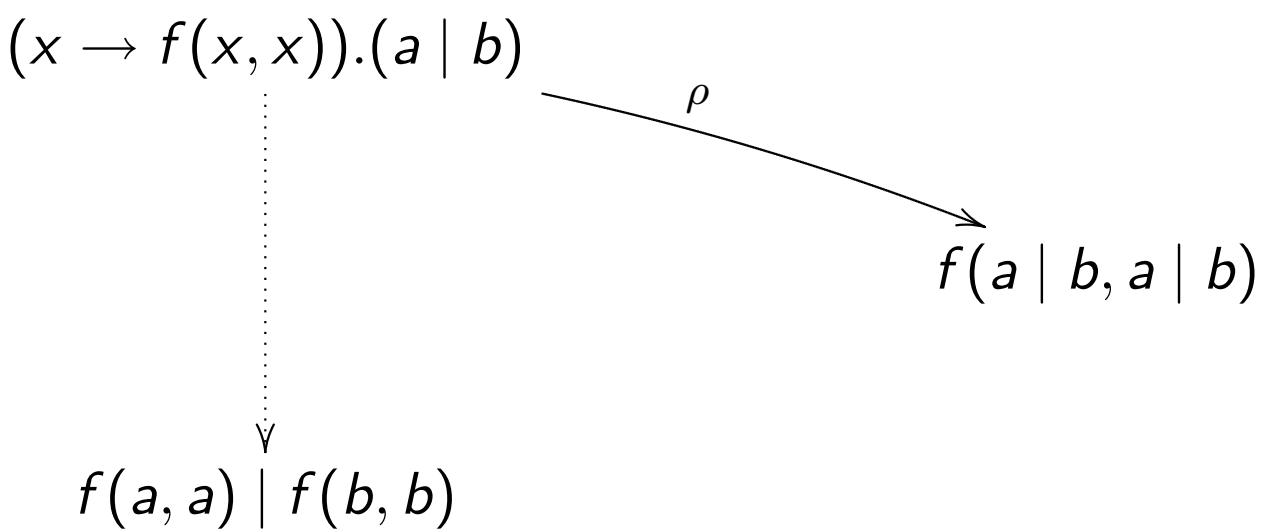
$$(x \to f(x,x)).(a \mid b)$$

The non-linear terms problem

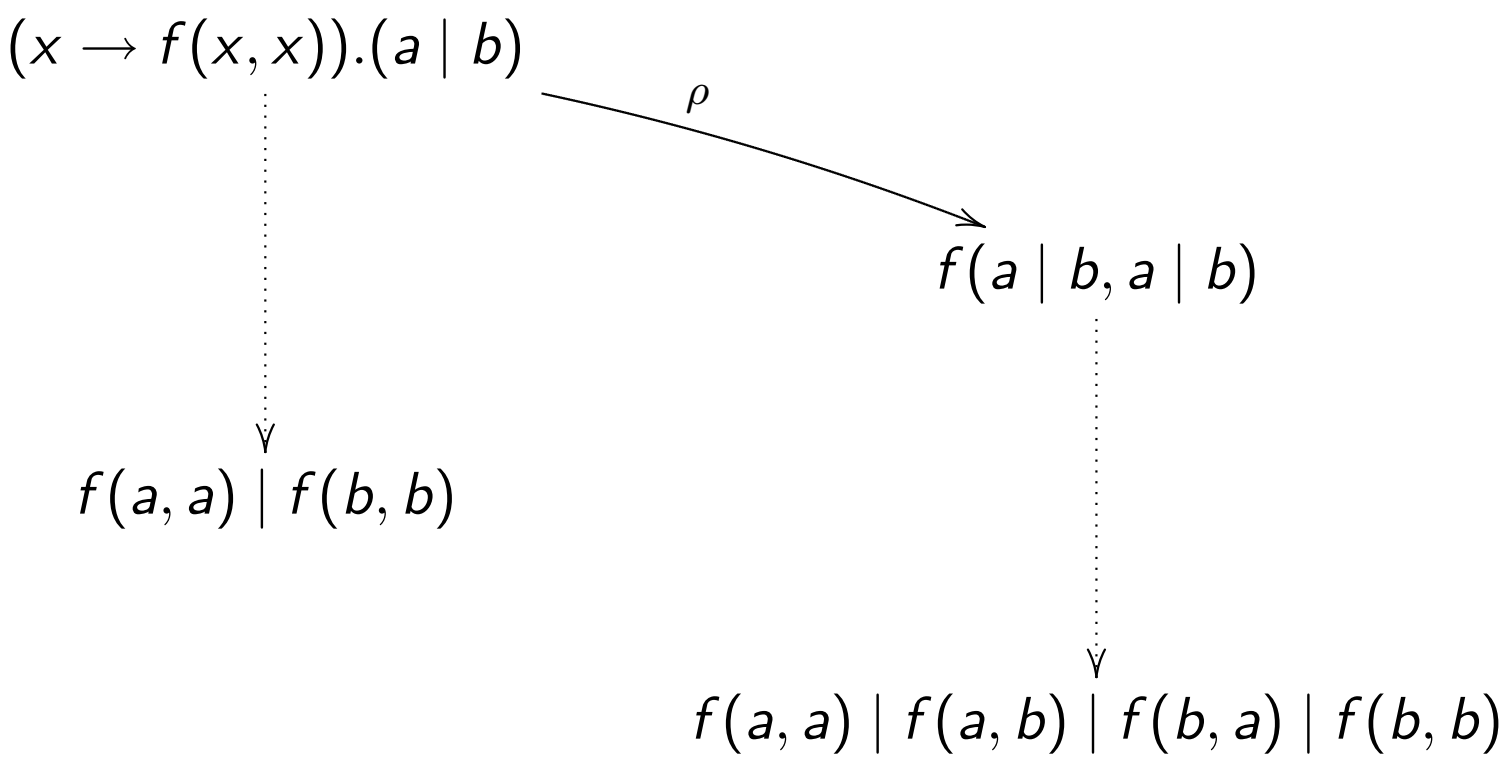
$$(x \to f(x,x)).(a \mid b)$$

$$f(a,a) \mid f(b,b)$$

The non-linear terms problem



The non-linear terms problem



Solutions may be to...

- prohibit non-linear terms

Solutions may be to...

- prohibit non-linear terms
- handle *non-deterministic* terms inside the $\mathcal{S}ol$ definition

Solutions may be to...

- prohibit non-linear terms
- handle *non-deterministic* terms inside the $\mathcal{S}ol$ definition
- other solution ?

Some terms could be called "non-deterministic"

- $a \mid b$

Some terms could be called "non-deterministic"

- $a \mid b$
- $(f(x, y) \longrightarrow (x \mid y)).f(a, b)$

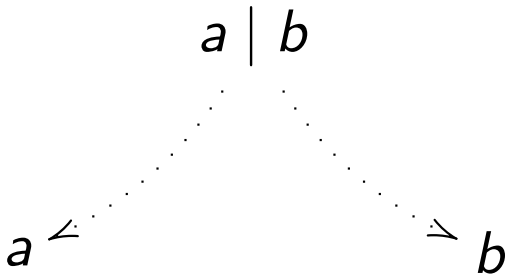
Some terms could be called "non-deterministic"

- $a \mid b$
- $(f(x, y) \longrightarrow (x \mid y)).f(a, b)$
- $(x + y \longrightarrow x).(a + b)$ where $+$ is commutative

Some terms could be called "non-deterministic"

