π with leftovers: a mechanisation in Agda

Uma Zalakain and Ornela Dardha
University of Glasgow

Motivation

π-calculus?

computational foundation for communication and concurrency

like the λ-calculus, but with communication instead of β-reduction

linear π-calculus?

communication channels must be used exactly once

serves as a target encoding for session types

goal

common underlying framework for type systems for the π-calculus

$$\boxed{\text{session-typed } \pi\text{-calculus}} \longrightarrow \boxed{\text{linear } \pi\text{-calculus}}$$

^this

Constraints

- untyped syntax and semantics
- must support shared channels
- goal: to prove type safety

Overview

- syntax
- semantics
- type system
- type safety
- future work

Notation

TYPES	blue violet, uppercased, indices as subscripts
constructors	burnt orange
functions	olive green
variables	black, in italics

Well-Scoped Syntax

$$\frac{n : \mathbb{N}}{0 : \mathrm{VAR}_{1+n}} \qquad \frac{x : \mathrm{VAR}_n}{1{+}x : \mathrm{VAR}_{1+n}}$$

$$\begin{array}{rll}
\mathrm{PROCESS}_n ::= & \mathbb{0}_n & \text{(inaction)} \\
| & \boldsymbol{\nu}\ \mathrm{PROCESS}_{1+n} & \text{(restriction)} \\
| & \mathrm{PROCESS}_n \parallel \mathrm{PROCESS}_n & \text{(parallel)} \\
| & \mathrm{VAR}_n\ ()\ \mathrm{PROCESS}_{1+n} & \text{(input)} \\
| & \mathrm{VAR}_n\ \langle\ \mathrm{VAR}_n\ \rangle\ \mathrm{PROCESS}_n & \text{(output)}
\end{array}$$

e.g.

$$\begin{array}{llllllll}
(\boldsymbol{\nu}\ x) & (x\ (\ y\ )\ . & y\ \langle\ a\ \rangle\ . & \mathbb{0} \parallel (\boldsymbol{\nu}\ y) & (x\ \langle\ y\ \rangle\ . & y\ (\ z\ )\ . & \mathbb{0})) \\
\boldsymbol{\nu} & (0\ () & 0\ \langle\ a\ \rangle & \mathbb{0} \parallel \boldsymbol{\nu} & (1\ \langle\ 0\ \rangle & 0\ () & \mathbb{0}))
\end{array}$$

Structural Congruence

$$\frac{}{\text{comp-assoc} : P \parallel (Q \parallel R) \cong (P \parallel Q) \parallel R}$$

$$\frac{}{\text{comp-sym} : P \parallel Q \cong Q \parallel P} \qquad \frac{}{\text{comp-end} : P \parallel \mathbb{0}_n \cong P}$$

$$\frac{}{\text{scope-end} : \boldsymbol{\nu}\ \mathbb{0}_{1+n} \cong \mathbb{0}_n}$$

$$\frac{uQ : \mathsf{UNUSED}_0\ Q}{\text{scope-ext} : \boldsymbol{\nu}\ (P \parallel Q) \cong (\boldsymbol{\nu}\ P) \parallel \mathsf{lower}_0\ \ Q\ uQ}$$

$$\frac{}{\text{scope-comm} : \boldsymbol{\nu}\ \boldsymbol{\nu}\ P \cong \boldsymbol{\nu}\ \boldsymbol{\nu}\ \mathsf{swap}_0\ P}$$

$\simeq$ is the congruent equivalence closure of $\cong$

Reduction Relation

$$\frac{}{\text{internal} : \text{CHANNEL}_n} \qquad \frac{i : \text{VAR}_n}{\text{external } i : \text{CHANNEL}_n}$$

$$\frac{i\ j : \text{VAR}_n \qquad P : \text{PROCESS}_{1+n} \qquad Q : \text{PROCESS}_n}{\text{comm} : i\ ()\ P \parallel i\ \langle\ j\ \rangle\ Q \longrightarrow_{\text{external } i} \text{lower}_0\ (P\ [\ 0 \mapsto 1{+}j\ ])\ uP' \parallel Q}$$

$$\frac{red : P \longrightarrow_c P'}{\text{par } red : P \parallel Q \longrightarrow_c P' \parallel Q} \qquad \frac{red : P \longrightarrow_c Q}{\text{res } red : \boldsymbol{\nu}\ P \longrightarrow_{\text{dec } c} \boldsymbol{\nu}\ Q}$$

$$\frac{eq : P \simeq P' \qquad red : P' \longrightarrow_c Q}{\text{struct } eq\ red : P \longrightarrow_c Q}$$

untyped syntax

operational semantics

type system

Type System

- syntax directed
- independent type and usage contexts
- extra leftover usage context
- intrinsic context splits
- parametrised by a set of usage algebras

