Handling Mobility Failures by Modal Types

Giuseppe Primiero

Department of Computer Science
Middlesex University, London
www.cs.mdx.ac.uk/people/giuseppe-primiero/

Negation as a Rule to Abort Processes, [Pfenning, 2000]

$$\frac{\Delta; \Gamma \vdash m : \bot}{\Delta; \Gamma \vdash \mathit{ABORT}(m) : \alpha}$$

Negation as a Rule to Abort Processes, [Pfenning, 2000]

$$\frac{\Delta; \Gamma \vdash m : \bot}{\Delta; \Gamma \vdash \mathit{ABORT}(m) : \alpha}$$

+ preserves correctness in a normalisation process
+ reflects redundancy requirements in distributed settings

Negation as a Rule to Abort Processes, [Pfenning, 2000]

$$\frac{\Delta; \Gamma \vdash m : \bot}{\Delta; \Gamma \vdash \mathit{ABORT}(m) : \alpha}$$

+ preserves correctness in a normalisation process
+ reflects redundancy requirements in distributed settings

- it forbids any attempt at identifying and resolving the source of error
- failures in distributed settings should be debugged for transparent re-addressing of resources

Two cases of mobility errors

i. A broadcast function fails due to wrong resources sought at the right location;

ii. a locally sent function fails due to right resources being sought at the wrong location.

Tasks

1. Formulate a type system to express local computations and their outputs;
2. Allow *mobility* of code and values via modal-style functions, interpreting global and local validity;
3. Interpret mobility failures and their resolution.

Definition (Syntax of TypErr)

$$\texttt{Terms} := x_i \mid a_i, i \in \texttt{Locations}$$

$$\texttt{Locations} := 1 < \cdots < n$$

$$\texttt{Types} := \alpha \mid \alpha \times \beta \mid \alpha + \beta \mid \alpha \supset \beta \mid \alpha \to \beta$$
$$\texttt{Types}^{\perp} := \alpha \mid \perp \mid \alpha \supset \perp \mid \alpha \to \perp$$

$$\texttt{Ops} := run_i(\alpha) \mid run_{i \cap j}(\alpha \times \beta) \mid run_{i \cup j}(\alpha \cdot \beta), \cdot \in \{+, \supset\},$$
$$\mid exec(\alpha) \mid exec(\alpha \to \beta), \alpha, \beta \in \bigcup\{\texttt{Types}, \texttt{Types}^{\perp}\}$$

$$\texttt{Data Stacks (Contexts)} := \Gamma_i \mid \Delta_i \mid \bullet_i \Gamma, \bullet \in \{\Box, \Diamond\}$$
$$\texttt{Remote Ops} := BROAD(\Box_{i \cup j}\Gamma, \alpha) \mid SEND(\Diamond_{i \cap j}\Gamma, \alpha)$$

$$\texttt{Errors} := fail@_i(\alpha), \alpha \notin \texttt{Types}^{\perp} \mid abort(\alpha)$$
$$\texttt{Access} := access@_i(a : \alpha)$$

Contexts: where is information accessible?

Contexts with running processes are considered local; contexts with output values are considered global:

Definition (Contexts)

$$\Gamma_n := \{run_i(\alpha), \ldots, run_n(\nu) \mid x_i : \alpha, \ldots, x_n : \nu\}$$
$$\Delta_n := \{exec(\alpha), \ldots, exec(\nu) \mid a_i : \alpha, \ldots, a_n : \nu\}$$

Contexts: where is information accessible?

Local and global information is expressed modally:

Definition (Modal Contexts)

For any context metavariable Γ_i, metavariable α in $\mathtt{Type}$ and a metavariable $\mathtt{Ops}$

$$\Box_n\Gamma = \{exec(\alpha) \mid \text{ for all } \alpha \in \Gamma_n\}$$
$$\Diamond_n\Gamma = \{\mathtt{Ops}(\alpha) \mid run_i(\alpha) \text{ for at least one } \alpha \in \Gamma_n\}$$

A fragment of the rules

$$\frac{}{\Delta_i, x_j : \alpha \vdash run_{i \cap j}(\alpha)}\ \text{Run}$$

$$\frac{x_j : \alpha \vdash run_j(\alpha) \qquad \Delta_i \vdash a_j : \alpha}{\Delta_i \vdash exec(\alpha)}\ \text{Value}$$

$$\frac{\Delta_i \vdash exec(\alpha)}{\Delta_i \vdash access@_j(a : \alpha)}\ @\text{I} \qquad \frac{\Delta_i ; \Gamma_i \vdash access@_j(a : \alpha)}{\Delta_i ; \Gamma_i, x_j : \alpha \vdash run_{i \cap j}(\alpha)}\ @\text{E}$$