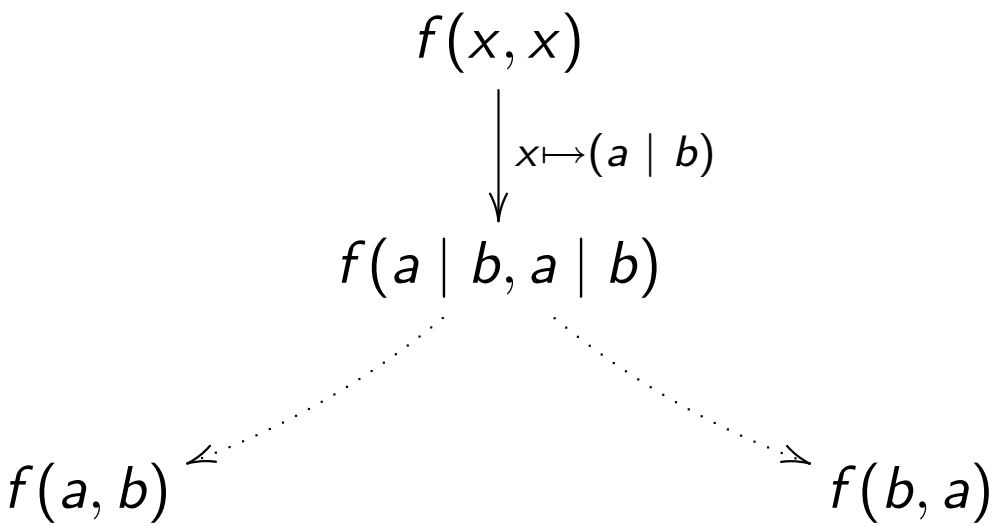
- $a \mid b$
- $(f(x, y) \longrightarrow (x \mid y)).f(a, b)$
- $(x + y \longrightarrow x).(a + b)$ where $+$ is commutative
- ...

They represent several values



They represent several values

... and substituting a variable for such a term is dangerous



Values

Values $\approx$ Deterministic terms

$$
\begin{array}{rll}
V ::= & & \\
& | x & \text{Variable} \\
& | a & \text{Constant} \\
& | P \longrightarrow T & \text{Abstraction}
\end{array}
$$

Value substitution

- A value substitution is a substitution that only subsitutes variables for values.

Value substitution

- A value substitution is a substitution that only subsitutes variables for values.
- Then :

$$Sol(P \ll M) = \{\sigma / \sigma P =_T M \text{ and } \sigma \text{ value substitution}\}$$

Result

No more interference between non-linear terms and non-determinism :
$(x \longrightarrow f(x,x)).(a \mid b)$ cannot be reduced anymore.

Result

No more interference between non-linear terms and non-determinism :
$(x \longrightarrow f(x,x)).(a \mid b)$ cannot be reduced anymore.

- We add the rule

$$(\gamma) \quad N.(M_1 \mid M_2) \quad \mapsto \quad (N.M_1) \mid (N.M_2)$$

Interaction between a theory and the ρ-calculus

"::" is here an associative symbol.

$$(z \to ((x :: y \to x).(a :: z))).(b :: c)$$

Interaction between a theory and the ρ-calculus

"::" is here an associative symbol.

$$(z \to ((x :: y \to x).(a :: z))).(b :: c)$$

$$\downarrow \rho$$

$$(x :: y \to x).(a :: (b :: c))$$

Interaction between a theory and the ρ-calculus

" ::" is here an associative symbol.

