

Separation-Based Reasoning for Message Passing Concurrency

Adrian Francalanza¹ Julian Rathke² Vladimiro Sassone²

¹Department of Computer Science
ICT, University of Malta

²DSSE, ECS
Southampton University

Northern Concurrency Workshop
November 23, 2009

Motivation

- ▶ Compositionality of Reasoning through Locality
- ▶ Uni-direction of proofs.

A Hoare-Style Logic for Message Passing

$$\{ \varphi \} P \{ \psi \}$$

1. What state should φ and ψ describe? What would be an adequate state transformer?

A Hoare-Style Logic for Message Passing

$$\{ \varphi \} P \{ \psi \}$$

1. What state should φ and ψ describe? What would be an adequate state transformer?
2. How do we define locality? How do we handle interference ?

A Hoare-Style Logic for Message Passing

$$\{ \varphi \} P \{ \psi \}$$

1. What state should φ and ψ describe? What would be an adequate state transformer?
2. How do we define locality? How do we handle interference ?
3. How do we handle bi-directionality of computation.

A State-Oriented View of Processes

$c!1 \parallel d!2 \parallel c?x.d?y.c!(x+y)$

A State-Oriented View of Processes

$c!1 \parallel d!2 \parallel c?x.d?y.c!(x+y)$

A State-Oriented View of Processes

$$\underbrace{c!1 \quad || \quad d!2}_{\text{state}} \quad || \quad c?x.d?y.c!(x+y)$$

A State-Oriented View of Processes

$$c!1 \quad || \quad d!2 \quad || \quad \underbrace{c?x.d?y.c!(x+y)}_{\text{state transformer}}$$

A State-Oriented View of Processes

$c!1 \parallel d!2 \parallel c?x.d?y.c!(x+y)$

↓

reduces to the new state

↓

$c!3$

A Hoare-Style Logic for Message Passing

$$\{ \varphi \} P \{ \psi \}$$

1. What state should φ and ψ describe? What would be an adequate state transformer?
2. How do we define locality? How do we handle interference ?
3. How do we handle bi-directionality of computation.

Interference and Non-determinism

Interference Examples

$$P_1 \parallel P_2 \parallel P_3$$

Interference and Non-determinism

Interference Examples

$$c!v_1 \quad || \quad c!v_2 \quad || \quad c?x.P$$
$$c!v \quad || \quad c?x.P_1 \quad || \quad c?x.P_2$$

Interference and Non-determinism

Interference Examples

$$\left. \begin{array}{l} c!v_1 \quad || \quad c!v_2 \quad || \quad c?x.P \\ c!v \quad || \quad c?x.P_1 \quad || \quad c?x.P_2 \end{array} \right\} \text{races!}$$

Interference and Non-determinism

Interference Examples

$$c!v_1 \quad || \quad c!v_2 \quad || \quad c?x.P$$
$$c!v \quad || \quad c?x.P_1 \quad || \quad c?x.P_2$$

Non-trivial Determinism

$$c!v_1 \quad || \quad d!v_2 \quad || \quad c?x.d?y.(a!\langle \rangle || c!\langle x+y \rangle) \quad || \quad a?.c?y.d!\langle z+1 \rangle$$

Interference and Non-determinism

Interference Examples

$$c!v_1 \quad || \quad c!v_2 \quad || \quad c?x.P$$
$$c!v \quad || \quad c?x.P_1 \quad || \quad c?x.P_2$$

Non-trivial Determinism

$$c!v_1 \quad || \quad d!v_2 \quad || \quad c?x.d?y.(a!\langle \rangle || c!\langle x+y \rangle) \quad || \quad a?.c?z.d!\langle z+1 \rangle$$

Interference and Non-determinism

Interference Examples

$$c!v_1 \quad || \quad c!v_2 \quad || \quad c?x.P$$

$$c!v \quad || \quad c?x.P_1 \quad || \quad c?x.P_2$$

Non-trivial Determinism

$$c!v_1 \quad || \quad d!v_2 \quad || \quad c?x.d?y.(a!\langle \rangle || c!\langle x+y \rangle) \quad || \quad a?.c?y.d!\langle z+1 \rangle$$

$$c!v \quad || \quad c!v \quad || \quad c?x.P_1 \quad || \quad c?x.P_2$$

Interference and Non-determinism

Interference Examples

$$c!v_1 \quad || \quad c!v_2 \quad || \quad c?x.P$$

$$c!v \quad || \quad c?x.P_1 \quad || \quad c?x.P_2$$

Non-trivial Determinism

$$c!v_1 \quad || \quad d!v_2 \quad || \quad c?x.d?y.(a!\langle \rangle || c!\langle x+y \rangle) \quad || \quad a?.c?y.d!\langle z+1 \rangle$$

$$c!v \quad || \quad c!v \quad || \quad c?x.P_1 \quad || \quad c?x.P_2$$

$$c!v_1 \quad || \quad c!v_2 \quad || \quad c?x.P \quad || \quad c?x.P$$

Interference and Non-determinism

Interference Examples

$$c!v_1 \quad || \quad c!v_2 \quad || \quad c?x.P$$
$$c!v \quad || \quad c?x.P_1 \quad || \quad c?x.P_2$$

