Structural Equivalence in Reversible Calculus of Communicating Systems

Southeast Regional Programming Languages Seminar 2019

Clément Aubert[1] Ioana-Domnina Cristescu[2]

[1]Augusta University – School of Computer & Cyber Sciences

[2]INRIA – TAMIS team

Augusta, GA, May 11th, 2019

Goal

Specifying Reversible Concurrent Computation

Goal

Specifying Computation

- What?
  Formally prove correctness / “Correct by design”.
- Why?
  - To get correct programs!
  - Foundation for programming methodology.
  - Qualitative correctness.
- How?
  Type systems, proof assistant, model checking, static code analysis, etc.

Goal

Specifying Concurrent Computation

- What?
  Concurrent = multiprocessing, parallel, distributed, etc.
- Why?
  - Concurrent programs are tricky.
  - Understand what concurrency can give.
  - Absence of livelock, deadlock, etc.
- How?
  Process calculi (CCS, π-calculus, ...), event structures, Petri nets, actor model, etc.

Goal

Specifying **Reversible** Computation

- What?
  Computation that can backtrack.
- Why?
  - Non-destructive computation.
  - Landauer's principle: free energy!
  - Quantum computing.
- How?
  Reversible automaton, quantum circuits, etc.

Goal

Specifying Reversible Concurrent Computation

Goal

Specifying Reversible Concurrent Computation

- What?
  Memory needs to be "enough", "not too big", **and** distributed.
- Why?
  - Combine all the *benefits* of reversible *and* concurrent computation!
  - But also all the *difficulties* . . .
  - Network of reversible computers!
- How?
  Reversing process calculi, reversible event structures, etc.

Goal

Specifying Reversible Concurrent Computation

RCCS
adds
Reversibility
to the
Calculus of **C**ommunicating **S**ystems

A Tension in any Specification

Formal VS Easy to use

A Tension in any Specification

Formal VS Easy to use

From a Textbook on Process Algebra

"In a process-algebraic approach to system verification, one typically writes two specifications. One, call it SYS, captures the design of the actual system and the other, call it SPEC, describes the system's desired 'high-level' behavior. One may then establish the correctness of SYS with respect to SPEC by showing that SYS behaves the 'same as' SPEC." (Bergstra, Ponse, and Smolka, 2001, p. V)

A Tension in any Specification

Formal VS Easy to use

From a Textbook on Process Algebra

"In a process-algebraic approach to system verification, one typically writes two specifications. One, call it SYS, captures the design of the actual system and the other, call it SPEC, describes the system's desired 'high-level' behavior. One may then establish the correctness of SYS with respect to SPEC by showing that SYS behaves the 'same as' SPEC." (Bergstra, Ponse, and Smolka, 2001, p. V)

A.k.a. *Who works without α-equivalence (renaming of bound variables)?*

In CCS

1. Define $\mathrm{SYS}_{\mathrm{CCS}}$
2. Define a structural equivalence $\equiv$
3. $\mathrm{SYS}_{\mathrm{CCS}} + \equiv = \mathrm{SPEC}_{\mathrm{CCS}}$
4. Prove $\mathrm{SYS}_{\mathrm{CCS}}$ “=” $\mathrm{SPEC}_{\mathrm{CCS}}$

Was RCCS defined the same way?

CCS process

$$P, Q \coloneqq \lambda.P \;\Big|\; \sum_{i \in I} P_i \;\Big|\; A \;\Big|\; P \mid Q \;\Big|\; P \backslash a \;\Big|\; P[a \leftarrow b] \;\Big|\; 0$$