$$(z \to ((x :: y \to x).(a :: z))).(b :: c)$$

$$\downarrow \rho$$

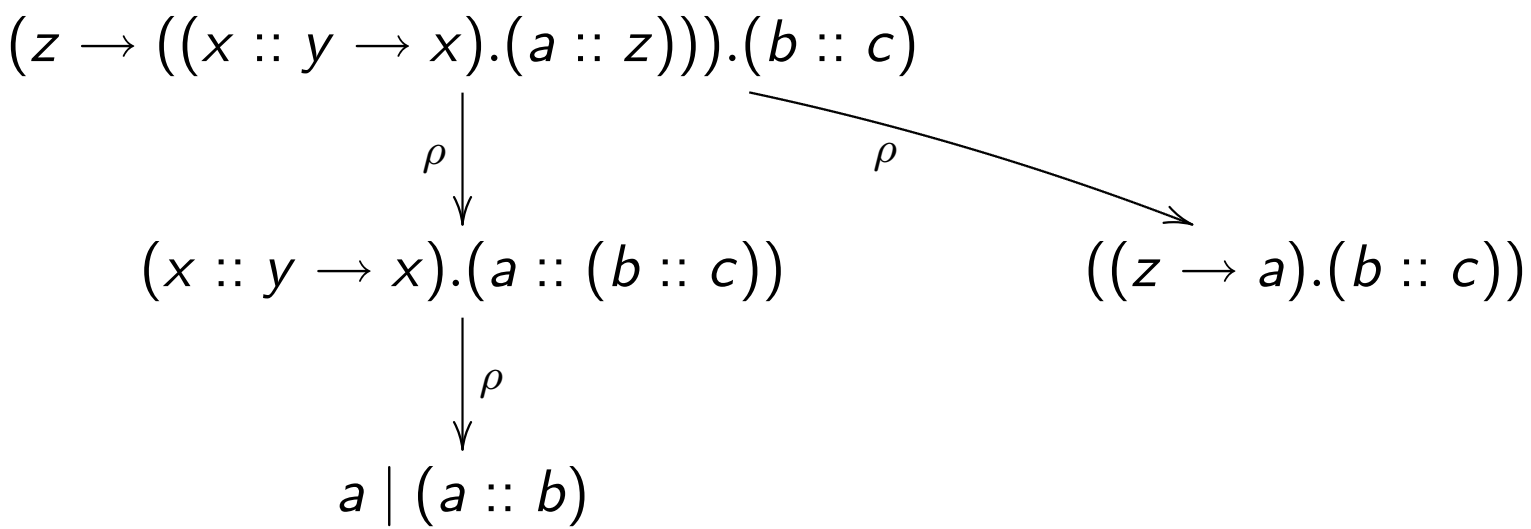
$$(x :: y \to x).(a :: (b :: c))$$

$$\downarrow \rho$$

$$a \mid (a :: b)$$

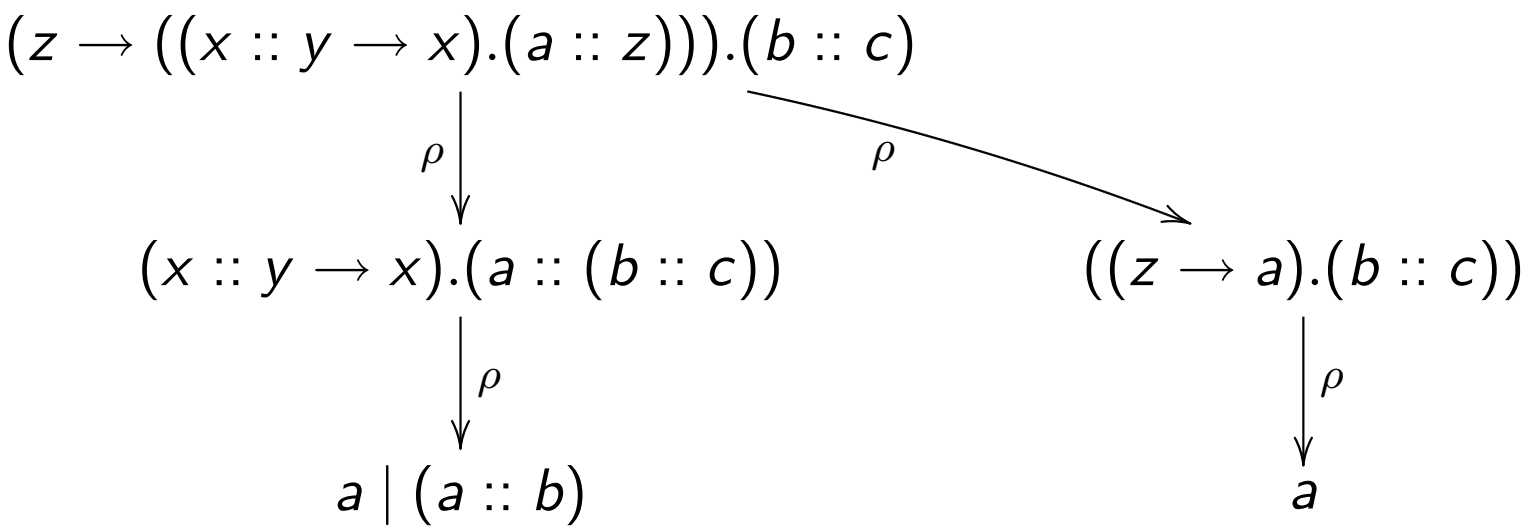
Interaction between a theory and the ρ-calculus

" ::" is here an associative symbol.



