Comparison of process algebra equivalences

Vashti Galpin

vashti@cs.wits.ac.za

Programme for Highly Dependable Systems
Department of Computer Science
University of the Witwatersrand
South Africa

http://www.cs.wits.ac.za/~vashti

Outline

- introduction and motivation
- what is a process algebra?
- what is a semantic equivalence?
- why is comparison important?
- what is a format?
- how can comparison be done using formats?
- conclusions

Process algebras

- motivation
  - mathematical models
  - specification and verification
  - formal methods
- components
  - syntax
  - operational semantics
  - semantic equivalence
- an example: CCS (Calculus of Communicating Systems)

Syntax

- define processes
- actions: $A \cup \overline{A} \cup \{\tau\}$
- operators: subset of full CCS

$$P \quad ::= \quad 0 \quad | \quad a.P \quad | \quad P + P \quad | \quad P \mid P$$

- examples of processes
  - $a.0$
  - $b.Q$
  - $a.0 + b.0$
  - $a.0 \mid b.0$
  - $a.(b.0 + c.0) \mid d.0$

Operational semantics

- rule sets, describe behaviour of processes formally
- generate labelled transition system: $p \xrightarrow{a} p'$
- rules for subset of CCS

$$\frac{}{a.x \xrightarrow{a} x}$$

$$\frac{x \xrightarrow{a} y}{x + x' \xrightarrow{a} y} \qquad \frac{x \xrightarrow{a} y}{x' + x \xrightarrow{a} y}$$

$$\frac{x \xrightarrow{a} y}{x \mid x' \xrightarrow{a} y \mid x'} \qquad \frac{x \xrightarrow{a} y}{x' \mid x \xrightarrow{a} x' \mid y} \qquad \frac{x \xrightarrow{a} y \quad x' \xrightarrow{\overline{a}} y'}{x \mid x' \xrightarrow{\tau} y \mid y'}$$

Proof of a transition

- tree of rules plus substitutions
- prove: $a.b.0 \mid \overline{a}.0 \xrightarrow{\tau} b.0 \mid 0$
- $\sigma_1(x) = b.0 \quad \sigma_2(x) = 0$
  $\sigma_3(x) = a.b.0 \quad \sigma_3(y) = b.0 \quad \sigma_3(x') = \overline{a} \quad \sigma_3(y') = 0$

$$\sigma_1\left(\frac{}{a.x \xrightarrow{a} x}\right) \quad \sigma_2\left(\frac{}{\overline{a}.x \xrightarrow{\overline{a}} x}\right)$$

$$\sigma_3\left(\frac{x \xrightarrow{a} y \quad x' \xrightarrow{\overline{a}} y'}{x \mid x' \xrightarrow{\tau} y \mid y'}\right)$$

$$\longrightarrow$$

$$\frac{}{a.b.0 \xrightarrow{a} b.0} \qquad \frac{}{\overline{a}.0 \xrightarrow{\overline{a}} 0}$$

$$\frac{a.b.0 \xrightarrow{a} b.0 \quad \overline{a}.0 \xrightarrow{\overline{a}} 0}{a.b.0 \mid \overline{a}.0 \xrightarrow{\overline{a}} b.0 \mid 0}$$

Semantic equivalence

- some notion of similar behaviour
- equivalent?

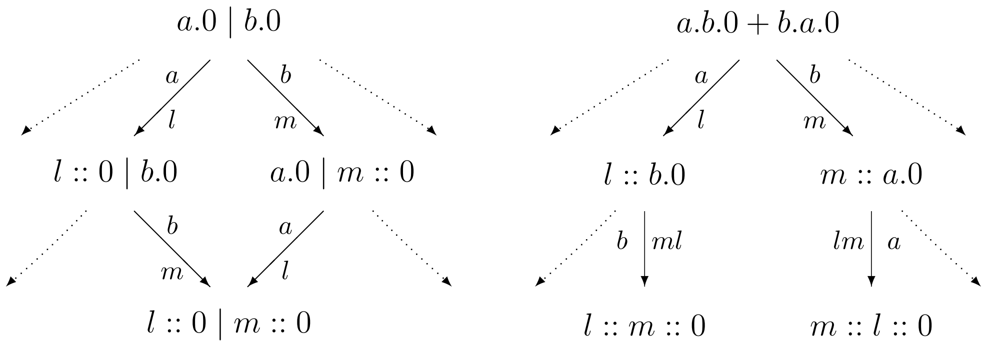
$$\begin{tikzcd}
& a.b.0 + a.c.0 \arrow[dl, "a"'] \arrow[dr, "a"] & \\
b.0 \arrow[d, "b"'] & & c.0 \arrow[d, "c"] \\
0 & & 0
\end{tikzcd}
\qquad
\begin{tikzcd}
& a.(b.0 + c.0) \arrow[d, "a"'] & \\
& b.0 + c.0 \arrow[dl, "b"'] \arrow[dr, "c"] & \\
0 & & 0
\end{tikzcd}$$

