

Type Systems for the Termination of Mobile Processes

Romain Demangeon

ENS Lyon

(Joint work with D.Hirschkoff, ENS Lyon and D.Sangiorgi, University of Bologna)

An overview of recent results

Starting point

- *Ensuring termination by typability* [06] (Deng, Sangiorgi)

Our contributions

- *On the complexity of termination inference for processes* [TGC'07] (Demangeon, Hirschhoff, Kobayashi, Sangiorgi).
- *Static and dynamic typing for the termination of mobile processes* [TCS'08] (Demangeon, Hirschhoff, Sangiorgi).
- *Termination in higher-order concurrent calculi* [FSEN'09] (Demangeon, Hirschhoff, Sangiorgi).

- 1 Termination in the π -calculus
- 2 Weight-based Type Systems
- 3 Allowing Forms of Recursion
- 4 Higher-order Mobile Calculi

- 1 Termination in the π -calculus
- 2 Weight-based Type Systems
- 3 Allowing Forms of Recursion
- 4 Higher-order Mobile Calculi

Termination for mobile processes

Termination

- Deciding that a given system cannot reduce anymore.
- Existence of multiple techniques for sequential programs (rewriting theory, proof theory).
- Termination of mobile systems is challenging. Structures evolve at run-time.

The Formalism

π -calculus: studying termination from different points of view within a unique framework:

- First-order: termination and name-passing.
- Higher-order: termination and process-passing.

π -calculus

- Syntax: $P ::= \mathbf{0} \mid \bar{a}\langle v \rangle.P \mid a(x).P \mid !a(x).P \mid (P \mid P)$
- Only replicated inputs.
- Two reductions:

$$\frac{}{a(x).P \mid \bar{a}\langle v \rangle.Q \rightarrow P[v/x] \mid Q}$$

$$\frac{}{!a(x).P \mid \bar{a}\langle v \rangle.Q \rightarrow !a(x).P \mid P[v/x] \mid Q}$$

π -calculus

- Syntax: $P ::= \mathbf{0} \mid \bar{a}\langle v \rangle.P \mid a(x).P \mid !a(x).P \mid (P \mid P)$
- Only replicated inputs.
- Two reductions:

$$\frac{}{a(x).P \mid \bar{a}\langle v \rangle.Q \rightarrow P[v/x] \mid Q}$$

$$\frac{}{!a(x).P \mid \bar{a}\langle v \rangle.Q \rightarrow !a(x).P \mid P[v/x] \mid Q}$$

Replication: source of divergence

with C.C.S notation

$$P_1 = \bar{a} \mid !a.\bar{a} \quad P_2 = \bar{a} \mid !a.\bar{b} \mid !b.\bar{a} \quad P_3 = \bar{c}\langle a \rangle \mid \bar{a} \mid c(y).!y.\bar{a}$$

π -calculus

- Syntax: $P ::= \mathbf{0} \mid \bar{a}\langle v \rangle.P \mid a(x).P \mid !a(x).P \mid (P \mid P)$
- Only replicated inputs.
- Two reductions:

$$\frac{}{a(x).P \mid \bar{a}\langle v \rangle.Q \rightarrow P[v/x] \mid Q}$$

$$\frac{}{!a(x).P \mid \bar{a}\langle v \rangle.Q \rightarrow !a(x).P \mid P[v/x] \mid Q}$$

Replication: source of divergence

with C.C.S notation

$$P_1 = \bar{a} \mid !a.\bar{a} \quad P_2 = \bar{a} \mid !a.\bar{b} \mid !b.\bar{a} \quad P_3 = \bar{c}\langle a \rangle \mid \bar{a} \mid c(y).!y.\bar{a}$$

- 1 Termination in the π -calculus
- 2 Weight-based Type Systems**
- 3 Allowing Forms of Recursion
- 4 Higher-order Mobile Calculi

S1: the original type system

Principles

- Assigning *levels* (integers) to channels.
- Assigning levels to processes: maximum level of a name used in output (not appearing under a replication).

$$Q_1 = !a.(\bar{b} \mid \bar{b} \mid \bar{c}) \mid !b.\bar{c} \mid \bar{a} \mid \bar{c}$$

- Typing $!a^n(\tilde{x}).P$ with $P : m$ when $n > m$.
- Firing a replicated input: trading an output for several smaller outputs.