Interaction between a theory and the ρ-calculus

" ::" is here an associative symbol.



Interaction between a theory and the ρ-calculus

- Solutions of a pattern matching problem can change if we substitute a variable for a value...

$$x :: y \ll a :: z \qquad \neq \qquad x :: y \ll a :: (b :: c)$$

Interaction between a theory and the ρ-calculus

- Solutions of a pattern matching problem can change if we substitute a variable for a value...

$$x :: y \ll a :: z \qquad \neq \qquad x :: y \ll a :: (b :: c)$$

- Solutions :

Interaction between a theory and the ρ-calculus

- Solutions of a pattern matching problem can change if we substitute a variable for a value...

$$x :: y \ll a :: z \qquad \neq \qquad x :: y \ll a :: (b :: c)$$

- Solutions :
  - a stronger pattern matching that considers any variable x as a potential $M_1 :: M_2$

Interaction between a theory and the ρ-calculus

- Solutions of a pattern matching problem can change if we substitute a variable for a value...

$$x :: y \ll a :: z \qquad \neq \qquad x :: y \ll a :: (b :: c)$$

- Solutions :
  - a stronger pattern matching that considers any variable x as a potential $M_1 :: M_2$
  - to prohibit a certain class of theories (such as associativity !)

Interaction between a theory and the ρ-calculus

- Solutions of a pattern matching problem can change if we substitute a variable for a value...

$$x :: y \ll a :: z \qquad \neq \qquad x :: y \ll a :: (b :: c)$$

- Solutions :
  - a stronger pattern matching that considers any variable x as a potential $M_1 :: M_2$
  - to prohibit a certain class of theories (such as associativity !)
  - variable are not values

rho-terms

Definition of. . .

- Patterns,
- Terms,
- Values.

- Patterns :

$$
\begin{array}{rll}
P ::= & & \\
& |x & \text{Variable} \\
& |a & \text{Constant} \\
& |(\ldots(f.P).P\ldots) & \text{First Order terms}
\end{array}
$$

- Patterns :

$$
\begin{array}{rll}
P ::= & & \\
& |x & \text{Variable} \\
& |a & \text{Constant} \\
& |(\ldots(f.P).P\ldots) & \text{First Order terms}
\end{array}
$$

- Terms :

$$
\begin{array}{rll}
T ::= & & \\
& |x & \text{Variable} \\
& |a & \text{Constant} \\
& |T.T & \text{Application} \\
& |P \rightarrow T & \text{Abstraction} \\
& |T \mid T & \text{Structure (non-determinism)}
\end{array}
$$

- Patterns :

$$
\begin{array}{rll}
P ::= & & \\
& | x & \text{Variable} \\
& | a & \text{Constant} \\
& | (\ldots(f.P).P\ldots) & \text{First Order terms}
\end{array}
$$

- Terms :

$$
\begin{array}{rll}
T ::= & & \\
& | x & \text{Variable} \\
& | a & \text{Constant} \\
& | T.T & \text{Application} \\
& | P \rightarrow T & \text{Abstraction} \\
& | T \mid T & \text{Structure (non-determinism)}
\end{array}
$$

- Values :

$$
\begin{array}{rll}
V ::= & & \\
& | a & \text{Constant} \\
& | (\ldots(f.V).V\ldots) & \text{First Order terms} \\
& | P \rightarrow T & \text{Abstraction}
\end{array}
$$

The ρ-calculus modulo

$$\rho : \quad (P \to M).V \quad \mapsto \quad \sigma_1 M \mid \ldots \mid \sigma_n M \text{ avec } \begin{cases} \{\sigma_i\}_i = \mathcal{S}ol(P \ll V) \\ n \geq 1 \end{cases}$$

$$\gamma : \quad M.(N_1 \mid N_2) \quad \mapsto \quad (M.N_1) \mid (M.N_2)$$

$$\delta : \quad (N_1 \mid N_2).V \quad \mapsto \quad (N_1.V) \mid (N_2.V)$$

Result

- The obtained calculus is simple (3 rules).

