Typing migration-control in $\text{I}_{\text{sd}}^\pi$

Francisco Martins

DI/FCUL

Joint work with

António Ravara
Lexically Scoped Distributed $\pi$

- Extends $\pi$, distributing processes over networks of named sites where they compute.
- Processes allowed to:
  - communicate via channels (as in $\pi$), but only locally
  - migrate from site to site.
Main concepts

In $lsd\pi$

- channels are
  - resources associated *uniquely* to sites
  - located at *creation* time
    - In $s[(\nu c) \ldots]$, channel $c$ is created *locally*, at $s$;
    - In $r[(\nu c@s) \ldots]$, channel $c$ is created *remotely*, to be located at $s$;
Main concepts

In $lsd\pi$

- channels are
  - resources associated *uniquely* to sites
  - located at *creation* time
    - In $s[(\nu\ c) \ldots]$, channel $c$ is created *locally*, at $s$;
    - In $r[(\nu\ c@s) \ldots]$, channel $c$ is created *remotely*, to be located at $s$;

- sites are
  - *collections* of channels;
  - *shells* of local computations.
Typing migration-control in lsd\(\pi\) – p.4

\[ r[a@s!\langle b\rangle] \parallel s[a?(x) P] \]
Typing migration-control in lsdπ – p.4
Typing migration-control in lsdπ

\[ r[s ! a] \parallel s[a ? (x) P] \]

\[ r[0] \parallel s[(a @ s ! b) \sigma_{rs} \mid a ? (x) P] \]

\[ r[0] \parallel s[a ! b @ r \mid a ? (x) P] \]
Typing migration-control in lsd\(\pi\) – p.4
Lexically Scoping Distribution:

what you see is what you get!
Lexically Scoping Distribution:

what you see is what you get!

a flavour ...

\[
(\nu a@s) \ r_{G_1}[a@s!\langle b\rangle] \parallel s_{G_2}[a?(x:S) \ P]
\]

\[\therefore\] simple names are local;
remote names are explicitly located;
Ensuring security policies: sites allowed to

- send messages (remote communicating)
- migrate processes
- create remote names
Processes

\[ P, Q ::= 0 \mid u!\langle v \rangle \mid u?(x : S) \ P \mid P \mid Q \mid (\nu u) \ P \]

simple channels \( a, b, c, x, y \)
channels \( u, v \) \( ::= a \mid a@s \)
sites \( r, s, t \)
set of sites \( R, S \)
Syntax ($\text{lsd}^\pi$)

Processes

\[ P, Q ::= 0 \mid u!\langle v \rangle \mid u?(x : S)\ P \mid P \parallel Q \mid (\nu\ u)\ P \]

Networks

\[ N, M ::= 0 \mid s_G[P] \mid N \parallel M \mid (\nu\ a@s)\ N \]

simple channels \ a, b, c, x, y \ channels \ u, v ::= a \mid a@s \]

sites \ r, s, t \ set of sites \ R, S \]

G? stay tuned ...
Syntax (types)

\[ \Gamma ::= \{ s_1 : (\varphi_1, G_1), \ldots, s_n : (\varphi_n, G_n) \} \quad \text{typings} \]

\[ \varphi ::= \{ a_1 : \gamma_1, \ldots, a_n : \gamma_n \} \quad \text{site types} \]

\[ G ::= \{ \text{rem} : S_1, \text{mig} : S_2, \text{new} : S_3 \} \quad \text{site policies} \]
Syntax (types)

\[
\Gamma ::= \{s_1 : (\varphi_1, G_1), \ldots, s_n : (\varphi_n, G_n)\} \quad \text{typings}
\]

\[
\varphi ::= \{a_1 : \gamma_1, \ldots, a_n : \gamma_n\} \quad \text{site types}
\]

\[
G ::= \{\text{rem} : S_1, \text{mig} : S_2, \text{new} : S_3\} \quad \text{site policies}
\]

\[
\gamma ::= \text{ch}(\gamma)@S^t | \text{val} \quad \text{channel types}
\]

\[
t ::= o | i | b \quad \text{site tags}
\]
Free names

$$ (\nu \ a) \ (\ldots a \ldots a@s \ldots) $$
Free names

\[(\nu a) \left( \ldots a \ldots a@s \ldots \right)\]

\[(\nu a@s) \left( \ldots a \ldots a@s \ldots \right)\]
Free names

\[
\nu \ a \ (\ldots \ a \ldots \ a@s \ldots )
\]

\[
\nu \ a@s \ (\ldots \ a \ldots \ a@s \ldots )
\]