Non-trivial Determinism

$$c!v_1 \quad || \quad d!v_2 \quad || \quad c?x.d?y.(a!\langle \rangle || c!\langle x+y \rangle) \quad || \quad a?.c?y.d!\langle z+1 \rangle$$
$$\left. \begin{array}{l} c!v \quad || \quad c!v \quad || \quad c?x.P_1 \quad || \quad c?x.P_2 \\ c!v_1 \quad || \quad c!v_2 \quad || \quad c?x.P \quad || \quad c?x.P \end{array} \right\} \text{not covered!}$$

Reasoning about Determinism (Confluence)

$$c!v_1 \parallel d!v_2 \parallel c?x.d?y.(a!\langle \rangle \parallel c!\langle x+y \rangle) \parallel a?.c?z.d!\langle z+1 \rangle$$

Reasoning about Determinism (Confluence)

$$\begin{array}{ccccccc} c!v_1 & \parallel & d!v_2 & \parallel & c?x.d?y.(a!\langle \rangle \parallel c!\langle x+y \rangle) & \parallel & a?.c?z.d!\langle z+1 \rangle \\ \{ \uparrow c \} & & \{ \uparrow d \} & & \{ \downarrow c, \downarrow d, \uparrow a \} & & \{ \downarrow a \} \end{array}$$

Reasoning about Determinism (Confluence)

$$\begin{array}{ccccccc} c!v_1 & \parallel & d!v_2 & \parallel & c?x.d?y.(a!\langle \rangle \parallel c!\langle x+y \rangle) & \parallel & a?.c?z.d!\langle z+1 \rangle \\ \{ \uparrow c \} & & \{ \uparrow d \} & & \{ \downarrow c, \downarrow d, \uparrow a \} & & \{ \downarrow a \} \end{array}$$

Reasoning about Determinism (Confluence)

$$\begin{array}{ccccccc} c!v_1 & \parallel & d!v_2 & \parallel & c?x.d?y.(a!\langle \rangle \parallel c!\langle x+y \rangle) & \parallel & a?.c?z.d!\langle z+1 \rangle \\ \{ \uparrow c \} & & \{ \uparrow d \} & & \{ \downarrow c, \downarrow d, \uparrow a \} & & \{ \downarrow a \} \end{array}$$

Reasoning about Determinism (Confluence)

$$\begin{array}{ccc} \parallel d!v_2 \parallel & \parallel d?y.(a!\langle \rangle \parallel c!\langle v_1+y \rangle) \parallel & \parallel a?.c?z.d!\langle z+1 \rangle \\ \{ \uparrow d \} & \{ \downarrow c, \downarrow d, \uparrow a, \uparrow c \} & \{ \downarrow a \} \end{array}$$

Reasoning about Determinism (Confluence)

$$\begin{array}{ccc} \parallel & d!v_2 & \parallel & d?y.(a!\langle \rangle \parallel c!\langle v_1 + y \rangle) & \parallel & a?.c?z.d!\langle z + 1 \rangle \\ & \{ \uparrow d \} & & \{ \downarrow c, \downarrow d, \uparrow a, \uparrow c \} & & \{ \downarrow a \} \end{array}$$

Reasoning about Determinism (Confluence)

$$\begin{array}{ccc} \parallel & \parallel & (a!\langle \rangle \parallel c!\langle v_1 + v_2 \rangle) \parallel a?.c?.z.d!\langle z + 1 \rangle \\ & & \{\downarrow c, \downarrow d, \uparrow a, \uparrow c, \uparrow d\} \qquad \qquad \qquad \{\downarrow a\} \end{array}$$

Reasoning about Determinism (Confluence)

$$\begin{array}{ccc} \parallel & \parallel & (a!\langle \rangle \parallel c!\langle v_1 + v_2 \rangle) \parallel a?.c?.z.d!\langle z + 1 \rangle \\ & & \{\downarrow c, \downarrow d, \uparrow a, \uparrow c, \uparrow d\} \qquad \qquad \qquad \{\downarrow a\} \end{array}$$

Reasoning about Determinism (Confluence)

$$\begin{array}{c} \parallel \\ \parallel \quad \parallel \quad a! \langle \rangle \quad \parallel \quad c! \langle v_1 + v_2 \rangle \quad \parallel \quad a?.c?.z.d! \langle z+1 \rangle \\ \{ \uparrow a, \downarrow c, \uparrow d \} \quad \{ \uparrow c, \downarrow d \} \quad \{ \downarrow a \} \end{array}$$

Reasoning about Determinism (Confluence)

$$\begin{array}{c} \parallel \\ \parallel \quad \parallel \quad \text{a!}\langle \rangle \quad \parallel \quad \text{c!}\langle v_1 + v_2 \rangle \quad \parallel \quad \text{a?.c?z.d!}\langle z + 1 \rangle \\ \{ \uparrow \text{a}, \downarrow \text{c}, \uparrow \text{d} \} \quad \{ \uparrow \text{c}, \downarrow \text{d} \} \quad \{ \downarrow \text{a} \} \end{array}$$

Reasoning about Determinism (Confluence)

$$\begin{array}{ccccccc} \parallel & & \parallel & & \parallel & c!\langle v_1 + v_2 \rangle & \parallel & c?z.d!\langle z + 1 \rangle \\ & & & & & \{ \uparrow c, \downarrow d \} & & \{ \downarrow a, \uparrow a, \downarrow c, \uparrow d \} \end{array}$$