Result

- The obtained calculus is simple (3 rules).
- The obtained calculus is confluent.

Result

- The obtained calculus is simple (3 rules).
- The obtained calculus is confluent.
- We have no restriction on linearity or theories.

Result

- The obtained calculus is simple (3 rules).
- The obtained calculus is confluent.
- We have no restriction on linearity or theories.
- Good expressiveness : call-by-value lambda-calculus, fixpoints, Caml

Fixpoints

let *plus* be

$$REC \longrightarrow \left(\begin{array}{l} (add.z.Y \mid \longrightarrow Y) \mid \\ (add.(s.X).Y \longrightarrow s.((REC.REC).(add.X.Y))) \end{array} \right)$$

in

Fixpoints

let *plus* be

$$REC \longrightarrow \left(\begin{array}{l} (add.z.Y \mid \longrightarrow Y) \mid \\ (add.(s.X).Y \longrightarrow s.((REC.REC).(add.X.Y))) \end{array} \right)$$

in

$$(plus.plus).(add.(s.(s.z)).(s.z))$$

Fixpoints

let *plus* be

$$REC \longrightarrow \left(\begin{array}{l} (add.z.Y| \longrightarrow Y) \mid \\ (add.(s.X).Y \longrightarrow s.((REC.REC).(add.X.Y))) \end{array} \right)$$

in

$$(plus.plus).(add.(s.(s.z)).(s.z))$$
$$\downarrow$$
$$\left(\begin{array}{l} (add.z.Y| \longrightarrow Y) \mid \\ (add.(s.X).Y \longrightarrow s.((plus.plus).(add.X.Y))) \end{array} \right).(add.(s.(s.z)).(s.z))$$

Fixpoints

let *plus* be

$$REC \longrightarrow \left(\begin{array}{l} (add.z.Y| \longrightarrow Y) \mid \\ (add.(s.X).Y \longrightarrow s.((REC.REC).(add.X.Y))) \end{array} \right)$$

in

$$(plus.plus).(add.(s.(s.z)).(s.z))$$
$$\downarrow$$
$$\left(\begin{array}{l} (add.z.Y| \longrightarrow Y) \mid \\ (add.(s.X).Y \longrightarrow s.((plus.plus).(add.X.Y))) \end{array} \right).(add.(s.(s.z)).(s.z))$$
$$\downarrow$$
$$\left(\begin{array}{l} (add.z.Y \longrightarrow Y).(add.(s.(s.z)).(s.z)) \mid \\ (add.(s.X).Y \longrightarrow s.((plus.plus).(add.X.Y))).(add.(s.(s.z)).(s.z)) \end{array} \right)$$

Fixpoints

let *plus* be

$$REC \longrightarrow \left(\begin{array}{l} (add.z.Y| \longrightarrow Y) \mid \\ (add.(s.X).Y \longrightarrow s.((REC.REC).(add.X.Y))) \end{array} \right)$$

in

$$(plus.plus).(add.(s.(s.z)).(s.z))$$
$$\downarrow$$
$$\left(\begin{array}{l} (add.z.Y| \longrightarrow Y) \mid \\ (add.(s.X).Y \longrightarrow s.((plus.plus).(add.X.Y))) \end{array} \right).(add.(s.(s.z)).(s.z))$$
$$\downarrow$$
$$\left(\begin{array}{l} (add.z.Y \longrightarrow Y).(add.(s.(s.z)).(s.z)) \mid \\ (add.(s.X).Y \longrightarrow s.((plus.plus).(add.X.Y))).(add.(s.(s.z)).(s.z)) \end{array} \right)$$
$$\downarrow$$
$$\left(\begin{array}{l} (add.z.Y \longrightarrow Y).(add.(s.(s.z)).(s.z)) \mid \\ s.((plus.plus).(add.(s.z).(s.z))) \end{array} \right)$$