Usage Algebra

- partial
- neutral element $0\cdot$
- minimal element $0\cdot$
- associative
- commutative
- deterministic
- cancellative
- decidable

$$
\begin{aligned}
0\cdot &: C \\
1\cdot &: C \\
\_ \coloneqq \_ \cdot \_ &: C \to C \to C \to \mathsf{SET}
\end{aligned}
$$

	carrier	operation
linear	$0 : \mathsf{Lin}$ $1 : \mathsf{Lin}$	$0 \coloneqq 0 \cdot 0$ $1 \coloneqq 1 \cdot 0$ $1 \coloneqq 0 \cdot 1$
graded	$0 : \mathsf{Gra}$ $1{+} : \mathsf{Gra} \to \mathsf{Gra}$	$\forall\, x\, y\, z$ $\to x \equiv y + z$ $\to x \coloneqq y \cdot z$
shared	$\omega : \mathsf{Sha}$	$\omega \coloneqq \omega \cdot \omega$

Indexed Usage Algebras

$$
\begin{aligned}
\mathsf{IDX} &: \mathsf{SET} \\
\exists\mathsf{IDX} &: \mathsf{IDX} \\
\mathsf{USAGE} &: \mathsf{IDX} \to \mathsf{SET} \\
\mathsf{ALGEBRAS} &: (idx : \mathsf{IDX}) \to \mathsf{ALGEBRA}_{\mathsf{USAGE}_{idx}}
\end{aligned}
$$

Capability Notation

capability input / output **multiplicity** $0\cdot$, $1\cdot$, ...

$$
\begin{aligned}
C^2 &= C \times C \\
\ell_{\emptyset} &= 0\cdot \;,\; 0\cdot \\
\ell_{\mathsf{i}} &= 1\cdot \;,\; 0\cdot \\
\ell_{\mathsf{o}} &= 0\cdot \;,\; 1\cdot \\
\ell_{\#} &= 1\cdot \;,\; 1\cdot \\
(x_l \;,\; x_r) \coloneqq (y_l \;,\; y_r) \cdot^2 (z_l \;,\; z_r) &= (x_l \coloneqq y_l \cdot z_l) \times (x_r \coloneqq y_r \cdot z_r)
\end{aligned}
$$

Types

$$\frac{}{\mathbb{1} : \mathsf{TYPE}} \qquad \frac{n : \mathbb{N}}{\mathsf{B}[\ n\ ] : \mathsf{TYPE}} \qquad \frac{t : \mathsf{TYPE} \qquad \begin{array}{l} idx : \mathsf{IDX} \\ x : \mathsf{USAGE}^{2}_{idx} \end{array}}{\mathsf{C}[\ t\ ;\ x\ ] : \mathsf{TYPE}}$$

e.g. $\mathsf{C}[\ \mathsf{C}[\ \mathbb{1}\ ;\ \omega\ ]\ ;\ \ell_{\mathsf{i}}\ ]$

Typing Relation

$$\begin{array}{rl} \text{PRECTX}_n & \text{list of TYPE s of length } n \\ \text{IDXS}_n & \text{list of IDX s of length } n \\ \text{CTX}_{idxs} & \text{list of USAGE}^2\text{s indexed over } idxs\ : \text{IDXS}_n \end{array}$$

$$\frac{\gamma : \text{PRECTX}_n \qquad \begin{array}{c} idxs : \text{IDXS}_n \\ \Gamma : \text{CTX}_{idxs} \end{array} \qquad P : \text{PROCESS}_n \qquad \Delta : \text{CTX}_{idxs}}{\gamma \,;\, \Gamma \vdash P \triangleright \Delta : \text{SET}}$$

$$\frac{\gamma : \text{PRECTX}_n \quad \begin{array}{c} idxs : \text{IDXS}_n \\ \Gamma : \text{CTX}_{idxs} \end{array} \quad i : \text{VAR}_n \quad t : \text{TYPE} \quad \begin{array}{c} idx : \text{IDX} \\ y : \text{USAGE}^2_{idx} \end{array} \quad \Delta : \text{CTX}_{idxs}}{\gamma \,;\, \Gamma \ni_i t \,;\, y \triangleright \Delta : \text{SET}}$$

Variable Typing Rules

$$\frac{x \coloneqq y \cdot^2 z}{0 : \gamma \,,\, t \,;\, \Gamma \,,\, x \ni_0 t \,;\, y \triangleright \Gamma \,,\, z}$$

$$\frac{loc_i : \gamma \qquad ; \Gamma \qquad \ni \quad {}_i\ t \,;\, x \triangleright \Delta}{1+\ loc_i : \gamma \,,\, t' \,;\, \Gamma \,,\, x' \ni_{1+i} t \,;\, x \triangleright \Delta \,,\, x'}$$

Typing Rules

$$\frac{}{\mathsf{end} : \gamma \,;\, \Gamma \vdash \mathbb{0} \triangleright \Gamma}
\qquad
\frac{l : \gamma \,;\, \Gamma \vdash P \triangleright \Delta \qquad r : \gamma \,;\, \Delta \vdash Q \triangleright \Xi}{\mathsf{comp}\; l\; r : \gamma \,;\, \Gamma \vdash P \parallel Q \triangleright \Xi}$$

