What Is TyCO, After All?

Final Seminar

Maxime Gamboni

EPFL
Plan

1. What I had to do
2. What I did
3. What is yet to be done
Asynchronous $\pi$-calculus with Nested Variants

What is TyCO?

- $\pi_a^V$ with one-level variants only
- message input and destruction is atomic

Does $\pi_a^V$ have more expressive power than TyCO?
We want a good and fully abstract encoding from $\pi^V_a$ to TyCO and the other way round.

6. Fully Abstract Encoding

$P \sim Q$ if and only if $[P] \sim [Q]$

6. Distributed Encoding

$[P|Q] = [P][Q]$ and $[(\nu a) P] = (\nu a)[P]$
From the original document, I did the following changes:

- Case Reduction Relation (doesn’t take a step)
- Linear Receptiveness
- Undecidability of D-Link
- Definition of Receptive Equivalences
- Made the Nested Encoding Syntax-Directed
- Minor Fixes (Substitution, Operational Correspondence, Full Abstraction . . . )
Case Reduction Semantics

We tried several semantics for handling of $\pi_a^V$'s case reduction:

1. Structural Congruence $\equiv$
   (Breaks Subject Congruence)

2. $\tau$-transition $\rightarrow$
   (Full Abstraction on weak equivalences only)

3. Directional Congruence $\Rightarrow$
   (works :-))
Linear Weakening (Receptiveness)

- The problem:

For a linear, the typability of \((\nu a) P\) requires a to be read and written in \(P\). But:

\[
(\nu a) (a!_k (\nu x).Q \mid a?\{l_j(y_j)=P_j \mid j \in J\}) \xrightarrow{\tau} (\nu a)(\nu x)(Q \mid P_k \{x/y_k\})
\]

In that example type soundness is broken!

- The answer:

Linear Weakening
We had introduced the concept of *Dynamic Links* to avoid extrusion of plain names.

Its definition is recursive using input and bound output:

\[ a \gg b \overset{\text{def}}{=} a?*\{l_j(x) = b!l_j(\nu z).z \gg x \mid j \in J\} \] (uniform case)

I spent a few weeks to prove its (receptive) typability before seeing that it is undecidable (so I made it an axiom)
Minor Changes

- Dynamic links have to work on branching inputs as well.
- The first version of the $\pi^V$ $\rightarrow$ TyCO encoding was type-directed but it could be made syntax directed only.
- $\pi^V_a$-TyCO Full Abstraction could be simplified.
So, does it work, finally?

Short Answer: No.
Long Answer:
It works only on a subset of $\pi^V_a$ processes.

1. The encoding doesn’t work on processes that receive on received names because it breaks uniformity.
   $d?(x).x?(y)$

2. The operational correspondence is broken on processes that do input and free output on a name
   $a!x | a?(y).P$
There is still some work to be done in the area!

- Are TyCO and $\pi^V_a$ equivalent?

“Probably”...
Thank You For Following Me (or attempting to)!

Questions?