Reasoning about Determinism (Confluence)

$$\begin{array}{ccccccc} \parallel & & \parallel & & \parallel & c!\langle v_1 + v_2 \rangle & \parallel & c?z.d!\langle z + 1 \rangle \\ & & & & & \{ \uparrow c, \downarrow d \} & & \{ \downarrow a, \uparrow a, \downarrow c, \uparrow d \} \end{array}$$

The Language

Values and Expressions

$v, u ::= 0 \mid 1 \mid 2 \dots$

$e ::= v \mid x \mid e + e \mid e - e \mid \dots$ (side-effect free)

$b ::= e \leq e \mid \neg b \mid b \wedge b$ (side-effect free)

Processes

$P, Q ::= a!e \mid a?x.P$ (communication)
| if b then P else Q (branching)
| $K(e)[\vec{c}/\vec{d}]$ (proc. definition and renaming)
| nil | $P \parallel Q$ | (new a) P (structure)

Reduction Semantics

$$\text{rIFT} \frac{b \Downarrow tt}{\text{if } b \text{ then } P \text{ else } Q \longrightarrow P}$$

$$\text{rIFF} \frac{b \Downarrow ff}{\text{if } b \text{ then } P \text{ else } Q \longrightarrow Q}$$

$$\text{rCOM} \frac{e \Downarrow v}{a!e \parallel a?x.P \longrightarrow P\{\!\{v/x\}\!\}}$$

$$\text{rPRC} \frac{K(x) \triangleq P \quad e \Downarrow v}{K(e)[\vec{c}/\vec{d}] \longrightarrow P\{\!\{v/x\}\!\}\{\!\{\vec{c}/\vec{d}\}\!\}}$$

Reduction Semantics

$$\text{rIfT} \frac{b \Downarrow tt}{\text{if } b \text{ then } P \text{ else } Q \longrightarrow P}$$

$$\text{rIfF} \frac{b \Downarrow ff}{\text{if } b \text{ then } P \text{ else } Q \longrightarrow Q}$$

$$\text{rCom} \frac{e \Downarrow v}{a!e \parallel a?x.P \longrightarrow P\{\!\{v/x\}\!\}}$$

$$\text{rPRC} \frac{K(x) \triangleq P \quad e \Downarrow v}{K(e)[\vec{c}/\vec{d}] \longrightarrow P\{\!\{v/x\}\!\}\{\!\{\vec{c}/\vec{d}\}\!\}}$$

Confined Processes

Permission Sets and Systems

$$\rho, \mu \subseteq \text{PERM} \stackrel{\text{def}}{=} \{\downarrow, \uparrow\} \times \text{NAMES}$$
$$S, T, R ::= [P]_{\rho} \mid S \parallel T \mid (\text{new } c) S$$

Confined Processes

Permission Sets and Systems

$$\rho, \mu \subseteq \text{PERM} \stackrel{\text{def}}{=} \{\downarrow, \uparrow\} \times \text{NAMES}$$
$$S, T, R ::= [P]_{\rho} \mid S \parallel T \mid (\text{new } c) S$$

Confined Processes

Permission Sets and Systems

$$\rho, \mu \subseteq \text{PERM} \stackrel{\text{def}}{=} \{\downarrow, \uparrow\} \times \text{NAMES}$$
$$S, T, R ::= [P]_\rho \mid S \parallel T \mid (\text{new } c) S$$

Example

$$c!v_1 \parallel d!v_2 \parallel c?x.d?y.(a!\langle \rangle \parallel c!\langle x+y \rangle) \parallel \dots$$

Confined Processes

Permission Sets and Systems

$$\rho, \mu \subseteq \text{PERM} \stackrel{\text{def}}{=} \{\downarrow, \uparrow\} \times \text{NAMES}$$

$$S, T, R ::= [P]_{\rho} \mid S \parallel T \mid (\text{new } c) S$$

Example

$$[c!v_1]_{\{\uparrow c\}} \parallel [d!v_2]_{\{\uparrow d\}} \parallel [c?x.d?y.(a!\langle \rangle \parallel c!\langle x+y \rangle)]_{\{\downarrow c, \downarrow d, \uparrow a\}} \parallel \dots$$

Reduction Semantics

$$\text{cCom} \frac{e \Downarrow v}{a!e \parallel a?x.P \longrightarrow P\{v/x\}}$$

Confined Processes

Reduction Semantics

$$\text{cCom} \frac{e \Downarrow v}{[a!e]_{\rho} \parallel [a?x.P]_{\mu} \longrightarrow [P\{v/x\}]_{\mu \uplus \rho}}$$

Confined Processes

Reduction Semantics

$$\text{cCom} \frac{e \Downarrow v \quad \uparrow c \in \rho \quad \downarrow c \in \mu}{\llbracket a!e \rrbracket_{\rho} \parallel \llbracket a?x.P \rrbracket_{\mu} \longrightarrow \llbracket P\{v/x\} \rrbracket_{\mu \uplus \rho}}$$

Confined Processes

Reduction Semantics

$$\text{cCom} \frac{e \Downarrow v \quad \uparrow c \in \rho \quad \downarrow c \in \mu}{\llbracket a!e \rrbracket_{\rho} \parallel \llbracket a?x.P \rrbracket_{\mu} \longrightarrow \llbracket P \{v/x\} \rrbracket_{\mu \uplus \rho}}$$

$$\text{cSpl} \frac{}{\llbracket P \parallel Q \rrbracket_{\rho \uplus \mu} \longrightarrow \llbracket P \rrbracket_{\rho} \parallel \llbracket Q \rrbracket_{\mu}}$$

