1 Overview

Interaction Nets
   IN Example: List Processing
   Linearity: δ and ε Nodes
   IN Example: Unary Arithmetics
   Application: SK Combinators
Non-determinism in IN
   IN with Multiple Reduction Rules (INMR)
   IN with Multiple Principal Ports (INMPP)
      INMPP Example: Queue Merger
   IN with Multiple Ports (INMP)
      INMP Example: Variable (Reference)
   IN with Multiple Connections (INMC)
      INMC Example: Process Graphs
Inter-representation of Non-Deterministic Models

INMPP as INMC: Port Diamonds
INMR as INMP: Self-Commitment
INMPP as INMP: Marker Nodes
INMC as INMP: Explicit Connectors

Representing the $\pi$-calculus in MIN=INMP+INMPP

The $\pi$-calculus

Reduction Rules
MIN$_\pi$ Nodes, the Translation
MIN$_\pi$ Rules
Send/Receive
Input/Output
Blocking and Unblocking
Completeness and Soundness

2 Interaction Nets

Lafont (1990), inspired by Proof Nets of Linear Logic.
Principal ports, linearity, binary local interaction, preserves interface.
2.1 IN Example: List Processing

List is represented as a chain of cons nodes

Rules for the append operation

Example: append lists (1) and (2)
2.2 Linearity: δ and ε Nodes

(Here z is any binary node.)
To duplicate a net, use δ.
To erase a net, use ε.

2.2.1 IN Example: Unary Arithmetics

The number 5 in unary notation

Arithmetic rules, corresponding to

\[ sx + y = s(x + y) \quad 0 + y = y \quad sx \times y = x \times y + y \quad 0 \times y = 0. \]
2.3 Application: SK Combinators

SK Combinators: \( S_{x y z} = x z (y z) \), \( K_{x y} = x \)
Make application explicit: \( \oplus(\oplus(\ominus(s, X), Y), Z) = \oplus(\ominus(X, Z), \ominus(Y, Z)) \)

\[ \begin{array}{c}
S \quad \ominus \quad X \quad X \\
\oplus \quad S_1 \quad \ominus \quad T \quad T \\
\ominus \quad S_2 \quad \ominus \quad T \quad T \\
\ominus \quad \ominus \quad \ominus \quad \ominus \quad T \\
\end{array} \]

\[ \begin{array}{c}
X \quad Y \quad X \\
\ominus \quad S_1 \quad \ominus \quad Y \quad Y \\
\ominus \quad S_2 \quad \ominus \quad Z \\
\ominus \quad \ominus \quad \ominus \quad \ominus \\
\end{array} \]

\[ \begin{array}{c}
k \quad X \\
\ominus \quad k_1 \quad \ominus \quad X \\
\ominus \quad \ominus \quad \ominus \quad \ominus \\
\end{array} \]

\[ \begin{array}{c}
X \quad Y \\
\ominus \quad k_1 \quad \ominus \quad Y \\
\ominus \quad \ominus \quad \ominus \quad \ominus \\
\end{array} \]

\[ \begin{array}{c}
\ominus(\ominus(k, X), Y) = X \\
\end{array} \]
3 Non-determinism in IN

Traditional IN are *confluent*, therefore are limited to deterministic, functional programming. To represent agents, objects, processes, non-determinism, we need to break the confluence.

Several possible ways to do that:

- IN with Multiple Reduction Rules (INMR)
- IN with Multiple Principal Ports (INMPP)
- IN with Multiple Ports (INMP)
- IN with Multiple Connections (INMC)

3.1 IN with Multiple Reduction Rules (INMR)

Turns out that we can limit ourselves to reflexive asymmetric rules only:

\[ n \bowtie n \rightarrow \text{net}, \] where “net” is not symmetric.
3.2 IN with Multiple Principal Ports (INMPP)

3.2.1 INMPP Example: Queue Merger

In the diagram, the concept of merging queues is illustrated. The graph shows multiple instances of merge nodes connected to different principal ports, indicating the flow of data or information through these ports in the context of INMPP.
3.3 IN with MultiplePorts (INMP)

We also often need to specify the applicability of rules with arity constraints.