S1: the original type system

Principles

- Assigning *levels* (integers) to channels.
- Assigning levels to processes: maximum level of a name used in output (not appearing under a replication).

$$Q_1 = !a^3.(\bar{b}^2 \mid \bar{b}^2 \mid \bar{c}^1) \mid !b^2.\bar{c}^1 \mid \bar{a}^3 \mid \bar{c}^1$$

- Typing $!a^n(\tilde{x}).P$ with $P : m$ when $n > m$.
- Firing a replicated input: trading an output for several smaller outputs.

The System S1

Typing rule for replicated input

$$\frac{\Gamma, \tilde{x} : \tilde{T} \vdash P : m \quad m < n}{\Gamma \vdash !a^n(\tilde{x}).P : 0} \quad (\text{S1})$$

Giving level 0 to replicated process

- Guarded by a replication.
- $!a(x).(\bar{b} \mid !x(y).P)$: outputs in P do not count for the replication on a .

Soundness

- If P is S1-typable, P terminates
- (*Proof*) the multiset of the levels of every outputs (not under a replication) is decreasing for the multiset ordering.

- $Q_1 = !a^3.(\bar{b}^2 \mid \bar{b}^2 \mid \bar{c}^1) \mid !b^2.\bar{c}^1 \mid \bar{a}^3 \mid \bar{c}^1$
 $[1, 0, 1] \rightarrow [0, 2, 2] \rightarrow [0, 1, 3] \rightarrow [0, 0, 4]$.
- $P_1 = \bar{a}^n \mid !a^n.\bar{a}^n$ not S1-typable as it forces $n > n$.
- Similarly $P_2 = \bar{a}^n \mid !a^n.\bar{b}^m \mid !b^m.\bar{a}^n$ gives $n > m > n$.

Soundness

- If P is S1-typable, P terminates
- (*Proof*) the multiset of the levels of every outputs (not under a replication) is decreasing for the multiset ordering.

- $Q_1 = !a^3.(\bar{b}^2 \mid \bar{b}^2 \mid \bar{c}^1) \mid !b^2.\bar{c}^1 \mid \bar{a}^3 \mid \bar{c}^1$
 $[1, 0, 1] \rightarrow [0, 2, 2] \rightarrow [0, 1, 3] \rightarrow [0, 0, 4]$.
- $P_1 = \bar{a}^n \mid !a^n.\bar{a}^n$ not S1-typable as it forces $n > n$.
- Similarly $P_2 = \bar{a}^n \mid !a^n.\bar{b}^m \mid !b^m.\bar{a}^n$ gives $n > m > n$.
- In $P_3 = \bar{c}^k\langle a \rangle \mid \bar{a}^n x \mid c^k(y).!y^m.\bar{a}^n$, the type of c forces $n = m$.

Soundness

- If P is S1-typable, P terminates
 - (*Proof*) the multiset of the levels of every outputs (not under a replication) is decreasing for the multiset ordering.
-
- $Q_1 = !a^3.(\bar{b}^2 \mid \bar{b}^2 \mid \bar{c}^1) \mid !b^2.\bar{c}^1 \mid \bar{a}^3 \mid \bar{c}^1$
 $[1, 0, 1] \rightarrow [0, 2, 2] \rightarrow [0, 1, 3] \rightarrow [0, 0, 4]$.
 - $P_1 = \bar{a}^n \mid !a^n.\bar{a}^n$ not S1-typable as it forces $n > n$.
 - Similarly $P_2 = \bar{a}^n \mid !a^n.\bar{b}^m \mid !b^m.\bar{a}^n$ gives $n > m > n$.
 - In $P_3 = \bar{c}^k\langle a \rangle \mid \bar{a}^n x \mid c^k(y).!y^m.\bar{a}^n$, the type of c forces $n = m$.

Type inference

- Inference for S1 is polynomial.
- Graph where nodes are names and edges are constraints \Rightarrow topological sort gives a solution if there is one.

Length of reductions

- P typable process of size N : less than N^{N+1} reductions from P .
- $S_n = \bar{a}_1 \mid !a_1.(\bar{a}_2 \mid \bar{a}_2) \mid !a_2.(\bar{a}_3 \mid \bar{a}_3 \mid \bar{a}_3) \mid \dots$
 - Size of S_n : $\Theta(n^2)$
 - S_n can perform $\Theta(!n)$ reductions.

Type inference

- Inference for S1 is polynomial.
- Graph where nodes are names and edges are constraints \Rightarrow topological sort gives a solution if there is one.