A are (recursive) definitions of processes

$$\frac{}{\lambda.P \xrightarrow{\lambda} P}\ \text{act.}$$

$$\frac{}{\lambda.P \xrightarrow{\lambda} P}\ \text{act.} \qquad \frac{P_j \xrightarrow{\alpha} P'_j \qquad j \in I}{\sum_{i \in I} P_i \xrightarrow{\alpha} P'_i}\ \text{sum.} \qquad \frac{P \xrightarrow{\alpha} P' \qquad A \overset{\text{def}}{=} P}{A \xrightarrow{\alpha} P'}\ \text{rec.}$$

$$\frac{}{\lambda.P \xrightarrow{\lambda} P}\ \text{act.} \qquad \frac{P_j \xrightarrow{\alpha} P_j' \quad j \in I}{\sum_{i \in I} P_i \xrightarrow{\alpha} P_i'}\ \text{sum.} \qquad \frac{P \xrightarrow{\alpha} P' \quad A \stackrel{\text{def}}{=} P}{A \xrightarrow{\alpha} P'}\ \text{rec.}$$

$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q}\ \text{com.}_1 \qquad \frac{Q \xrightarrow{\alpha} Q'}{P \mid Q \xrightarrow{\alpha} P \mid Q'}\ \text{com.}_2$$

$$\frac{P \xrightarrow{\lambda} P' \quad Q \xrightarrow{\overline{\lambda}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}\ \text{syn.}$$

$$\frac{}{\lambda.P \xrightarrow{\lambda} P}\ \text{act.} \qquad \frac{P_j \xrightarrow{\alpha} P'_j \quad j \in I}{\sum_{i \in I} P_i \xrightarrow{\alpha} P'_i}\ \text{sum.} \qquad \frac{P \xrightarrow{\alpha} P' \quad A \overset{\text{def}}{=} P}{A \xrightarrow{\alpha} P'}\ \text{rec.}$$

$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q}\ \text{com.}_1 \qquad \frac{Q \xrightarrow{\alpha} Q'}{P \mid Q \xrightarrow{\alpha} P \mid Q'}\ \text{com.}_2$$

$$\frac{P \xrightarrow{\lambda} P' \quad Q \xrightarrow{\overline{\lambda}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}\ \text{syn.}$$

$$\frac{P \xrightarrow{\alpha} P' \quad a \neq \alpha}{P \backslash a \xrightarrow{\alpha} P' \backslash a}\ \text{res.} \qquad \frac{P \xrightarrow{\alpha} P'}{P[\vec{a} \leftarrow \vec{b}] \xrightarrow{\alpha[\vec{a} \leftarrow \vec{b}]} P'[\vec{a} \leftarrow \vec{b}]}\ \text{ren.}$$

$$A \stackrel{\text{def}}{=} a.0 + (b.0 \mid ((\overline{c}.A \mid c.0)\backslash c))$$

$$A \xrightarrow{a} 0$$

$$\begin{aligned} A &\xrightarrow{b} 0 \mid ((\overline{c}.A \mid c.0)\backslash c) \\ &\xrightarrow{\tau} 0 \mid (A \mid 0)\backslash c \end{aligned}$$

$$\begin{aligned} A &\xrightarrow{\tau} b.0 \mid (A \mid 0\backslash c) \\ &\xrightarrow{b} 0 \mid (A \mid 0)\backslash c \end{aligned}$$

Make my life easier!

$$
\begin{array}{rll}
0 \mid (A \mid 0)\backslash c & \text{“is the same as”} & A \\
P \mid 0 & \text{“is the same as”} & P \\
P \mid Q & \text{“is the same as”} & Q \mid P \\
P\backslash a & \text{“is the same as”} & P[a \leftarrow b]\backslash b \\
& \dots &
\end{array}
$$

Make my life easier!

$$
\begin{array}{rll}
0 \mid (A \mid 0)\backslash c & \text{“is the same as”} & A \\
P \mid 0 & \text{“is the same as”} & P \\
P \mid Q & \text{“is the same as”} & Q \mid P \\
P\backslash a & \text{“is the same as”} & P[a \leftarrow b]\backslash b \\
& \dots &
\end{array}
$$