\[
\nu \ a@r \ (\ldots \ a \ldots \ a@r \ldots )
\]
Structural congruence (some rules)

\[ s_G[P | Q] \equiv s_G[P] \parallel s_G[Q] \]
**Structural congruence (some rules)**

\[
s_G[P | Q] \equiv s_G[P] \parallel s_G[Q]
\]

\[
(\nu \ a@r) \ s_G[P] \equiv s_G[(\nu \ a@r) \ P] \quad r \neq s
\]
Structural congruence (some rules)

\[ s_G[P \mid Q] \equiv s_G[P] \parallel s_G[Q] \]

\[ (\nu a@r) \ s_G[P] \equiv s_G[(\nu a@r) \ P] \quad r \neq s \]

\[ (\nu a@s) \ s_G[P] \equiv s_G[(\nu a@s) \ P] \quad a \notin \text{fn}(P) \]

Example:

\[ (\nu a@s) \ s_G[a!\langle b\rangle \mid a@s!\langle c\rangle] \not\equiv s_G[(\nu a@s) \ a!\langle b\rangle \mid a@s!\langle c\rangle] \]
**Structural congruence (some rules)**

\[ s_G[P | Q] \equiv s_G[P] \parallel s_G[Q] \]

\[ (\nu a@r) \ s_G[P] \equiv s_G[(\nu a@r) \ P] \quad r \neq s \]

\[ (\nu a@s) \ s_G[P] \equiv s_G[(\nu a@s) \ P] \quad a \notin \text{fn}(P) \]

\[ (\nu a@s) \ s_G[P] \equiv s_G[(\nu a) \ P] \quad a@s \notin \text{fn}(P) \]

Example:

\[ (\nu a@s) \ s_G[a!\langle b\rangle | a@s!\langle c\rangle] \not\equiv s_G[(\nu a) a!\langle b\rangle | a@s!\langle c\rangle] \]
Reduction rules (some rules)

\[ s_G[a ! \langle v \rangle | a ?(x : S) P] \rightarrow P[v/x] \]
Reduction rules (some rules)

\[
\begin{align*}
  s_G[a!\langle v \rangle \mid a?(x : S) \ P] & \rightarrow P[v/x] \\
  s_{G_1}[P] \parallel r_{G_2}[a@s?(x : S) \ Q] & \rightarrow s_{G_1}[P \mid (a@s?(x : S) \ Q)\sigma_{rs}] \parallel r_{G_2}[0], \quad r \neq s
\end{align*}
\]

\(\sigma_{rs}\) translates free names from \(r\) to \(s\):

\[
\begin{align*}
  \sigma_{rs}(s) &= s & \sigma_{rs}(a@s) &= a \\
  \sigma_{rs}(a) &= a@r & \sigma_{rs}(a@t) &= a@t, & t \not\in \{r, s\}
\end{align*}
\]
Security policies violation

Let $r \neq t$

- Remote communication

$$s\{\text{rem:}\{t\}\}[P] \parallel r_{G_1}[a@s!\langle x\rangle]$$
Let $r \neq t$

- Remote communication

\[
s_{\text{rem}:\{t\}}[P] \parallel r_{G_1}[a@s!\langle x \rangle] \parallel s_{\text{rem}:\{t\}}[b@r?(x:S)\ a@s!\langle x \rangle] \parallel r_{\text{mig}:\{s\}}[0]
\]
Let $r \neq t$

- Remote communication

\[
s\{\text{rem:}\{t\}\}[P] \parallel r_{G_1}[a\@s ! \langle x \rangle]
\]

\[
s\{\text{rem:}\{t\}\}[b\@r?(x : S) \ a\@s ! \langle x \rangle] \parallel r_{\{\text{mig:}\{s\}\}}[0]
\]

\[
s\{\text{rem:}\{r\}\}[a?(x : \{t\} \ 0] \parallel r_{G_1}[a\@s ! \langle b\@r \rangle]
\]
Security policies violation

Let $r \not= t$

- Remote communication
- Migration

$$s_{\{\text{mig:}\{t\}\}}[P] \parallel r_{G_1}[a@s?(x:S)Q]$$
Let $r \neq t$

- Remote communication
- Migration

\[ s_{\{ \text{mig:}\{t\}\}}[P] \parallel r_{G_1}[a@s?(x : S) \ Q] \]

\[ s_{\{ \text{rem:}\{r\}\}}[a?(x : \{r, t\}) \ x ! \langle c \rangle] \parallel r_{\emptyset}[a@s ! \langle b \rangle] \]
Security policies violation