Confined Processes

Reduction Semantics

$$\text{cCom} \frac{e \Downarrow v \quad \uparrow c \in \rho \quad \downarrow c \in \mu}{\llbracket a!e \rrbracket_{\rho} \parallel \llbracket a?x.P \rrbracket_{\mu} \longrightarrow \llbracket P \llbracket v/x \rrbracket \rrbracket_{\mu \uplus \rho}}$$

$$\text{cSpl} \frac{}{\llbracket P \parallel Q \rrbracket_{\rho \uplus \mu} \longrightarrow \llbracket P \rrbracket_{\rho} \parallel \llbracket Q \rrbracket_{\mu}}$$

$$\text{cLcl} \frac{}{\llbracket (\text{new } c)P \rrbracket_{\rho} \longrightarrow (\text{new } c) \llbracket P \rrbracket_{\rho \uplus \{\downarrow c, \uparrow c\}}}$$

Confined Processes

What do we gain?

- ▶ Explicit *Permission Ownership*: $[P]_p$

Confined Processes

What do we gain?

- ▶ Explicit *Permission Ownership*: $\lceil P \rceil_p$
- ▶ Explicit *Ownership Transfer*: $c\text{COM}$, $c\text{SPL}$, $c\text{LCL}$

Confined Processes

What do we gain?

- ▶ Explicit *Permission Ownership*: $\lceil P \rceil_\rho$
- ▶ Explicit *Ownership Transfer*: cCOM, cSPL, cLCL
- ▶ Explicit Races and Separation:

Confined Processes

What do we gain?

- ▶ Explicit *Permission Ownership*: $\lceil P \rceil_\rho$
- ▶ Explicit *Ownership Transfer*: $c\text{COM}$, $c\text{SPL}$, $c\text{LCL}$
- ▶ Explicit Races and Separation:

$$e\text{OUT} \frac{\uparrow c \notin \rho}{\lceil c!e \rceil_\rho \longrightarrow_{\text{race}}}$$

Confined Processes

What do we gain?

- ▶ Explicit *Permission Ownership*: $\lceil P \rceil_\rho$
- ▶ Explicit *Ownership Transfer*: $c\text{COM}$, $c\text{SPL}$, $c\text{LCL}$
- ▶ Explicit Races and Separation:

$$\text{EOUT} \frac{\uparrow c \notin \rho}{\lceil c!e \rceil_\rho \longrightarrow_{\text{race}}}$$

$$\text{EOIN} \frac{\downarrow c \notin \rho}{\lceil c?x.P \rceil_\rho \longrightarrow_{\text{race}}}$$

Confined Processes

What do we gain?

- ▶ Explicit *Permission Ownership*: $\lceil P \rceil_\rho$
- ▶ Explicit *Ownership Transfer*: $c\text{COM}$, $c\text{SPL}$, $c\text{LCL}$
- ▶ Explicit Races and Separation:

$$\begin{array}{c} \text{EOUT} \frac{\uparrow c \notin \rho}{\lceil c!e \rceil_\rho \longrightarrow_{\text{race}}} \\ \text{EOIN} \frac{\downarrow c \notin \rho}{\lceil c?x.P \rceil_\rho \longrightarrow_{\text{race}}} \\ \text{ESep} \frac{\rho \cap \mu \neq \emptyset}{\lceil P \rceil_\rho \parallel \lceil Q \rceil_\mu \longrightarrow_{\text{race}}} \end{array}$$

Confinement and linear Permissions

How does it work?

Recall:

$$c!v_1 \quad || \quad c!v_2 \quad || \quad c?x.P$$

Confinement and linear Permissions

How does it work?

Recall:

$$c!v_1 \quad \parallel \quad c!v_2 \quad \parallel \quad [c?x.P]_{\{\downarrow c, \dots\}}$$

Confinement and linear Permissions

How does it work?

Recall:

$$\llbracket c!v_1 \rrbracket_{\{\dots\}} \parallel \llbracket c!v_2 \rrbracket_{\{\uparrow c, \dots\}} \parallel \llbracket c?x.P \rrbracket_{\{\downarrow c, \dots\}} \longrightarrow_{\text{race}}$$

because $\llbracket c!v_1 \rrbracket_{\{\dots\}} \longrightarrow_{\text{race}}$

Confinement and linear Permissions

How does it work?

Recall:

$$\llbracket c!v_1 \rrbracket_{\{\uparrow c, \dots\}} \parallel \llbracket c!v_2 \rrbracket_{\{\dots\}} \parallel \llbracket c?x.P \rrbracket_{\{\downarrow c, \dots\}} \longrightarrow_{\text{race}}$$

because $\llbracket c!v_2 \rrbracket_{\{\dots\}} \longrightarrow_{\text{race}}$

Confinement and linear Permissions

How does it work?

