What is the right structural congruence for the (Reversible) Calculus of Communicating Systems?

11th International School on Rewriting

Clément Aubert\textsuperscript{1} Ioana-Domnina Cristescu\textsuperscript{2}

\textsuperscript{1}Augusta University – School of Computer & Cyber Sciences

\textsuperscript{2}INRIA – TAMIS team

Paris, 1-6 July 2019
Goal

Specifying Reversible Concurrent Computation

— What?
Concurrent (multiprocessing, parallel, distributed, etc.) computation that can backtrack. 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.
Introduction

Goal

Specifying Reversible Concurrent Computation

RCCS adds Reversibility to the Calculus of Communicating Systems
CCS System

1 Operators:

\[ P, Q := \lambda. P \mid \sum_{i \in I} P_i \mid A \mid P \mid Q \mid P \setminus a \mid P[a \leftarrow b] \mid 0 \]

2 Labeled Transition System:

\[
\begin{align*}
\alpha & \quad P \xrightarrow{\alpha} P' \\
\alpha & \quad Q \xrightarrow{\alpha} Q' \\
\alpha & \quad P \mid Q \xrightarrow{\alpha} P' \mid Q' \\
\alpha & \quad P \mid Q \xrightarrow{\alpha} Q' \mid P \\
\lambda & \quad P \xrightarrow{\lambda} P' \\
\lambda & \quad Q \xrightarrow{\lambda} Q' \\
\tau & \quad P \mid Q \xrightarrow{\tau} P' \mid Q'
\end{align*}
\]

3 Structural Equivalence:

\[
\begin{align*}
P \mid 0 & \equiv P, \\
P \mid Q & \equiv Q \mid P, \\
P + Q & \equiv Q + P,
\end{align*}
\]

etc.
RCCS System

1 Operators:

\[ T := m \triangleright P \]  (Reversible Thread)
\[ R, S := T \mid R \mid S \mid R \backslash a \]  (RCCS Processes)

2 Labeled Transition System:

\[ m \triangleright \lambda.P \xrightarrow{i: \lambda} \langle i, \lambda, 0 \rangle.m \triangleright P \]
\[ \langle i, \lambda, 0 \rangle.m \triangleright P \xrightarrow{i: \lambda} * \quad m \triangleright \lambda.P' \]

etc.

3 Structural Equivalence:

\[ m \triangleright (P \mid Q) \equiv (\forall.m \triangleright P) \mid (\forall.m \triangleright Q) \]
But hold on

1. Isn’t that mixing the syntactical sugar and the system?
2. How come the congruence does not include e.g. $R \mid S \equiv S \mid R$?
3. How do we know it’s the right $\equiv$?
Lemma

If $P \xrightarrow{\alpha} P'$ with the “pure” LTS and $P \equiv Q$ then $Q \xrightarrow{\alpha} Q'$ with the “sweetened” LTS and $P' \equiv Q'$.

Semantics

\[ \forall P, Q, \llbracket P \rrbracket \equiv \llbracket Q \rrbracket \iff P \equiv Q \]

Syntactics

Every term $P$ has a “normal form”.

Lemma

If \( P \xrightarrow{\alpha} P' \) with the “pure” LTS and \( P \equiv Q \) then \( Q \xrightarrow{\alpha} Q' \) with the “sweetened” LTS and \( P' \equiv Q' \).

Where are we?

\[ \forall P, Q, P \neq Q \quad \forall P, Q, P \equiv Q \]

Semantics

\[ \forall P, Q, \llbracket P \rrbracket \equiv \llbracket Q \rrbracket \iff P \equiv Q \]

Syntactics

Every term \( P \) has a “normal form”.
Lemma

If $P \xrightarrow{\alpha} P'$ with the “pure” LTS and $P \equiv Q$ then $Q \xrightarrow{\alpha} Q'$ with the “sweetened” LTS and $P' \equiv Q'$.

Where are we?

$\forall P, Q, P \not\equiv Q$

$\forall P, Q, P \equiv Q$

Semantics

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

No! Usually, $\llbracket P + 0 \rrbracket \equiv \llbracket P \rrbracket$.

Syntactics

Every term $P$ has a “normal form”.
Lemma

If $P \xrightarrow{\alpha} P'$ with the “pure” LTS and $P \equiv Q$ then $Q \xrightarrow{\alpha} Q'$ with the “sweetened” LTS and $P' \equiv Q'$.

Where are we?

$\forall P, Q, P \neq Q$

$\forall P, Q, P \equiv Q$

Semantics

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

No! Usually, $\llbracket P + 0 \rrbracket \not\simeq \llbracket P \rrbracket$.

Syntactics

Every term $P$ has a “normal form”.

So what?