Length of reductions

- P typable process of size N : less than N^{N+1} reductions from P .
- $S_n = \bar{a}_1 \mid !a_1.(\bar{a}_2 \mid \bar{a}_2) \mid !a_2.(\bar{a}_3 \mid \bar{a}_3 \mid \bar{a}_3) \mid \dots$
 - Size of S_n : $\Theta(n^2)$
 - S_n can perform $\Theta(!n)$ reductions.

- 1 Termination in the π -calculus
- 2 Weight-based Type Systems
- 3 Allowing Forms of Recursion**
- 4 Higher-order Mobile Calculi

S2: a more expressive system

Principles

- No recursion is allowed in S1. $!a(x).P$ typable $\Rightarrow \bar{a} \notin P$.
- Recursions:
 - sometimes desirable (modelling recursive functions)
 - sometimes innocuous like in $!a.b.\bar{a}$.
- S2: more expressive system.
 - Input sequences are taken as a whole. $!a_1(\tilde{x}_1) \dots a_k(\tilde{x}_k).P$
 - Comparing multisets of levels

$Q_2 = !a(x)^n.b(y)^m.\bar{a}^n\langle x \rangle \mid !a^n(z).\bar{b}^m\langle z \rangle \mid \bar{a}^n\langle v_1 \rangle \mid \bar{b}^m\langle v_2 \rangle$

- Ruled out by S1: $n > m$.
- Can be type-checked in S2 with $n = 2$ and $m = 1$. The first replication is typed as $\{1, 2\} >_{mul} \{2\}$.

S2: a more expressive system

Principles

- No recursion is allowed in S1. $!a(x).P$ typable $\Rightarrow \bar{a} \notin P$.
- Recursions:
 - sometimes desirable (modelling recursive functions)
 - sometimes innocuous like in $!a.b.\bar{a}$.
- S2: more expressive system.
 - Input sequences are taken as a whole. $!a_1(\tilde{x}_1) \dots a_k(\tilde{x}_k).P$
 - Comparing multisets of levels

$$Q_2 = !a(x)^n . b(y)^m . \bar{a}^n \langle x \rangle \mid !a^n(z) . \bar{b}^m \langle z \rangle \mid \bar{a}^n \langle v_1 \rangle \mid \bar{b}^m \langle v_2 \rangle$$

- Ruled out by S1: $n > m$.
- Can be type-checked in S2 with $n = 2$ and $m = 1$. The first replication is typed as $\{1, 2\} >_{mul} \{2\}$.

Typing rule for replicated inputs

$$\frac{\Gamma, \tilde{x}_1 : \tilde{T}_1, \dots, \tilde{x}_k : \tilde{T}_k \vdash P : M \quad \{n_1, \dots, n_k\} >_{mul} M}{\Gamma \vdash !a_1^{n_1}(\tilde{x}_1) \dots a_k^{n_k}(\tilde{x}_k).P : \emptyset} \quad (S2)$$

Complexity

- S1-typable \Rightarrow S2-typable.
- Inference for S2 is NP-complete.
- Polynomial if we use algebraical operations $(n_1 + \dots + n_k)$ on levels in the type system.

S3: A system using partial orders

Principles

- A system allowing replications with no decreasing in weight.
- A well-founded partial order between names of the same level ensures termination.

Typing complex structures

$!p(a, b).a.(\bar{b} \mid \bar{p}\langle a, b \rangle)$ (a and b play the same role)

- Models the behavior of a list structure.
- Typed with a low level for p , a higher level for a, b and $a \succ b$.
- Type of p : first argument dominates the second one.
- S4: extension of S3 in which we can type tree structures
 $!p(a, b, c).a.(\bar{b} \mid \bar{c} \mid \bar{p}\langle a, b, c \rangle)$. \Rightarrow the weight increases.

- 1 Termination in the π -calculus
- 2 Weight-based Type Systems
- 3 Allowing Forms of Recursion
- 4 Higher-order Mobile Calculi**

HOPi as a framework for higher order mobility

HOPi

- Messages exchanged are processes.
- $a(X).P_1 \mid \bar{a}\langle Q \rangle.P_2 \rightarrow P_1[Q/X] \mid P_2.$
- Replication is no longer present ...

Another form of recursion

- ... however, divergences arise.
- $Q_0 = a(X).(\bar{a}\langle X \rangle \mid X)$ and $P_0 = Q_0 \mid \bar{a}\langle Q_0 \rangle.$ (P_0 diverges.)