Recall:

$$\llbracket c!v_1 \rrbracket_{\{\uparrow c, \dots\}} \parallel \llbracket c!v_2 \rrbracket_{\{\uparrow c, \dots\}} \parallel \llbracket c?x.P \rrbracket_{\{\downarrow c, \dots\}} \longrightarrow_{\text{race}}$$

$$\text{because } \llbracket c!v_1 \rrbracket_{\{\uparrow c, \dots\}} \parallel \llbracket c!v_2 \rrbracket_{\{\uparrow c, \dots\}} \longrightarrow_{\text{race}}$$

Confinement Properties

Race Preservation

$S \longrightarrow^* T$ and $S \longrightarrow_{\text{race}}$ implies $T \longrightarrow_{\text{race}}$

Correspondence

$S \not\rightarrow_{\text{race}}$ implies $\begin{cases} S \longrightarrow T \text{ implies } |S| \longrightarrow^* |T| \\ |S| \longrightarrow |T| \text{ implies } S \longrightarrow^* T \end{cases}$

Evaluation Confluence (Determinism)

$\left. \begin{array}{l} S \longrightarrow^* T_1 \not\rightarrow \text{ and } T_1 \not\rightarrow_{\text{race}} \\ \text{and } S \longrightarrow^* T_2 \not\rightarrow \text{ and } T_2 \not\rightarrow_{\text{race}} \end{array} \right\} \text{ implies } |T_1| \equiv |T_2|$

Recap: A Hoare-Style Logic for Message Passing

$$\{ \varphi \} \quad ? \quad \{ \psi \}$$

1. What state should φ and ψ describe? What would be an adequate state transformer?
2. How do we define locality? How do we handle interference ?
3. How do we handle bi-directionality of computation.

Recap: A Hoare-Style Logic for Message Passing

$$\{ \varphi \} \quad ? \quad \{ \psi \}$$

1. What state should φ and ψ describe? What would be an adequate state transformer?
2. How do we define locality? How do we handle interference ?
3. How do we handle bi-directionality of computation.

Recap: A Hoare-Style Logic for Message Passing

$$\{ \varphi \} \quad ? \quad \{ \psi \}$$

1. What state should φ and ψ describe? What would be an adequate state transformer? **confined processes!**
2. How do we define locality? How do we handle interference ?
3. How do we handle bi-directionality of computation.

Recap: A Hoare-Style Logic for Message Passing

$$\{ \varphi \} S \{ \psi \}$$

1. What state should φ and ψ describe? What would be an adequate state transformer? **confined processes!**
2. How do we define locality? How do we handle interference ? **confined processes!**
3. How do we handle bi-directionality of computation.

Hoare Rules For Message Passing

$$\{ \varphi \} S \{ \psi \}$$

Two factors at work!

1. Verify uni-directionally that from φ , S will lead us to ψ .
2. Assuming a race-free T satisfying φ , construct a state transformer S that is race-free.

Hoare Rules For Message Passing

$$\{ \varphi \} S \{ \psi \}$$

Recall...

$$\begin{array}{ccccc} \parallel & \parallel & a!\langle \rangle & \parallel & c!\langle v_1 + v_2 \rangle & \parallel & a?.c?z.d!\langle z + 1 \rangle \\ & & \{\uparrow a, \downarrow c, \uparrow d\} & & \{\uparrow c, \downarrow d\} & & \{\downarrow a\} \end{array}$$

Hoare Rules For Message Passing

$$\{ \varphi \} S \{ \psi \}$$

Decomposition

$$\lceil a! \langle \rangle \rceil_{\{\uparrow a, \downarrow c, \uparrow d\}} \parallel \lceil c! \langle v_1 + v_2 \rangle \rceil_{\{\uparrow c, \downarrow d\}} \parallel \lceil a?.c?z.d! \langle z + 1 \rangle \rceil_{\{\downarrow a\}}$$

Hoare Rules For Message Passing

$$\{ \varphi \} S \{ \psi \}$$

Decomposition

$$\lceil a! \langle \rangle \rceil_{\{\uparrow a, \downarrow c, \uparrow d\}} \parallel \lceil c! \langle v_1 + v_2 \rangle \rceil_{\{\uparrow c, \downarrow d\}} \parallel \underbrace{\lceil a?.c?.z.d!z + 1 \rceil_{\{\downarrow a\}}}_{\text{analyse in isolation}}$$

Hoare Rules For Message Passing

$$\{ \varphi \} S \{ \psi \}$$

Decomposition

$$\left\{ \overbrace{a\langle \rangle * c\langle v_1 + v_2 \rangle}^{\varphi} \right\} [a?.c?z.d!z + 1]_{\{Ja\}} \left\{ \overbrace{d\langle v_1 + v_2 + 1 \rangle}^{\psi} \right\}$$

Hoare Rules For Message Passing

$$\{ \varphi \} S \{ \psi \}$$

Decomposition

$$\left\{ \overbrace{a\langle \rangle * c\langle v_1 + v_2 \rangle}^{\varphi} \right\} \underbrace{[a?.c?z.d!z + 1]_{\{Ja\}}}_{\text{replies on } a!\langle \rangle \text{ providing } \downarrow c, \uparrow d} \left\{ \overbrace{d\langle v_1 + v_2 + 1 \rangle}^{\psi} \right\}$$

Hoare Rules For Message Passing

$$\Gamma \vdash \{ \varphi \} S \{ \psi \}$$

Decomposition

$$\Gamma \vdash \left\{ \overbrace{a\langle \rangle * c\langle v_1 + v_2 \rangle}^{\varphi} \right\} \underbrace{[a?.c?z.d!z + 1]_{\{Ja\}}}_{\text{replies on } a\langle \rangle \text{ providing } \downarrow c, \uparrow d} \left\{ \overbrace{d\langle v_1 + v_2 + 1 \rangle}^{\psi} \right\}$$

$$\Gamma = a:\{\uparrow a, \downarrow c, \uparrow d\}, \quad c:\{\uparrow c\}, \quad d:\{\uparrow d\}$$

