A new format for process algebras

Vashti Galpin

vashti@cs.wits.ac.za

Department of Computer Science, University of the Witwatersrand

and

Department of Computer Science, University of Edinburgh

Outline and introduction

- process algebras
  - syntax, operational semantics, equivalence semantics
  - examples—CCS, extension to CCS
  - examples of bisimilar processes
- formats
  - existing formats and results
  - new format
    - justification
    - definition
    - congruence result
- conclusions

Process algebras

- concurrency + interaction
- components
  - syntax
  - operational semantics—define labelled transition system, proofs of transitions
  - equivalence semantics—equate processes with same behaviour
- examples
  - CCS
  - CSP
  - ACP
  - extensions to CCS—location, distribution, causality

CCS and its extensions

- syntax
  - $P ::= \mathrm{nil} \mid \alpha.P \mid P + P \mid P|P \mid P \backslash L \mid P[f]$
  - $\alpha \in \{a, b, c, \ldots, \overline{a}, \overline{b}, \overline{c}, \ldots\} \cup \{\tau\}$
  - $L \subset \{a, b, c, \ldots, \overline{a}, \overline{b}, \overline{c}, \ldots\}$
- operational semantics

$$\frac{}{\alpha.P \xrightarrow{\alpha} P} \qquad \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \qquad \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q}$$

- equivalence semantics, bisimulation—$P \sim Q$ iff for all α
  1. whenever $P \xrightarrow{\alpha} P'$, there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $P' \sim Q'$
  2. whenever $Q \xrightarrow{\alpha} Q'$, there exists P' such that $P \xrightarrow{\alpha} P'$ and $P' \sim Q'$

Extensions to CCS

- use additional information to capture characteristics of concurrency
- example—adding location information
  - new syntax: $l :: P$ where $l \in Loc$ disjoint from $\{a, b, c, \ldots, \overline{a}, \overline{b}, \overline{c}, \ldots\} \cup \{\tau\}$
  - new rules for operational semantics, $u \in Loc^*$

$$\frac{}{\alpha.P \xrightarrow{(\alpha,l)} l :: P} \qquad \frac{P \xrightarrow{(\alpha,u)} P'}{P + Q \xrightarrow{(\alpha,u)} P'} \qquad \frac{P \xrightarrow{(\alpha,u)} P'}{P|Q \xrightarrow{(\alpha,u)} P'|Q} \qquad \frac{P \xrightarrow{(\alpha,u)} P'}{l :: P \xrightarrow{(\alpha,lu)} l :: P'}$$

  - new labelled transition system: $\xrightarrow{(\alpha,u)}$
  - new equivalence: bisimulation matches on both action and location
  - example of non-interleaving equivalence

Examples

$$\begin{tikzcd}
& a.b.\,\mathrm{nil} + b.a.\,\mathrm{nil} \arrow[dl, "a"'] \arrow[dr, "b"] & \\
b.\,\mathrm{nil} \arrow[dr, "b"'] & & a.\,\mathrm{nil} \arrow[dl, "a"] \\
& \mathrm{nil} &
\end{tikzcd}
\quad \sim \quad
\begin{tikzcd}
& a.\,\mathrm{nil} \mid b.\,\mathrm{nil} \arrow[dl, "a"'] \arrow[dr, "b"] & \\
\mathrm{nil} \mid b.\,\mathrm{nil} \arrow[dr, "b"'] & & a.\,\mathrm{nil} \mid \mathrm{nil} \arrow[dl, "a"] \\
& \mathrm{nil} \mid \mathrm{nil} &
\end{tikzcd}$$

$$\begin{tikzcd}
& a.b.\,\mathrm{nil} + b.a.\,\mathrm{nil} \arrow[dl, "{(a,l)}"'] \arrow[dr, "{(b,m)}"] & \\
l :: b.\,\mathrm{nil} \arrow[d, "{(b,lm)}"'] & & m :: a.\,\mathrm{nil} \arrow[d, "{(a,ml)}"] \\
l :: m :: \mathrm{nil} & & m :: l :: \mathrm{nil}
\end{tikzcd}
\quad \not\sim_l \quad
\begin{tikzcd}
& a.\,\mathrm{nil} \mid b.\,\mathrm{nil} \arrow[dl, "{(a,l)}"'] \arrow[dr, "{(b,m)}"] & \\
l :: \mathrm{nil} \mid b.\,\mathrm{nil} \arrow[dr, "{(b,m)}"'] & & a.\,\mathrm{nil} \mid m :: \mathrm{nil} \arrow[dl, "{(a,l)}"] \\
& l :: \mathrm{nil} \mid m :: \mathrm{nil} &
\end{tikzcd}$$