$\Box$ as global portability

$$\frac{\Gamma_i, x_j : \alpha \vdash \mathit{run}_{i \cap j}(\alpha) \qquad \Box_i \Gamma, a_j : \alpha \vdash \mathit{exec}(\alpha)}{\mathit{BROAD}(\Box_{i \cup j} \Gamma, \alpha)} \text{ RPC1}$$

$$\frac{\mathit{BROAD}(\Box_{i \cup j} \Gamma, \alpha) \qquad \Box_i \Gamma \vdash a_j : \alpha}{\Delta_i, a_j : \alpha \vdash \mathit{exec}(\alpha)} \text{ PORT1}$$

$\Diamond$ as local portability

$$\frac{\Gamma_i, x_j : \alpha \vdash \mathit{run}_{i \cap j}(\alpha) \qquad \Diamond_i \Gamma \vdash \mathit{run}_j(\alpha)}{\mathit{SEND}(\Diamond_{i \cap j} \Gamma, \alpha)} \text{ RPC2}$$

$$\frac{\mathit{SEND}(\Diamond_{i \cap j} \Gamma, \alpha) \qquad \Diamond_i \Gamma \vdash \mathit{access@}_j(a : \alpha)}{\Gamma_i, x_j : \alpha \vdash \mathit{run}_{i \cap j}(\alpha)} \text{ PORT2}$$

Failures

Accessing wrong resources, possibly at the right location:

$$\frac{\Gamma_i, x_j : \alpha \vdash \mathit{run}_{i \cap j}(\beta) \qquad \mathit{BROAD}(\Box_{i \cup j}\Gamma, \gamma)}{\Box_i\Gamma, [x_j / c_j : \alpha / \gamma] \vdash \mathit{fail@}_{i \cap j}(\beta)}\ \text{FailPort1}$$

Re-selecting resources:

$$\frac{\Box_i\Gamma, c_j : \gamma \vdash \mathit{fail@}_{i \cap j}(\beta) \qquad \Box_i\Gamma \vdash a_j : \alpha}{\Box_i\Gamma, c_j : \gamma, a_j : \alpha \vdash \mathit{exec}(\beta)}\ \text{HFP1}$$

Failures (II)

Accessing (possibly correct) resources at wrong locations:

$$\frac{\Gamma_i; x_j : \alpha \vdash \mathit{run}_{i \cap j}(\beta) \qquad \mathit{SEND}(\Diamond_{i \cap k}\Gamma, \alpha)}{\Diamond_i \Gamma, [x_j / x_k] : \alpha \vdash \mathit{fail@}_{i \cap j}(\beta)}\ \mathsf{FailPort2}$$

Re-addressing resources at different locations:

$$\frac{\Diamond_i \Gamma, x_k : \alpha \vdash \mathit{fail@}_{i \cap j}(\beta) \qquad \Delta_i \vdash \mathit{access@}_j(a : \alpha)}{\Diamond_i \Gamma, \Delta_i, x_k : \alpha, x_j : \alpha \vdash \mathit{run}_{i \cap j}(\beta)}\ \mathsf{HFP2}$$

Meta-theory

- Local Soundness and Completeness of Connectives, Mobility and Error Functions
- Expansion and Reduction to safe states or full abort

Structural Rules

$$\frac{a_i : \alpha, c_j : \gamma \vdash exec(\beta)}{c_i : \gamma, a_j : \alpha \vdash exec(\beta)}\ \Delta\text{-Exchange}$$

$$\frac{a_i : \alpha \vdash exec(\beta)}{a_i : \alpha, c_j : \gamma \vdash exec(\beta)}\ \Delta\text{-Weakening}$$

$$\frac{x_i : \alpha \vdash run_i(\beta)}{x_i : \alpha, y_j : \gamma \vdash run_{i \cap j}(\beta)}\ \Gamma\text{-Weakening}$$

$$\frac{a_i : \alpha, a_j : \alpha \vdash exec(\beta)}{a_i : \alpha \vdash exec(\beta)}\ \Delta\text{-Contraction}$$

$$\frac{x_i : \alpha, x_j : \alpha \vdash run_{i \cap j}(\beta)}{x_i : \alpha \vdash run_i(\beta)}\ \Gamma\text{-Contraction}$$

Substructural Fragment

Theorem (Structural Failure)

Structural rules are not admissible for judgements deriving an instance of $fail@_i$.

1. failure of commutativity: resource allocation is relevant;
2. failure of weakening: resource redundancy at locations guarantees availability of a resolution strategy;
3. failure of contraction: location redundancy is not trivial.

Final Observations

- Logics in distributed settings require strategies to resolve errors of information transmission
- State-transition semantics
- Important for mechanical correctness checking

Thanks


Pfenning, F. (2000).
Constructive Logic.
Carnegie Mellon University, draft of december 28, 2000 edition.