The Logic

S is closed, stable and race-free. Formulas are closed

$\Gamma, S \models \mathbf{emp}$ if $S \equiv [\mathbf{nil}]_{\emptyset}$;

$\Gamma, S \models c\langle e \rangle$ if $\left\{ \begin{array}{l} S \equiv [c!e']_{\rho} \\ \text{with } e \Downarrow v, e' \Downarrow v \text{ and } \Gamma(c) \subseteq \rho; \end{array} \right.$

$\Gamma, S \models \varphi_1 * \varphi_2$ if $\left\{ \begin{array}{l} \text{exists } S_1, S_2 \text{ where } S \equiv S_1 \parallel S_2 \\ \text{with } \Gamma, S_1 \models \varphi_1 \text{ and } \Gamma, S_2 \models \varphi_2; \end{array} \right.$

Recap: A Hoare-Style Logic for Message Passing

$$\Gamma \vdash \{ \varphi \} S \{ \psi \}$$

- ▶ What state should φ and ψ describe? What would be an adequate state transformer?
- ▶ How do we define locality? How do we handle interference ?
- ▶ How do we handle bi-directionality of computation.

Recap: A Hoare-Style Logic for Message Passing

$$\Gamma \vdash \{ \varphi \} S \{ \psi \}$$

- ▶ ...
- ▶ How do we handle bi-directionality of computation.

$$\Gamma \vdash \{ a \langle v \rangle \} \quad [a?x.(b!x \parallel c?z.d!x)]_\rho \parallel [b?y.c!y]_\mu \quad \{ d \langle v \rangle \}$$

Recap: A Hoare-Style Logic for Message Passing

$$\Gamma \vdash \{ \varphi \} S \{ \psi \}$$

- ▶ ...
- ▶ How do we handle bi-directionality of computation.

$$\Gamma \vdash \{ a\langle v \rangle \} \quad [a?x.(b!x \parallel c?z.d!x)]_\rho \parallel [b?y.c!y]_\mu \quad \{ d\langle v \rangle \}$$

Recap: A Hoare-Style Logic for Message Passing

$$\Gamma \vdash \{ \varphi \} S \{ \psi \}$$

- ▶ ...
- ▶ How do we handle bi-directionality of computation.

$$\Gamma \vdash \{ a \langle v \rangle \} \quad [a?x.(b!x \parallel c?z.d!x)]_\rho \parallel [b?y.c!y]_\mu \quad \{ d \langle v \rangle \}$$

Recap: A Hoare-Style Logic for Message Passing

$$\Gamma \vdash \{ \varphi \} S \{ \psi \}$$

- ▶ ...
- ▶ How do we handle bi-directionality of computation.

$$\Gamma \vdash \{ a \langle v \rangle \} \quad [a?x.(b!x \parallel c?z.d!x)]_\rho \parallel [b?y.c!y]_\mu \quad \{ d \langle v \rangle \}$$

Recap: A Hoare-Style Logic for Message Passing

$$\Gamma \vdash \{ \varphi \} S \{ \psi \}$$

- ▶ ...
- ▶ How do we handle bi-directionality of computation.

$$\Gamma \vdash \{ a\langle v \rangle \} \quad [a?x.(b!x \parallel c?z.d!x)]_\rho \parallel [b?y.c!y]_\mu \quad \{ d\langle v \rangle \}$$

Recap: A Hoare-Style Logic for Message Passing

$$\Gamma \vdash \{ \varphi \} S \{ \psi \}$$

- ▶ ...
- ▶ How do we handle bi-directionality of computation.

$$\begin{array}{l} \Gamma \vdash \{ a\langle v \rangle \} \quad [a?x.(b!x \parallel c?z.d!x)]_p \quad \{ \varphi \} \\ \Gamma \vdash \{ \varphi \} \quad [b?y.c!y]_\mu \quad \{ d\langle v \rangle \} \end{array}$$

What would φ be?

The Logic

$$\begin{aligned}\varphi, \psi \in \text{FRM} &::= \mathbf{emp} \mid c\langle e \rangle \mid \varphi * \psi \mid \chi \multimap \varphi \mid \dots \\ \chi \in \text{SFRM} &::= \mathbf{emp} \mid c\langle e \rangle \mid \chi * \chi\end{aligned}$$

The Logic

$\varphi, \psi \in \text{FRM} ::= \mathbf{emp} \mid c\langle e \rangle \mid \varphi * \psi \mid \chi \multimap \varphi \mid \dots$

$\chi \in \text{SFRM} ::= \mathbf{emp} \mid c\langle e \rangle \mid \chi * \chi$

...