Typing Rules

$$\frac{t : \mathsf{TYPE} \qquad x : \mathsf{USAGE}^{2}_{idx} \qquad y : \mathsf{USAGE}_{idx'} \qquad cont : \gamma \,,\, \mathsf{C}[\, t \,;\, x \,] \,;\, \Gamma \,,\, (y \,,\, y) \vdash P \triangleright \Delta \,,\, \ell_{\varnothing}}{\mathsf{chan}\ t\ x\ y\ cont : \gamma \,;\, \Gamma \vdash \boldsymbol{\nu}\ P \triangleright \Delta}$$

$$\frac{chan_i : \gamma \,;\, \Gamma \ni_i \mathsf{C}[\, t \,;\, x \,] \,;\, \ell_{\mathsf{i}} \triangleright \Xi \qquad cont : \gamma \,,\, t \,;\, \Xi \,,\, x \vdash P \triangleright \Theta \,,\, \ell_{\varnothing}}{\mathsf{recv}\ chan_i\ cont : \gamma \,;\, \Gamma \vdash i\ ()\ P \triangleright \Theta} \qquad \frac{\ldots}{\mathsf{send}\ \ldots}$$

Example Derivation

```
p  :  PROCESS₁₊zero
p  =  ν
       (0 () (0 () 𝟘) ∥
       ν (1+ 0 ⟨ 0 ⟩ (0 ⟨ 1+ 1+ 0 ⟩ ) 𝟘))

_  :  [] , 𝟙 ; [] , ω ⊢ p ▹ ε
_  =  chan C[ 𝟙 ; ω ] ℓᵢ 1· (comp
       (recv 0 (recv 0 end))
       (chan 𝟙 ω 1· (send (1+ 0) 0 (send 0 (1+ 1+ 0) end))))
```

Before Leftover Typing

- using functions to update usage contexts

$$\frac{\dots \qquad (\mathsf{update}_i\ \dots\ \Gamma) \vdash P \qquad \dots}{\Gamma \vdash i\ ()\ P}$$

- extrinsic context splits

$$\frac{\Gamma \coloneqq \Delta \otimes \Xi \qquad \Delta \vdash P \qquad \Xi \vdash Q}{\Gamma \vdash P \parallel Q}$$

type system

$\boxed{\text{type safety}}$

Subject Congruence

- **framing** the only resources the well-typedness of a process depends on are the ones used by it.
- **weakening** inserting a new variable into the context preserves the well-typedness of a process as long as the usage annotation of the inserted variable is preserved as a leftover.
- **strengthening** removing an unused variable preserves the well-typedness of a process.
- **subject congruence** applying structural congruence rules to a well typed process preserves its well-typedness.

Subject Reduction

- **renaming**
  substituting a variable i for a variable j in a well-typed process P results in a well-typed process as long as the leftovers at index j after P are at least the consumption made by P at index i.

- **subject reduction**
  let $\gamma\ ;\ \Gamma \vdash P \triangleright \Xi$ and $P \longrightarrow_c Q$,
  - if c is internal, then $\gamma\ ;\ \Gamma \vdash Q \triangleright \Xi$.
  - if c is external i and $\Gamma \ni_i \ell_\# \triangleright \Delta$, then $\gamma\ ;\ \Delta \vdash Q \triangleright \Xi$.

Future Work

- prove preservation of well-balancedness
- product types, sum types, recursion
- decidable type checking

thank you!

^_^

questions?

Type Safety: Renaming

Renaming with accumulator

Let $\gamma \,;\, \Gamma_i \vdash P \triangleright \Psi_i$. Let there be some Γ, Ψ, Γ_j and Ψ_j such that:

- $\gamma \,;\, \Gamma_i \ni_i t \,;\, m \triangleright \Gamma$
- $\gamma \,;\, \Gamma_j \ni_j t \,;\, m \triangleright \Gamma$
- $\gamma \,;\, \Psi_i \ni_i t \,;\, n \triangleright \Psi$
- $\gamma \,;\, \Psi_j \ni_j t \,;\, n \triangleright \Psi$

$$
\begin{tikzcd}
\Gamma_i \arrow[r, "\vdash L"] \arrow[d, "\ni_i m"'] & \Xi_i \arrow[r, "\vdash R"] \arrow[d, dotted, "\ni_i l"'] & \Psi_i \arrow[d, "\ni_i n"] \\
\Gamma \arrow[r, dotted] \arrow[rr, bend right, "\begin{array}{c}\Delta \\ \Delta_i = \ell_{\varnothing}\end{array}"'] & \Theta \arrow[r, dotted] & \Psi \\
\Gamma_j \arrow[u, "\ni_j m"] \arrow[r, dotted, "\vdash L\,[\,i \mapsto j\,]"'] & \Xi_j \arrow[u, dotted, "\ni_j l"] \arrow[r, dotted, "\vdash R\,[\,i \mapsto j\,]"'] & \Psi_j \arrow[u, "\ni_j n"']
\end{tikzcd}
$$

Let there be some Δ such that $\Gamma \coloneqq \Delta \otimes \Psi$. Let Δ have usage $\ell_{\varnothing}$ at position i. Then $\gamma \,;\, \Gamma_j \vdash P\,[\,i \mapsto j\,] \triangleright \Psi_j$.