Let $r \neq t$

- Remote communication
- Migration
- Name creation

$s\{\text{new:}\{t\}\} [P] \parallel r_{G_1} [(\nu \ a@s) \ Q]$
Security policies violation

Let \( r \neq t \)

- Remote communication
- Migration
- Name creation

\[
\begin{align*}
&\quad s_{\{\text{new}:\{t\}\}}[P] \parallel r_{G_1}[\nu a@s) Q] \\
&\quad (\nu a@s) s_{\{\text{new}:\{r\}\}}[P] \parallel r_{G_1}[a@s ! \langle b \rangle] \parallel t_{G_2}[a@s ! \langle c \rangle]
\end{align*}
\]
Subtyping relation

\[ b \leq i \quad b \leq o \]

\[
\begin{align*}
R \subseteq S \quad & \quad S^o \leq R^o \\
S \subseteq R \quad & \quad S^i \leq R^i \\
t \leq t' \quad & \quad S^t \leq S^{t'}
\end{align*}
\]

\[
\gamma \leq \gamma \\
\gamma_1 \leq \gamma_2 \quad \gamma_2 \leq \gamma_3 \\
\gamma_1 \leq \gamma_3 \\
\gamma_1 \leq \gamma_2 \\
\gamma_1 \leq \gamma_2 \\
S^t \leq R^{t'} \\
\text{ch}(\gamma_1)@S^t \leq \text{ch}(\gamma_2)@R^{t'}
\]
Subtyping relation

\[ \begin{align*}
R \subseteq S & \quad \rightarrow \quad S^o \leq R^o \\
S \subseteq R & \quad \rightarrow \quad S^i \leq R^i \\
t \leq t' & \quad \rightarrow \quad S^t \leq S^{t'}
\end{align*} \]

\[ \begin{align*}
\gamma \leq \gamma & \quad \rightarrow \quad \gamma_1 \leq \gamma_2 \quad \rightarrow \quad \gamma_2 \leq \gamma_3 \\
& \quad \rightarrow \quad \gamma_1 \leq \gamma_3
\end{align*} \]

\[ \begin{align*}
\gamma_1 \leq \gamma_2 & \quad \rightarrow \quad S^t \leq R^{t'} \\
\text{ch}(\gamma_1)@S^t & \leq \text{ch}(\gamma_2)@R^{t'}
\end{align*} \]

\[ \therefore \quad \text{Outputs may grow} \]

\[ \text{Inputs may shrink} \]
Typing names

\[ \Gamma \vdash_s n : \gamma \]

\(\triangleright\) if \(n\) is a simple name, then it belongs to \(s\)
Judgments

- **Typing names**

  \[ \Gamma \vdash_s n : \gamma \]

  ▲ if \( n \) is a simple name, then it belongs to \( s \)

- **Typing processes**

  \[ \Gamma \vdash_{s,S} P \]

  ▲ simple names of \( P \) considered of \( s \)
  ▲ at runtime, \( P \) might be in any site of \( S \)

- **Example**

  \[ s_G[a?(x : \{r, t\}) \ x?(...)] P \]_s = \{r,t\}
Judgments

- **Typing names**
  \[ \Gamma \vdash_s n : \gamma \]
  - if \( n \) is a simple name, then it belongs to \( s \)

- **Typing processes**
  \[ \Gamma \vdash_{s,S} P \]
  - simple names of \( P \) considered of \( s \)
  - at runtime, \( P \) might be in any site of \( S \)

- **Typing networks**
  \[ \Gamma \vdash N \]
Typing located outputs

Rule

\[ \Gamma \vdash_s v : \gamma_2 \]

\[ \gamma_2 \leq \gamma_1 \]

\[ \Gamma(r)_1(a) = \text{ch}(\gamma_1)@\{r\}^b \]

\[ S \subseteq \Gamma(r)_2(\text{rem}) \]

\[ \Gamma \vdash_{s,S} a@r!\langle v \rangle \]
Typing located outputs

- Rule

\[
\begin{align*}
\Gamma \vdash_s v : \gamma_2 \\
\gamma_2 &\leq \gamma_1 \\
\Gamma(r)_1(a) &= \text{ch}(\gamma_1)@\{r\}^b \\
S &\subseteq \Gamma(r)_2(\text{rem}) \\
\hline
\Gamma \vdash_{s,S} a@r ! \langle v \rangle
\end{align*}
\]