$\Gamma, S \models \chi \multimap \varphi$ if $\left\{ \begin{array}{l} \Gamma \vdash S \perp \chi \\ \text{and } \Gamma, T \models \chi, \text{ where } S \perp T \\ \text{implies } S \parallel T \longrightarrow^* R \text{ and } \Gamma, R \models \varphi; \end{array} \right.$

The Logic

$$\begin{aligned}\varphi, \psi \in \text{FRM} &::= \mathbf{emp} \mid c\langle e \rangle \mid \varphi * \psi \mid \chi \multimap \psi \mid \dots \\ \chi \in \text{SFRM} &::= \mathbf{emp} \mid c\langle e \rangle \mid \chi * \chi\end{aligned}$$
$$\begin{aligned}\Gamma \vdash \{a\langle v \rangle\} \quad [a?x.(b!x \parallel c?z.d!x)]_{\rho} \quad & \{b\langle v \rangle * (c\langle v \rangle \multimap d\langle v \rangle)\} \\ \Gamma \vdash \{b\langle v \rangle * (c\langle v \rangle \multimap d\langle v \rangle)\} \quad [b?y.c!y]_{\mu} \quad & \{d\langle v \rangle\}\end{aligned}$$

The Proof System

$$\Gamma \models \{\varphi\} S \{\psi\} \stackrel{\text{def}}{=} \begin{cases} \Gamma, T \models \varphi \text{ and } T \perp S \\ \text{implies } T \parallel S \longrightarrow^* R \text{ and } \Gamma, R \models \psi \end{cases}$$

Proof Rules

$$\text{LOut} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{\mathbf{emp}\} \quad [c!e_1]_\rho \quad \{c\langle e_2 \rangle\}}$$

$$\text{LOutA} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{c\langle e_2 \rangle \multimap \varphi\} \quad [c!e_1]_\rho \quad \{\varphi\}}$$

$$\text{LIIn} \frac{\Gamma \vdash \{\varphi\} \quad [P\{e/x\}]_\rho \quad \{\psi\}}{\Gamma \vdash \{\varphi * c\langle e \rangle\} \quad [c?x.P]_{\rho \setminus \Gamma(c)} \quad \{\psi\}}$$

Proof Rules

$$\text{LOut} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{\mathbf{emp}\} \quad [c!e_1]_\rho \quad \{c\langle e_2 \rangle\}}$$

$$\text{LOutA} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{c\langle e_2 \rangle \multimap \varphi\} \quad [c!e_1]_\rho \quad \{\varphi\}}$$

$$\text{LIIn} \frac{\Gamma \vdash \{\varphi\} \quad [P\{e/x\}]_\rho \quad \{\psi\}}{\Gamma \vdash \{\varphi * c\langle e \rangle\} \quad [c?x.P]_{\rho \setminus \Gamma(c)} \quad \{\psi\}}$$

Proof Rules

$$\text{LOut} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{\mathbf{emp}\} \quad [c!e_1]_\rho \quad \{c\langle e_2 \rangle\}}$$

$$\text{LOutA} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{c\langle e_2 \rangle \multimap \varphi\} \quad [c!e_1]_\rho \quad \{\varphi\}}$$

$$\text{LIIn} \frac{\Gamma \vdash \{\varphi\} \quad [P\{e/x\}]_\rho \quad \{\psi\}}{\Gamma \vdash \{\varphi * c\langle e \rangle\} \quad [c?x.P]_{\rho \setminus \Gamma(c)} \quad \{\psi\}}$$

Proof Rules

$$\text{LOut} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{\mathbf{emp}\} \quad [c!e_1]_\rho \quad \{c\langle e_2 \rangle\}}$$

$$\text{LOutA} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{c\langle e_2 \rangle -\boxtimes \varphi\} \quad [c!e_1]_\rho \quad \{\varphi\}}$$

$$\text{LIIn} \frac{\Gamma \vdash \{\varphi\} \quad [P\{e/x\}]_\rho \quad \{\psi\}}{\Gamma \vdash \{\varphi * c\langle e \rangle\} \quad [c?x.P]_{\rho \setminus \Gamma(c)} \quad \{\psi\}}$$

Proof Rules

$$\text{LOut} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{\mathbf{emp}\} \quad [c!e_1]_\rho \quad \{c\langle e_2 \rangle\}}$$

$$\text{LOutA} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{c\langle e_2 \rangle \multimap \varphi\} \quad [c!e_1]_\rho \quad \{\varphi\}}$$

$$\text{LIIn} \frac{\Gamma \vdash \{\varphi\} \quad [P\langle e/x \rangle]_\rho \quad \{\psi\}}{\Gamma \vdash \{\varphi * c\langle e \rangle\} \quad [c?x.P]_{\rho \setminus \Gamma(c)} \quad \{\psi\}}$$

Proof Rules

$$\text{LOut} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{\mathbf{emp}\} \quad [c!e_1]_\rho \quad \{c\langle e_2 \rangle\}}$$

$$\text{LOutA} \frac{e_1 \Downarrow v \quad e_2 \Downarrow v \quad \Gamma(c) \in \rho}{\Gamma \vdash \{c\langle e_2 \rangle \multimap \varphi\} \quad [c!e_1]_\rho \quad \{\varphi\}}$$

$$\text{LIIn} \frac{\Gamma \vdash \{\varphi\} \quad [P\{e/x\}]_\rho \quad \{\psi\}}{\Gamma \vdash \{\varphi * c\langle e \rangle\} \quad [c?x.P]_{\rho \setminus \Gamma(c)} \quad \{\psi\}}$$

Proof Rules

$$\text{L}_{\text{PAR}} \frac{\begin{array}{c} \Gamma \vdash \{\varphi_1\} \quad S \quad \{\psi_1 * \varphi_3\} \\ \Gamma \vdash \{\varphi_2 * \varphi_3\} \quad T \quad \{\psi_2\} \quad \Gamma \vdash \psi_1 \perp \psi_2 \end{array}}{\Gamma \vdash \{\varphi_1 * \varphi_2\} \quad S \parallel T \quad \{\psi_1 * \psi_2\}}$$

$$\text{L}_{\text{ADJ}} \frac{\Gamma \vdash \{\varphi * \chi\} \quad S \quad \{\psi\} \quad \Gamma \vdash S \perp \chi}{\Gamma \vdash \{\varphi\} \quad S \quad \{\chi \multimap \psi\}}$$

$$\text{L}_{\text{ADJH}} \frac{\Gamma \vdash \{\varphi\} \quad S \quad \{\chi \multimap \psi\}}{\Gamma \vdash \{\varphi * \chi\} \quad S \quad \{\psi\}}$$