- bisimulation — whenever p and q are related
  1. if $p \xrightarrow{a} p'$, there exists q such that $q \xrightarrow{a} q'$ and p' and q' are related
  2. if $q \xrightarrow{a} q'$, there exists p such that $p \xrightarrow{a} p'$ and p' and q' are related
- if there is a bisimulation containing (p, q) then $p \sim q$
- equivalent?

$$\begin{tikzcd}
& a.0 \mid b.0 \arrow[dl, "a"'] \arrow[dr, "b"] & \\
0 \mid b.0 \arrow[dr, "b"'] & & a.0 \mid 0 \arrow[dl, "a"] \\
& 0 \mid 0 &
\end{tikzcd}
\qquad
\begin{tikzcd}
& a.b.0 + b.a.0 \arrow[dl, "a"'] \arrow[dr, "b"] & \\
b.0 \arrow[dr, "b"'] & & a.0 \arrow[dl, "a"] \\
& 0 &
\end{tikzcd}$$

Other semantics

- introduce new operators, new rules, new equivalence
- locations



- want to compare, understand relationship

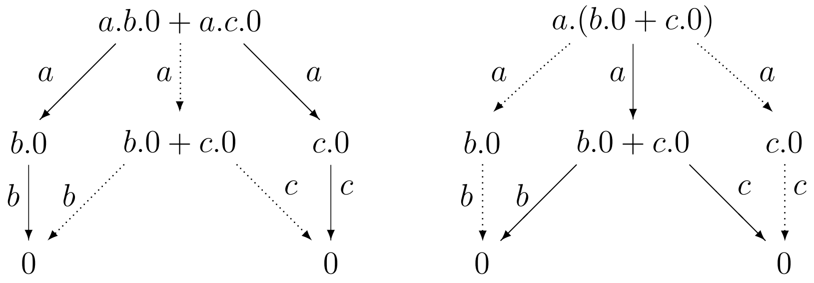
Comparison

- extension
  - combining two rule sets, $R_0 \oplus R_1$
  - compare R_0 with $R_0 \oplus R_1$
- conservative extension — no new transitions
- notation: $\sim_R$ — bisimulation with respect to a rule set R
- abstracting extension up to bisimulation — $\sim_{R_0} \subseteq \sim_{R_0 \oplus R_1}$
- refining extension up to bisimulation — $\sim_{R_0 \oplus R_1} \subseteq \sim_{R_0}$
- capture different semantics by combining rule sets

- add these rules to CCS rules

$$\frac{x \xrightarrow{a} y \qquad x' \xrightarrow{a} y'}{x + x' \xrightarrow{a} y + y'} \qquad \frac{x \xrightarrow{a} y \qquad y \xrightarrow{b} x'}{x \xrightarrow{a} b.x'}$$

- new transitions are added



Formats

- metatheory of process algebra
- reason about rule sets in general
- *tyft/tyxt* format: results about conservative extensions

$$\frac{\{t_i \xrightarrow{a_i} y_i \mid i \in I\}}{f(x_1, \ldots, x_n) \xrightarrow{a} t}$$

- propose new format: extended *tyft/tyxt* format

$$\frac{\{t_i \xrightarrow{\lambda_i} y_i \mid i \in I\}}{f(\eta_1, \ldots, \eta_m, x_1, \ldots, x_n) \xrightarrow{\lambda} t}$$

- conditions on process variables and label variables

Main features of new format

- treats actions syntactically, not schematically
  - allows for more general definition of bisimulation
- uses many-sorted algebras
  - use of different sorts gives power for extension results
- bisimulation is a congruence with respect to operators defined using the format
- example: CCS prefix rule

$$\frac{}{a.x \xrightarrow{a} x} \qquad \text{becomes} \qquad \frac{}{\mathsf{pref}(z,x) \xrightarrow{z} x}$$

Abstracting extension up to bisimulation

- if
  - R_0 and R_1 extended *tyft/tyxt*
  - R_0 pure, label-pure (conditions on variables)
  - R_1 well-founded (condition on premises)
  - $R_0 \oplus R_1$ type-0 (condition on function symbol and sort of label in rule conclusion)
- then $\sim_{R_0} \subseteq \sim_{R_0 \oplus R_1}$
- proof
  - define relation over processes, show bisimulation
  - given proof of a transition, modify substitutions to show matching transition
  - induction on depth of proof, induction on variables in premises

Refining extension up to bisimulation

- if
  - R_0 and R_1 extended *tyft/tyxt*
  - R_0 pure, label-pure (conditions on variables)
  - $R_0 \oplus R_1$ type-1 (condition on function symbol and sort of label in rule conclusion)
- then $\sim_{R_0 \oplus R_1} \subseteq \sim_{R_0}$
- proof
  - lemma: transition proved from $R_0 \oplus R_1$ with last rule from R_0, transition can be proved from R_0
  - contrapositive
  - show no added transition can 'fix' non-equivalent processes

Applications, further work and conclusions

- applications
  - use to express process algebras
  - new result: pomset bisimulation is a proper subset of n-multiprocessor bisimulation
- further work
  - comparison with other recent formats: Bernstein, Ferrari and Montanari, Fokkink and Verhoef
  - open questions
- conclusions