Structural Equivalence

$$
\begin{array}{cc}
P \mid 0 \equiv P & \\
P \mid Q \equiv Q \mid P & P + Q \equiv Q + P \\
(P \mid Q) \mid V \equiv P \mid (Q \mid V) & (P + Q) + V \equiv P + (Q + V) \\
(P\backslash a) \mid Q \equiv (P \mid Q)\backslash a \text{ with } a \notin \mathrm{fn}(Q) & (P\backslash a)\backslash b \equiv (P\backslash b)\backslash a \\
A \stackrel{\text{def}}{=} P \Rightarrow A \equiv P & P =_\alpha Q \Rightarrow P \equiv Q
\end{array}
$$

$$\frac{}{\lambda.P \xrightarrow{\lambda} P}$$

$$\frac{P_j \xrightarrow{\alpha} P'_j \qquad j \in I}{\sum_{i \in I} P_i \xrightarrow{\alpha} P'_i}$$

$$\frac{P \xrightarrow{\alpha} P' \qquad A \stackrel{\text{def}}{=} P}{A \xrightarrow{\alpha} P'}$$

$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q}$$

$$\frac{Q \xrightarrow{\alpha} Q'}{P \mid Q \xrightarrow{\alpha} P \mid Q'}$$

$$\frac{P \xrightarrow{\lambda} P' \qquad Q \xrightarrow{\overline{\lambda}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

$$\frac{P \xrightarrow{\alpha} P' \qquad a \neq \alpha}{P \backslash a \xrightarrow{\alpha} P' \backslash a}$$

$$\frac{P \xrightarrow{\alpha} P'}{P[\vec{a} \leftarrow \vec{b}] \xrightarrow{\alpha[\vec{a} \leftarrow \vec{b}]} P'[\vec{a} \leftarrow \vec{b}]}$$

$$\frac{}{\lambda.P \xrightarrow{\lambda} P} \qquad \frac{P_j \xrightarrow{\alpha} P_j' \quad j \in I}{\sum_{i\in I} P_i \xrightarrow{\alpha} P_j'} \qquad \frac{P \xrightarrow{\alpha} P' \quad A \stackrel{\text{def}}{=} P}{A \xrightarrow{\alpha} P'}$$

$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P \mid Q \xrightarrow{\alpha} P \mid Q'} \qquad \frac{P \xrightarrow{\lambda} P' \quad Q \xrightarrow{\overline{\lambda}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

$$\frac{P \xrightarrow{\alpha} P' \quad a \neq \alpha}{P\backslash a \xrightarrow{\alpha} P'\backslash a} \qquad \frac{P \xrightarrow{\alpha} P'}{P[\vec{a} \leftarrow \vec{b}] \xrightarrow{\alpha[\vec{a} \leftarrow \vec{b}]} P'[\vec{a} \leftarrow \vec{b}]}$$

$$\frac{P_1' \equiv P_1 \quad P_1 \xrightarrow{\alpha} P_2 \quad P_2 \equiv P_2'}{P_1' \xrightarrow{\alpha} P_2'}$$

$$\frac{}{\lambda.P \xrightarrow{\lambda} P} \qquad \frac{P_j \xrightarrow{\alpha} P'_j \quad j \in I}{\sum_{i \in I} P_i \xrightarrow{\alpha} P'_i} \qquad \cancel{\frac{P \xrightarrow{\alpha} P' \quad A \stackrel{\text{def}}{=} P}{A \xrightarrow{\alpha} P'}}$$

$$\cancel{\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q}} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P \mid Q \xrightarrow{\alpha} P \mid Q'} \qquad \frac{P \xrightarrow{\lambda} P' \quad Q \xrightarrow{\overline{\lambda}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

$$\frac{P \xrightarrow{\alpha} P' \quad a \neq \alpha}{P \backslash a \xrightarrow{\alpha} P' \backslash a} \qquad \cancel{\frac{P \xrightarrow{\alpha} P'}{P[\vec{a} \leftarrow \vec{b}] \xrightarrow{\alpha[\vec{a} \leftarrow \vec{b}]} P'[\vec{a} \leftarrow \vec{b}]}}$$

$$\frac{P'_1 \equiv P_1 \quad P_1 \xrightarrow{\alpha} P_2 \quad P_2 \equiv P'_2}{P'_1 \xrightarrow{\alpha} P'_2}$$

Lemma

If $P \xrightarrow{\alpha} P'$ with SYS_{CCS} and $P \equiv Q$ then $Q \xrightarrow{\alpha} Q'$ with SPEC_{CCS} and $P' \equiv Q'$.