A weight-based type system for higher order π

Typing rule for output

$$\frac{\Gamma \vdash P : k \quad \Gamma \vdash Q : m \quad k < n}{\Gamma \vdash \bar{a}^n \langle P \rangle . Q : \max(m, n)} \quad (\text{S4})$$

Principles

- Forbidding self-emissions: for $\bar{a} \langle P \rangle$ to be typable, $\bar{a} \notin P$.
- Use of levels: for $\bar{a}^n \langle P \rangle$ with $P : m$, we must have $n > m$.
- *Soundness*: typable processes terminate.

Examples

Ruling out P_0

$Q_0 = a(X).(\bar{a}\langle X \rangle \mid X)$ and $P_0 = Q_0 \mid \bar{a}\langle Q_0 \rangle$

- P_0 contains $\bar{a}^n\langle Q_0 \rangle$ and Q_0 contains an output on a^n .
- To type P_0 , we must have $n > n$.

A typable process

$Q_3 = a(X).(X \mid X) \mid \bar{a}\langle \bar{b}\langle \mathbf{0} \rangle \rangle \mid b(Y).Y$

Examples

Ruling out P_0

$Q_0 = a(X).(\bar{a}\langle X \rangle \mid X)$ and $P_0 = Q_0 \mid \bar{a}\langle Q_0 \rangle$

- P_0 contains $\bar{a}^n\langle Q_0 \rangle$ and Q_0 contains an output on a^n .
- To type P_0 , we must have $n > n$.

A typable process

$Q_3 = a^2(X).(X \mid X) \mid \bar{a}^2\langle \bar{b}^1\langle \mathbf{0} \rangle \rangle \mid b^1(Y).Y$
 $[1, 0] \rightarrow [0, 2] \rightarrow [0, 1]$

Relation with S1

- S4 is to HOPi what S1 is to π .
- Polynomial inference: similar to S1 inference.
- Standard encoding of HOPi in π [SangiorgiWalker01]: nearly every typable HOPi process is encoded into a S1-typable process.

Further extensions

Exploring termination in richer higher-order calculi:

- A calculus where messages are functions of any order.
- A calculus where localized computation and passivation are allowed.

Conclusion

Examples of level-based termination

- Lock-freedom (Kobayashi & Sangiorgi).
- Responsiveness (Acciai & Boreale).

Future Works

- Develop extensions of the system for higher-order message passing.
- Analyse other approaches to termination (logical relations).
- Explore the use of polymorphism.
- Study termination in calculi composed of a functional part (termination proved with logical relations) and an imperative part (termination proved with a multiset decreasing). (π with 'functional names', λ +ref (Boudol [Concur07]), λ +objects).

Annex 1

Reduction from 3SAT

- Initial problem: clauses $(C_j)_j$ of 3 literals l_j^1, l_j^2, l_j^3 , propositional variables $V = v_1, \dots, v_p$.
- Solution: mapping $val : V \rightarrow \{\mathbf{True}, \mathbf{False}\}$ s.t. for every clause C_i at least one element of $\{val(l_j^1), val(l_j^2), val(l_j^3)\}$ is **True**.
- Creation of a name τ . Having a level $\geq niv(\tau) = \text{"Is valued to True"}$.
- Creation of two names x_i, x'_i for each prop.var. v_i and adding the module $!x_i.x'_i.\bar{\tau} \mid !\tau.\tau.(\bar{x}_i \mid \bar{x}'_i)$.
- For every clause $C_j = l_j^1, l_j^2, l_j^3$, if $l_j^i = v_k$ we use x_k if $l_j^i = \neg v_k$ we use x'_k . We add the module $!\tau.\tau.(\bar{l}_j^1 \mid \bar{l}_j^2 \mid \bar{l}_j^3) \mid !l_j^1.l_j^2.l_j^3.\bar{\tau}$.
- Equivalence between "the initial problem has a solution" and "the parallel composition of the module is typable".

Annex 2

Code of the *symbol table*

```

!p(a, b, s, e).a(v, r) if s = v then
   $\bar{r}\langle e \rangle \mid \bar{p}\langle a, b, s, e \rangle$ 
else
  if b ≠ nil then
     $\bar{b}\langle v, r \rangle \mid \bar{p}\langle a, b, s, e \rangle$ 
  else
     $(\nu c, f)(\bar{r}\langle f \rangle \mid \bar{p}\langle a, c, s, e \rangle \mid \bar{p}\langle c, nil, v, f \rangle)$ 

```

Initialization

```

 $\bar{p}\langle root, nil, dummy1, dummy2 \rangle$ 

```

Requests

```

 $\overline{root}\langle "Existence", r_1 \rangle \mid \overline{root}\langle "Essence", r_2 \rangle \mid \overline{root}\langle "Existence", r_2 \rangle$ 

```