Formats

- meta-theory of process algebras, deals with rules for operational semantics
- congruence results—a semantic equivalence is a congruence for an operator **op** if

$$\forall 1 \leqslant i \leqslant n,\ \ P_i \sim Q_i \Rightarrow \mathbf{op}(P_1, \ldots, P_n) \sim \mathbf{op}(Q_1, \ldots, Q_n)$$

- number of existing formats—De Simone, GSOS, *tyft/tyxt*, *ntyft/ntyxt*, *panth*
- *tyft/tyxt* format
  - single-sorted signature with standard definition of open terms, closed terms and substitutions, and notion of proof
  - rules have a specific form: y_i's, x_j's and x distinct variables, t_i's and t open terms

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

  - given a signature, a set of rules in *tyft/tyxt* format then bisimulation is a congruence for all operators

A new format

- why?
  - extensions to CCS have structured/non-atomic labels
  - schematic approach no longer works
  - interleaving is broken by passing action information into processes
  - require more general definition of bisimulation—work with equivalences over labels
- extended *tyft/tyxt* format
  - many-sorted signature with distinguished sort for process terms P, plus condition

$$\mathbf{op} : s_1, \ldots : s_n \to s,\ \ s \neq \mathsf{P} \Rightarrow s_i \neq \mathsf{P}\ \ \forall 1 \leqslant i \leqslant n$$

  - terms that have sort other than P can only appear as labels
  - similar notions of open terms, closed terms, substitutions and proofs

– rule format

$$\frac{\{p_i \xrightarrow{\lambda_i} y_i \mid i \in I\}}{f(\eta_1, \ldots, \eta_m, x_1, \ldots, x_n) \xrightarrow{\lambda} p} \quad \text{or} \quad \frac{\{p_i \xrightarrow{\lambda_i} y_i \mid i \in I\}}{x \xrightarrow{\lambda} p}$$

* y_i's, x_j's and x distinct variables of sort P
* p_i's and p open terms of sort P
* η_k's, λ_i's and λ open terms of sort other than P
* conditions on variables of sort other than P that appear in open terms

• work with more general bisimulation definition

• assume $\equiv$ is a congruence over closed terms with sort other than P, then $P \sim_{\equiv} Q$ iff for all closed terms λ

1. whenever $P \xrightarrow{\lambda} P'$, there exists Q' and λ' such that $Q \xrightarrow{\lambda'} Q'$, $\lambda \equiv \lambda'$ and $P' \sim_{\equiv} Q'$
2. whenever $Q \xrightarrow{\lambda} Q'$, there exists P' and λ' such that $P \xrightarrow{\lambda'} P'$, $\lambda \equiv \lambda'$ and $P' \sim_{\equiv} Q'$

Congruence result

• given a many-sorted signature and a set of rules that are well-founded, compatible with $\equiv$, then bisimulation with respect to $\equiv$ is a congruence for all operators

$$\forall 1 \leqslant k \leqslant m,\ \mu_k \equiv \nu_k, \quad \forall 1 \leqslant j \leqslant n,\ u_j \sim_{\equiv} v_j \Rightarrow$$
$$\mathbf{op}(\mu_1, \ldots, \mu_m, u_1, \ldots, u_n) \sim_{\equiv} \mathbf{op}(\nu_1, \ldots, \nu_m, v_1, \ldots, v_n)$$

• proof sketch

– define a relation containing the processes under consideration and prove it is a bisimulation

– for each pair in relation, consider transitions from each process and use induction on the depth of the proof of transitions

– this involves finding a new substitution to generate a proof that a matching transition exists

– technical details relate to ensuring that a well-defined substitution can be found

Conclusions

- have shown that congruence holds for new format
- more syntactic approach
- new format can express extensions to CCS and CCS
- can use to compare equivalences on different process algebras
- somewhat less direct