- Example

\[
\begin{align*}
\Gamma \vdash_s v : \text{ch}(\gamma)@\{s\}^b \\
\text{ch}(\gamma)@\{s\}^b &\leq \text{ch}(\gamma)@\{s, r\}^i \\
\Gamma(r)_1(a) &= \text{ch}(\text{ch}(\gamma)@\{s, r\}^i)@\{r\}^b \\
\{t\} &\subseteq \{s, t\} \\
\hline
\Gamma \vdash_{s,\{t\}} a@r ! \langle v \rangle
\end{align*}
\]
Typing located outputs

Rule

\[ \Gamma \vdash_s v : \gamma_2 \]
\[ \gamma_2 \leq \gamma_1 \]
\[ \Gamma(r)_1(a) = \text{ch}(\gamma_1)@\{r\}^b \]
\[ S \subseteq \Gamma(r)_2(\text{rem}) \]

\[ \Gamma \vdash_{s,S} a@r ! \langle v \rangle \]

Example

\[ \Gamma \vdash_s v : \text{ch}(\gamma)@\{s\}^b \]
\[ \text{ch}(\gamma)@\{s\}^b \leq \text{ch}(\gamma)@\{s, r\}^i \]
\[ \Gamma(r)_1(a) = \text{ch}(\text{ch}(\gamma)@\{s, r\}^i)@\{r\}^b \]
\[ \{t\} \subseteq \{s, t\} \]

\[ \Gamma \vdash_{s,\{t\}} a@r ! \langle v \rangle \]
Typing located outputs

- Rule

\[ \Gamma \vdash_s v : \gamma_2 \]
\[ \gamma_2 \leq \gamma_1 \]
\[ \Gamma(r)_1(a) = ch(\gamma_1)@\{r\}^b \]
\[ S \subseteq \Gamma(r)_2(rem) \]
\[ \Gamma \vdash_{s,S} a@r!\langle v \rangle \]

- Example

\[ \Gamma \vdash_s v : ch(\gamma)@\{s\}^b \]
\[ ch(\gamma)@\{s\}^b \leq ch(\gamma)@\{s, r\}^i \]
\[ \Gamma(r)_1(a) = ch(ch(\gamma)@\{s, r\}^i)@\{r\}^b \]
\[ \{t\} \subseteq \{s, t\} \]
\[ \Gamma \vdash_{s,\{t\}} a@r!\langle v \rangle \]
Typing located outputs

Rule

\[ \Gamma \vdash_s v : \gamma_2 \]
\[ \gamma_2 \leq \gamma_1 \]
\[ \Gamma(r)_1(a) = \text{ch}(\gamma_1)@\{r\}^b \]
\[ S \subseteq \Gamma(r)_2(\text{rem}) \]
\[ \Gamma \vdash_{s,S} a@r ! \langle v \rangle \]

Example

\[ \Gamma \vdash_s v : \text{ch}(\gamma)@\{s\}^b \]
\[ \text{ch}(\gamma)@\{s\}^b \leq \text{ch}(\gamma)@\{s, r\}^i \]
\[ \Gamma(r)_1(a) = \text{ch}\left(\text{ch}(\gamma)@\{s, r\}^i\right)@\{r\}^b \]
\[ \{t\} \subseteq \{s, t\} \]
\[ \Gamma \vdash_{s,\{t\}} a@r ! \langle v \rangle \]
Rule

\[ \Gamma \vdash_{s,\{r\}} P \]

\[ \Gamma(r)_1(a) = \text{ch}(\gamma_1)@\{r\}^b \]

\[ \Gamma(s)_1(x) = \text{ch}(\gamma_2)@R^b \]

\[ \text{ch}(\gamma_2)@R^b \leq \gamma_1 \]

\[ S \subseteq \Gamma(r)_2(\text{mig}) \]

\[ \Gamma \setminus x@s \vdash_{s,S} a@r?(x : R) \quad P \]
Typing located inputs

- **Rule**

\[
\begin{align*}
\Gamma \vdash_s, \{r\} P \\
\Gamma(r)_1(a) &= \text{ch}(\gamma_1)@\{r\}^b \\
\Gamma(s)_1(x) &= \text{ch}(\gamma_2)@R^b \\
\text{ch}(\gamma_2)@R^b &\leq \gamma_1 \\
S &\subseteq \Gamma(r)_2(\text{mig}) \\
\Gamma \setminus x@s &\vdash_s, S a@r?(x : R) P
\end{align*}
\]

- **Example**

\[
s_0[a@r?(x : \{t\}) x !\langle c\rangle] \parallel r_{\text{mig} : \{s\}}[a !\langle b@t\rangle] \parallel t_{\text{rem} : \{r\}}[0]
\]
Typing located names (nets)