- Multiport with no edges
- Multiport with exactly one edge
- Multiport with at least one edge
- Multiport with unspecified number of edges

A variable is an “object” that handles get and set requests.
Outcome depends on the order of interaction (non-determinism)
3.4 IN with Multiple Connections (INMC)

Allow hyper-edges (edges connecting more than two ports). We denote such with a connector point (bold dot)

3.4.1 INMC Example: Process Graphs
4 Inter-representation of Non-Deterministic Models

Which models can represent which others, and at what price?

- Complexity of translation
- Complexity of reduction
- Atomicity properties
- Commitment properties

4.1 INMP as INMC: Port Diamonds
### 4.2 INMR as INMP: Self-Commitment

```
\[ \begin{array}{c}
    \begin{array}{c}
        a \\
        X \\
        Y \\
    \end{array} \\
    \\
    \begin{array}{c}
        a' \\
        X \\
        Y \\
    \end{array} \\
    \\
    \begin{array}{c}
        S_{C_0} \\
        S_{C_1} \\
        S_{C_{N-1}} \\
    \end{array}
\end{array} \]
```
4.3 INMPP as INMP: Marker Nodes

4.4 INMC as INMP: Explicit Connectors
5 Representing the $\pi$-calculus in MIN=INMP+INMPP

Concurrency = Interaction + Non-determinism

5.1 The $\pi$-calculus

Zero $\mathbf{0}$ is the empty process.

Parallel Composition $\mathbf{P;Q}$

Output Prefix $\mathbf{c!v.P}$ sends value $v$ along channel $c$, then does $P$.

Input Prefix $\mathbf{c?x.P}$ receives value $v$ from channel $c$, then does $P[v/x]$.

Hiding/Restriction $\mathbf{(c)P}$ can’t interact on channel $c$.

5.1.1 $\pi$-calculus Example: Email and Phone

$t?x.\langle x!m, t!e, e?y \rangle$

A person talks on the telephone $t$ to another person and receives an email address $e$. Then s/he sends a message $m$ to that address $e$, which is received by a third person.
5.1.2 Reduction Rules

Comm $a ? x . P , a ! c . Q \rightarrow P [ c / x ] , Q$: analogous to $\beta$-reduction

Par If $P \rightarrow Q$ then $P , R \rightarrow Q , R$

Res If $P \rightarrow Q$ then $(x)P \rightarrow (x)Q$: restriction does not restrict the internal process.

5.2 $\text{MIN}_\pi$ Nodes

Channel, placeholder, blocker, unblocker
5.3 The Translation

\[
\begin{align*}
\text{[Q]} & \quad b \\
\text{[a?x.Q]} & \quad x \\
\text{[P]} & \quad a
\end{align*}
\]

5.3.1 Example: Translation of Email-Phone

\[
\begin{align*}
t?x.x!m, t!e, c?y
\end{align*}
\]
5.4 MIN$_\pi$ Reduction Rules

Example of reduction: Email-Phone

5.4.1 Send/Receive
5.4.2 Input/Output

(Link Migration)

5.4.3 Blocking and Unblocking
5.4.4 Example of Reduction

\[ \text{Diagram showing the reduction process of non-deterministic interaction nets.} \]

PhD Thesis 37 June 28, 1999

Vladimir Alexiev Non-deterministic Interaction Nets University of Alberta
5.5 Completeness and Soundness

Completeness:

\[ \forall P \rightarrow \forall P' \]

\[ \forall [P] \rightarrow \exists [P'] \]

Soundness: more problematic

\[ \forall P \rightarrow \exists P' \]

\[ \forall [P] \rightarrow \forall N \rightarrow \exists [P'] \]