RCCS process

$$P, Q \coloneqq \lambda.P \;\Big|\; \sum_{i \in I} P_i \;\Big|\; A \;\Big|\; P \mid Q \;\Big|\; P \backslash a \;\Big|\; P[a \leftarrow b] \;\Big|\; 0$$

RCCS process

$$P, Q \coloneqq \lambda.P \;\Big|\; \sum_{i \in I} P_i \;\Big|\; A \;\Big|\; P \mid Q \;\Big|\; P \backslash a \;\Big|\; P[a \leftarrow b] \;\Big|\; 0$$

$$e \coloneqq \langle i, \lambda, P \rangle \qquad \text{(Memory Events)}$$

$$m \coloneqq \varnothing \;\Big|\; \curlyvee .m \;\Big|\; e.m \qquad \text{(Memory Stacks)}$$

$$T \coloneqq m \triangleright P \qquad \text{(Reversible Thread)}$$

$$R, S \coloneqq T \;\Big|\; R \mid S \;\Big|\; R \backslash a \qquad \text{(RCCS Processes)}$$

$$\frac{i \notin \mathsf{I}(m)}{m \triangleright \lambda.P \xrightarrow{i:\lambda} \langle i, \lambda, 0\rangle.m \triangleright P}\ \text{act.}$$

$$\frac{i \notin \mathsf{I}(m)}{\langle i, \lambda, 0\rangle.m \triangleright P \xrightarrow{i:\lambda}_{*} m \triangleright \lambda.P}\ \text{act.}_{*}$$

$$\frac{}{m \triangleright \lambda.P \xrightarrow{i:\lambda} \langle i, \lambda, 0\rangle.m \triangleright P}\ \text{act.} \quad i \notin \mathrm{I}(m)$$

$$\frac{R \xrightarrow{i:\lambda} R' \qquad S \xrightarrow{i:\overline{\lambda}} S'}{R \mid S \xrightarrow{i:\tau} R' \mid S'}\ \text{syn.}$$

$$\frac{}{\langle i, \lambda, 0\rangle.m \triangleright P \xrightarrow{i:\lambda}_{*} m \triangleright \lambda.P}\ \text{act.}_{*} \quad i \notin \mathrm{I}(m)$$

$$\frac{R \xrightarrow{i:\lambda}_{*} R' \qquad S \xrightarrow{i:\overline{\lambda}}_{*} S'}{R \mid S \xrightarrow{i:\tau}_{*} R' \mid S'}\ \text{syn.}_{*}$$

$$\frac{}{m \triangleright \lambda.P \xrightarrow{i:\lambda} \langle i, \lambda, 0\rangle.m \triangleright P}\ \text{act.} \quad i \notin \mathrm{I}(m)$$

$$\frac{R \xrightarrow{i:\lambda} R' \qquad S \xrightarrow{i:\overline{\lambda}} S'}{R \mid S \xrightarrow{i:\tau} R' \mid S'}\ \text{syn.}$$

$$i \notin \mathrm{I}(S)\ \frac{R \xrightarrow{i:\alpha} R'}{R \mid S \xrightarrow{i:\alpha} R' \mid S}\ \text{com.}_1 \qquad i \notin \mathrm{I}(S)\ \frac{S \xrightarrow{i:\alpha} S'}{R \mid S \xrightarrow{i:\alpha} R' \mid S}\ \text{com.}_2$$

$$i \notin \mathrm{I}(m)\ \frac{}{\langle i, \lambda, 0\rangle.m \triangleright P \xrightarrow{i:\lambda}_* m \triangleright \lambda.P}\ \text{act.}_*$$

$$\frac{R \xrightarrow{i:\lambda}_* R' \qquad S \xrightarrow{i:\overline{\lambda}}_* S'}{R \mid S \xrightarrow{i:\tau}_* R' \mid S'}\ \text{syn.}_*$$

But. . .

what should we do with $m \triangleright (P \mid Q)$?