Rule

\[ \Gamma \vdash N \]

\[ S \setminus s \subseteq \Gamma(s)_2(\text{new}) \]

\( S \) is the set of sites where \( a@s \) occurs free in \( N \)

\[ \Gamma \setminus a@s \vdash (\nu a@s) N \]
Typing located names (nets)

Rule

\[ \Gamma \vdash N \]
\[ S \setminus s \subseteq \Gamma(s)_2(\text{new}) \]
\[ S \text{ is the set of sites where } a@s \text{ occurs free in } N \]
\[ \Gamma \setminus a@s \vdash (\nu a@s) N \]

Example

\[ (\nu a@s) s_{\{\text{new:\{}r\text{,rem:\{}r\text{}}\}}[0] \parallel r@0[a@s!\langle b\rangle] \]
Typing located names (nets)

Rule

\[
\Gamma \vdash N \\
S \setminus s \subseteq \Gamma(s)_2(\text{new}) \\
S \text{ is the set of sites where } a@s \text{ occurs free in } N \\
\Gamma \setminus a@s \vdash (\nu a@s) \quad N
\]

Example

\[
(\nu a@s) \quad s\{\text{new:}\{r\}, \text{rem:}\{r\}\}[0] \parallel r\emptyset\{a@s ! \langle b \rangle\}
\]

\[
S = \{r\} \\
\Gamma(s) = \{((\emptyset, \{\text{new : } \{r\}, \text{rem : } \{r\}\})\})\} \\
\Gamma(r) = \{(b : \text{ch}(\gamma)@\{r\}^b, \emptyset)\}
\]
\[ E = \{ N \mid N \rightarrow^* \nu X (M' \parallel M) \} \]

and \( M \) of the form

- Remote communication

\[ r_{G_1}[P] \parallel s_{G_2}[a@r!\langle v\rangle], \quad s \notin G_1(\text{rem}) \]
\[ s_G[a!\langle b@r\rangle \mid a?(x:S) P], \quad r \notin S \]
\[ \mathcal{E} = \{ N \mid N \xrightarrow{*} \nu \tilde{X} (M' \parallel M) \} \]

and \( M \) of the form

- **Remote communication**

\[
\begin{align*}
    r_{G_1}[P] &\parallel s_{G_2}[a \mathbin{@} r \! \langle v \rangle], & s \notin G_1(\text{rem}) \\
    s_G[a \! \langle b \mathbin{@} r \rangle \mid a\? (x : S) P], & r \notin S
\end{align*}
\]

- **Migration**

\[
\begin{align*}
    r_{G_1}[P] &\parallel s_{G_2}[a \mathbin{@} r \? (x : S) P], & s \notin G_1(\text{mig})
\end{align*}
\]
Runtime errors

\[ \mathcal{E} = \{ N | N \rightarrow^* v \vec{X}(M' \parallel M) \} \]

and \( M \) of the form

- Remote communication
  \[
  r_{G_1}[P] \parallel s_{G_2}[a@r!\langle v \rangle], \quad s \notin G_1(\text{rem})
  
  s_G[a!\langle b@r \rangle | a?(x : S) P], \quad r \notin S
  
  \]

- Migration
  \[
  r_{G_1}[P] \parallel s_{G_2}[a@r?(x : S) P], \quad s \notin G_1(\text{mig})
  
  \]

- Name creation
  \[
  r_{G_1}[P] \parallel s_{G_2}[(\nu a@r) \ P], \quad s \notin G_1(\text{new})
  
  \]
The usual properties

- Subject reduction
  \[ \text{if } \Gamma \vdash N \land N \rightarrow M, \text{ then } \Gamma \vdash M \]

- Well-typed networks free of runtime errors
  \[ \text{if } \Gamma \vdash N \land N \rightarrow^* M, \text{ then } M \notin \mathcal{E} \]
we propose a type system to control:

- remote communication
- process migration
- name creation
Conclusions and further work

- we propose a type system to control:
  - remote communication
  - process migration
  - name creation

- But,
  - \( \text{In } u? (x : S) \ P, \text{ type } S \text{ is fixed.} \)
  - Sites are constants (used explicitly in types!)
Conclusions and further work

we propose a type system to control:
  △ remote communication
  △ process migration
  △ name creation

But,
  △ In \( u?(x : S) P \), type \( S \) is fixed.
  △ Sites are constants (used explicitly in types!)

Further work
  △ Solve the above limitations :))
  △ Specify security policies at channel level
  △ Adjust security policies dynamically