Fixpoints

let *plus* be

$$REC \longrightarrow \left(\begin{array}{l} (add.z.Y| \longrightarrow Y) \ | \\ (add.(s.X).Y \longrightarrow s.((REC.REC).(add.X.Y))) \end{array} \right)$$

in

$$(plus.plus).(add.(s.(s.z)).(s.z))$$
$$\downarrow$$
$$\left(\begin{array}{l} (add.z.Y| \longrightarrow Y) \ | \\ (add.(s.X).Y \longrightarrow s.((plus.plus).(add.X.Y))) \end{array} \right).(add.(s.(s.z)).(s.z))$$
$$\downarrow$$
$$\left(\begin{array}{l} (add.z.Y \longrightarrow Y).(add.(s.(s.z)).(s.z)) \ | \\ (add.(s.X).Y \longrightarrow s.((plus.plus).(add.X.Y))).(add.(s.(s.z)).(s.z)) \end{array} \right)$$
$$\downarrow$$
$$\left(\begin{array}{l} (add.z.Y \longrightarrow Y).(add.(s.(s.z)).(s.z)) \ | \\ s.((plus.plus).(add.(s.z).(s.z))) \end{array} \right)$$
$$\downarrow$$
$$\ldots$$

Disadvantage

To reduce $(z \longrightarrow a).T$, we first have to reduce T to a value...

Disadvantage

To reduce $(z \longrightarrow a).T$, we first have to reduce T to a value...

- it could be very long...

Disadvantage

To reduce $(z \longrightarrow a).T$, we first have to reduce T to a value...

- it could be very long...
- ... even impossible ($T = \Omega$)

What for ?

- Call-by-value style reduction has disadvantages

What for ?

- Call-by-value style reduction has disadvantages
- Solution : constraint introduction

Introduction of constraints

Introduction :

$$(\rho) \qquad (P \longrightarrow M).V \quad \mapsto \quad \sigma_1 M \mid \ldots \mid \sigma_n M$$

Introduction of constraints

Introduction :

$$
\begin{array}{lrcl}
(\rho) & (P \longrightarrow M).V & \mapsto & \sigma_1 M \mid \ldots \mid \sigma_n M \\
 & & \Downarrow & \\
(\rho) & (P \longrightarrow M).N & \mapsto & M[P \ll N] \\
(eval) & M[P \ll V] & \mapsto & \sigma_1 M \mid \ldots \mid \sigma_n M
\end{array}
$$

Introduction of constraints

Introduction :

$$
\begin{array}{lrcl}
(\rho) & (P \to M).V & \mapsto & \sigma_1 M \mid \ldots \mid \sigma_n M \\
 & & \Downarrow & \\
(\rho) & (P \to M).N & \mapsto & M[P \ll N] \\
(eval) & M[P \ll V] & \mapsto & \sigma_1 M \mid \ldots \mid \sigma_n M \\
 & & & \\
(delete) & M[P \ll N] & \mapsto & M \\
 & & & \text{When } FV(M) \cap dom(P) = \emptyset
\end{array}
$$

Improving

$$x[y \ll T] \quad \mapsto \quad x$$

Improving

$$\begin{aligned} x[y \ll T] &\mapsto x \\ x[f(x,y) \ll f(a,T)] &\mapsto_? a \end{aligned}$$

. . . We need explicit constraint resolution to improve completely the calculus.

Results

We introduced a ρ-calculus modulo

Results

We introduced a ρ-calculus modulo

- confluent

Results

We introduced a ρ-calculus modulo

- confluent
- that handles any theory

Results

We introduced a ρ-calculus modulo

- confluent
- that handles any theory
- that could be freed from strategy reduction disadvantages...

Further work. . .

Further work...

- ...on expressiveness of the ρ-calculus modulo

Further work...

- ...on expressiveness of the ρ-calculus modulo
- ...on explicit constraints resolution

Further work. . .

- . . . on expressiveness of the ρ-calculus modulo
- . . . on explicit constraints resolution
- . . . on extensions of the ρ-calculus modulo