But. . .

what should we do with $m \triangleright (P \mid Q)$?

“Solution”

Define a structural congruence containing

$$m \triangleright (P \mid Q) \equiv (\curlyvee.m \triangleright P) \mid (\curlyvee.m \triangleright Q)$$

and add it to the system.

But hold on

1. Isn't that mixing $\mathrm{SYS}_{\mathrm{RCCS}}$ and $\mathrm{SPEC}_{\mathrm{RCCS}}$?
2. How do we know it's the right $\equiv$?

But hold on

1. Isn't that mixing $\mathrm{SYS}_{\mathrm{RCCS}}$ and $\mathrm{SPEC}_{\mathrm{RCCS}}$?
   It is, but it's probably fine.
2. How do we know it's the right $\equiv$?
   We don't. How do we know it's the right one for CCS?

Lemma

If $P \xrightarrow{\alpha} P'$ with $\mathrm{SYS}_{\mathrm{CCS}}$ and $P \equiv Q$ then $Q \xrightarrow{\alpha} Q'$ with $\mathrm{SPEC}_{\mathrm{CCS}}$ and $P' \equiv Q'$.

Lemma

If $P \xrightarrow{\alpha} P'$ with $\mathrm{SYS}_{\mathrm{CCS}}$ and $P \equiv Q$ then $Q \xrightarrow{\alpha} Q'$ with $\mathrm{SPEC}_{\mathrm{CCS}}$ and $P' \equiv Q'$.

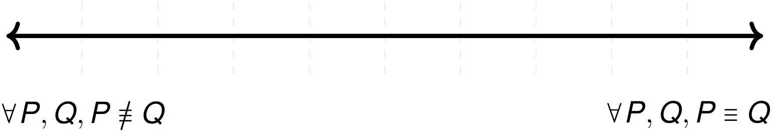
But . . . that's circular!

Lemma

If $P \xrightarrow{\alpha} P'$ with $\mathrm{SYS}_{\mathrm{CCS}}$ and $P \equiv Q$ then $Q \xrightarrow{\alpha} Q'$ with $\mathrm{SPEC}_{\mathrm{CCS}}$ and $P' \equiv Q'$.

But . . . that's circular!

Where are we?



Semantics

$$\forall P, Q, \llbracket P \rrbracket \cong \llbracket Q \rrbracket \iff P \equiv Q$$

Semantics

$\forall P, Q, [\![P]\!] \cong [\![Q]\!] \;\not\Leftrightarrow\; P \equiv Q$

No! In usual models, $[\![P + 0]\!] \cong [\![P]\!]$.

Semantics

$\forall P, Q,\ [\![P]\!] \cong [\![Q]\!] \ \not\Leftrightarrow\ P \equiv Q$

No! In usual models, $[\![P + 0]\!] \cong [\![P]\!]$.

Syntactics

Every term P has a "normal form".

Semantics

$\forall P, Q,\ [\![P]\!] \cong [\![Q]\!] \ \not\Leftrightarrow\ P \equiv Q$

No! In usual models, $[\![P + 0]\!] \cong [\![P]\!]$.

Syntactics

Every term P has a "normal form".

So what?

Semantics

$\forall P, Q, [\![P]\!] \cong [\![Q]\!] \not\Longleftrightarrow P \equiv Q$

No! In usual models, $[\![P + 0]\!] \cong [\![P]\!]$.

Syntactics

Every term P has a "normal form".

So what?

So ... the only thing left is the intuition?