Proof Rules

$$\text{L}_{\text{PAR}} \frac{\begin{array}{c} \Gamma \vdash \{\varphi_1\} \quad S \quad \{\psi_1 * \varphi_3\} \\ \Gamma \vdash \{\varphi_2 * \varphi_3\} \quad T \quad \{\psi_2\} \quad \Gamma \vdash \psi_1 \perp \psi_2 \end{array}}{\Gamma \vdash \{\varphi_1 * \varphi_2\} \quad S \parallel T \quad \{\psi_1 * \psi_2\}}$$

$$\text{L}_{\text{ADJ}} \frac{\Gamma \vdash \{\varphi * \chi\} \quad S \quad \{\psi\} \quad \Gamma \vdash S \perp \chi}{\Gamma \vdash \{\varphi\} \quad S \quad \{\chi \multimap \psi\}}$$

$$\text{L}_{\text{ADJH}} \frac{\Gamma \vdash \{\varphi\} \quad S \quad \{\chi \multimap \psi\}}{\Gamma \vdash \{\varphi * \chi\} \quad S \quad \{\psi\}}$$

Proof Rules

$$\text{L}_{\text{PAR}} \frac{\begin{array}{c} \Gamma \vdash \{\varphi_1\} \quad S \quad \{\psi_1 * \varphi_3\} \\ \Gamma \vdash \{\varphi_2 * \varphi_3\} \quad T \quad \{\psi_2\} \quad \Gamma \vdash \psi_1 \perp \psi_2 \end{array}}{\Gamma \vdash \{\varphi_1 * \varphi_2\} \quad S \parallel T \quad \{\psi_1 * \psi_2\}}$$

$$\text{L}_{\text{ADJ}} \frac{\Gamma \vdash \{\varphi * \chi\} \quad S \quad \{\psi\} \quad \Gamma \vdash S \perp \chi}{\Gamma \vdash \{\varphi\} \quad S \quad \{\chi \multimap \psi\}}$$

$$\text{L}_{\text{ADJH}} \frac{\Gamma \vdash \{\varphi\} \quad S \quad \{\chi \multimap \psi\}}{\Gamma \vdash \{\varphi * \chi\} \quad S \quad \{\psi\}}$$

Frame Rules

$$\text{LFRM} \frac{\Gamma \vdash \{\varphi_1\} \quad \mathcal{S} \quad \{\varphi_2\} \quad \Gamma \vdash \varphi_2 \perp \psi}{\Gamma \vdash \{\varphi_1 * \psi\} \quad \mathcal{S} \quad \{\varphi_2 * \psi\}}$$

$$\text{LFRMA} \frac{\Gamma \vdash \{\varphi\} \quad \mathcal{S} \quad \{\psi\} \quad \Gamma \vdash \mathcal{S} \perp \chi}{\Gamma \vdash \{\chi \text{-}\boxtimes \varphi\} \quad \mathcal{S} \quad \{\chi \text{-}\boxtimes \psi\}}$$

Frame Rules

$$\text{LFRM} \frac{\Gamma \vdash \{\varphi_1\} \quad S \quad \{\varphi_2\} \quad \Gamma \vdash \varphi_2 \perp \psi}{\Gamma \vdash \{\varphi_1 * \psi\} \quad S \quad \{\varphi_2 * \psi\}}$$

$$\text{LFRMA} \frac{\Gamma \vdash \{\varphi\} \quad S \quad \{\psi\} \quad \Gamma \vdash S \perp \chi}{\Gamma \vdash \{\chi -\boxtimes \varphi\} \quad S \quad \{\chi -\boxtimes \psi\}}$$

Main Result

Soundness

$$\Gamma \vdash \{\varphi\} S \{\psi\} \text{ implies } \Gamma \models \{\varphi\} S \{\psi\}$$

Application

Proved the correctness of an in-place Quicksort algorithm expressed in Value Passing CCS.

Summary

Summary

Model: Confined processes

- ▶ Give us explicit ownership, ownership transfer and units of separation
- ▶ Correspondence with process semantics
- ▶ Narrative as to why process is deterministic.

Summary

Model: Confined processes

- ▶ Give us explicit ownership, ownership transfer and units of separation
- ▶ Correspondence with process semantics
- ▶ Narrative as to why process is deterministic.

Logic: Asynchronous messages have a dual role:

- ▶ interpreted as the *state* of a process.
- ▶ main mechanism for mutual exclusion.

Summary

Model: Confined processes

- ▶ Give us explicit ownership, ownership transfer and units of separation
- ▶ Correspondence with process semantics
- ▶ Narrative as to why process is deterministic.

Logic: Asynchronous messages have a dual role:

- ▶ interpreted as the *state* of a process.
- ▶ main mechanism for mutual exclusion.

Proof System: Acts on soups of asynchronous messages

- ▶ constructs safe confined processes *i.e.*, deterministic.
- ▶ Use (restricted) spatial adjunct to handle bidirectionality of messages.