Read Operators and their Expressiveness In Process Algebra

Speaker: Maria Rita Di Berardini
dip. di Matematica e Informatica, Università di Camerino

F. Corradini
dip. di Matematica e Informatica, Università di Camerino

W. Vogler
Institut für Informatik, Universität Augsburg

Paco Meeting, Camerino 15 Settembre 2010
Non-blocking/non-consuming reading operations

**Motivating idea**: provide a way of modelling reading of resources without consuming them – multiple concurrent accesses to the same resource

Non-blocking reading is known from Petri nets in the form of read arcs

\[ t_1 \rightarrow r \leftarrow t_2 \]
Non-blocking/non-consuming reading operations

Read arcs:

• have been used to model a variety of applications (transaction serialisability, concurrent constraint programming, cryptographic protocols, ...)

• add relevant expressivity (the MUTEX problem cannot be solved without read arcs)

Also note that non-blocking reading recall the notion of persistence that exists e.g. in several calculi for describing and analysing security protocols.
In this paper

We enhance PAFAS (a process algebra for asynchronous timed concurrent systems) with non-blocking reading

• read-action prefix operator (flexible, complex semantics) – $\text{PAFAS}_r$
• read-set prefix operator (simpler semantics, but with syntactic restrictions) – $\text{PAFAS}_s$

We also study the expressiveness of these read operators:

• added expressivity of read-action prefixes (w.r.t. fair behaviour)
• compare the expressivity of $\text{PAFAS}_s$ with those of $\text{PAFAS}_r$ and Petri nets with read-arcs
• it is still an open problem if $\text{PAFAS}_r$ is more expressive than $\text{PAFAS}_s$
A Process Algebra for Faster Asynchronous systems

Basic Assumption:

- Actions have an upper time bound (1 or 0) as a maximal delay
- Patient prefixes: $\alpha.P$ (delay 1), $\alpha$ is either a visible action or $\tau$
- Urgent prefixes: $\alpha.P$ (delay 0)
- Patient processes: $a.P \not\rightarrow$ but $a.P \parallel a \cdot \text{nil} \rightarrow a.P \parallel a \cdot \text{nil}$

PAFAS

\[
\begin{align*}
P & ::= \text{nil} \mid x \mid \alpha.P \mid P + P \mid P \parallel_A P \mid P[\Phi] \mid \text{rec } x.P \\
Q & ::= \text{nil} \mid \alpha.P \mid Q + Q \mid Q \parallel_A Q \mid Q[\Phi] \mid \text{rec } x.Q
\end{align*}
\]

Initial processes: $P, P', ...$ General processes $Q, Q', ...$
**Transitional Semantics of PAFAS 1/2**

**Functional Behaviour:** \( Q \xrightarrow{\alpha} Q' \) – \( Q \) evolves in \( Q' \) by performing \( \alpha \)

\[
\begin{align*}
\mu &\in \{\alpha, \alpha\} \\
\mu.P &\xrightarrow{\alpha} P \\
\alpha \notin A &\quad Q_1 \xrightarrow{\alpha} Q'_1 \\
Q_1 \parallel_A Q_2 &\xrightarrow{\alpha} Q'_1 \parallel_A Q_2 \\
\alpha \in A &\quad Q_1 \xrightarrow{\alpha} Q'_1 \quad Q_2 \xrightarrow{\alpha} Q'_2 \\
Q_1 \parallel_A Q_2 &\xrightarrow{\alpha} Q'_1 \parallel_A Q'_2
\end{align*}
\]

Other rules are as expected
Temporal Behaviour: $Q$ let pass 1 unit of time and becomes $Q'$

- $Q \xrightarrow{X} Q'$
  - conditional time step of duration 1
  - $X \subseteq A$ (visible actions) is a refusal set

\[
\begin{align*}
\alpha.P &\xrightarrow{X} \alpha.P \\
\alpha.P &\xrightarrow{X} \alpha.P \\
Q_i &\xrightarrow{X} Q'_i \\
Q_1 + Q_2 &\xrightarrow{X} Q'_1 + Q'_2 \\
Q_i &\xrightarrow{X_i} Q'_i \\
X &\subseteq \left( A \cap (X_1 \cup X_2) \right) \cup ((X_1 \cap X_2) \setminus A) \\
Q_1 \parallel_A Q_2 &\xrightarrow{\alpha} Q'_1 \parallel_A Q'_2
\end{align*}
\]

other rules are as expected
**Temporal Behaviour**: $Q$ let pass 1 unit of time and becomes $Q'$

- $Q \xrightarrow{X} r Q'$
  - conditional time step of duration 1
  - $X \subseteq \mathbb{A}$ (visible actions) is a refusal set

Whenever $Q \xrightarrow{A} r Q'$, we write $Q \xrightarrow{1} Q'$ and call it a **1-step**
Adding non-blocking reading I ($\text{PAFAS}_r$)

**PAFAS +**

Read-action prefix operator

$$\mu \triangleright Q$$ where $\mu \in \{\alpha, \alpha\}$

behave as $Q$ but can also been read with $\alpha$ (**read action**)

since being read does not change the state, $\alpha$ can be performed repeatedly

The operational semantics of $\mu \triangleright Q$ needs two type of transition relations
Nested reading behaviours

**Ordinary action transitions:** \( Q \xrightarrow{\alpha} Q' \)

**Read action transitions:** \( Q \xrightarrow{\alpha} Q' \) (due to nested read actions)

Basically:

- \( P = a \triangleright (b \triangleright c + d) \xrightarrow{a} P \)
- \( P' = b \triangleright c + d \xrightarrow{b} P' \) and \( P = a \triangleright P' \xrightarrow{b} P \) 
  performing a read action does not change the state, also in a choice context
- \( P' = b \triangleright c + d \xleftarrow{c} \text{nil} \) and \( P = a \triangleright P' \xleftarrow{c} \text{nil} \) 
  (similarly for the ordinary \( d \))

\( \xrightarrow{\alpha} = \xrightarrow{\alpha} \cup \xrightarrow{\alpha} \)
Transitional Semantics of $\mu \triangleright Q$

**Functional behaviour**

$$\frac{\mu \in \{\alpha, \bar{\alpha}\}}{\mu \triangleright Q \overset{\alpha}{\rightarrow} \mu \triangleright Q}$$

$$\frac{Q \overset{\alpha}{\rightarrow} Q'}{\mu \triangleright Q \overset{\alpha}{\rightarrow} \mu \triangleright Q'}$$

$$\frac{Q \overset{\alpha}{\leftrightarrow} Q'}{\mu \triangleright Q \overset{\alpha}{\leftrightarrow} \mu \triangleright Q'}$$

**Temporal behaviour**

$$\frac{Q \overset{X}{\rightarrow_r} Q'}{\alpha \triangleright Q \overset{X}{\rightarrow_r} \alpha \triangleright Q'}$$

$$\frac{Q \overset{X}{\rightarrow_r} Q' \quad \alpha \notin X \cup \{\tau\}}{\alpha \triangleright Q \overset{X}{\rightarrow_r} \alpha \triangleright Q'}$$
Adding non-blocking reading II (PAFAS$_s$)

PAFAS + Read-set prefix operator

\[ \{\mu_1, \ldots, \mu_n\} \triangleright Q \]

\(\{\mu_1, \ldots, \mu_n\}\) contains all read actions currently enabled

We try to avoid nested reading

<table>
<thead>
<tr>
<th>{a, b} \triangleright c</th>
<th>a \triangleright (b \triangleright c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>{a, b} \triangleright (c + d)</td>
<td>a \triangleright (b \triangleright c + d)</td>
</tr>
</tbody>
</table>

Functional behaviour: \(\alpha \mapsto \alpha\) (\(= \alpha \mapsto \alpha\))

\[ \mu_i \in \{\alpha, \alpha\} \]

\[ \{\mu_1, \ldots, \mu_n\} \triangleright Q \overset{\alpha}{\leftrightarrow} Q' \]

\[ Q \overset{\alpha}{\leftrightarrow} Q' \]

\[ \{\mu_1, \ldots, \mu_n\} \triangleright Q \overset{\alpha}{\leftrightarrow} Q' \]
Proper terms = read-proper + rec-proper

Not all PAFAS\textsubscript{s} terms have a reasonable semantics:

- \( P_1 = \{a\} \triangleright (\{b\} \triangleright c) \leftrightarrow \{b\} \triangleright c \) instead of \( \{a\} \triangleright (\{b\} \triangleright c) \)
- \( P_2 = (\{b\} \triangleright c) + a \leftrightarrow \{b\} \triangleright c \) instead of \( \{a\} \triangleright (\{b\} \triangleright c) \)

We only consider read-proper terms:

1. for all subterms \( \{\mu_1, \ldots, \mu_n\} \triangleright Q_1, Q_1 \) is read-guarded
2. all subterm \( Q_1 + Q_2 \) are read-guarded

where read-guarded = read-set prefix operators are in the scope of some action-prefix: \( a.\{b\} \triangleright c \) but not \( \{b\} \triangleright c \) or \( \{b\} \triangleright c + a \)
Proper terms = read-proper + rec-proper

Not all \( \text{PAFAS}_s \) terms have a reasonable semantics:

- \( P_1 = \{a\} \triangleright (\{b\} \triangleright c) \stackrel{b}{\leftrightarrow} \{b\} \triangleright c \) instead of \( \{a\} \triangleright (\{b\} \triangleright c) \)
- \( P_2 = (\{b\} \triangleright c) + a \stackrel{b}{\leftrightarrow} \{b\} \triangleright c \) instead of \( \{a\} \triangleright (\{b\} \triangleright c) \)

We only consider \textbf{read-proper} terms:

1. for all subterms \( \{\mu_1, \ldots, \mu_n\} \triangleright Q_1, Q_1 \) is read-guarded
2. all subterm \( Q_1 + Q_2 \) are read-guarded

where \textbf{read-guarded} = read-set prefix operators are in the scope of some action-prefix: \( a.\{b\} \triangleright c \) but not \( \{b\} \triangleright c \) or \( \{b\} \triangleright c + a \)
Proper terms $= \text{read-proper} + \text{rec-proper}$

$P_3 = \text{rec } x.{\{a\} \triangleright b.(c + x)}$ is read-proper, but $P_3 \xrightarrow{b} c + P_3$ that is not read-proper because of $P_3$

$P_3$ is not rec-proper:

1. $\{a\} \triangleright b.(c + x)$ is not read-guarded, and
2. $x$ is not guarded in $c + x$
Proper terms = read-proper + rec-proper

\[ P_3 = \text{rec } x. \{a\} \triangleright b.(c + x) \text{ is read-proper, but } P_3 \xrightarrow{b} c + P_3 \text{ that is not read-proper because of } P_3 \]

\[ P_3 \text{ is not rec-proper:} \]

1. \( \{a\} \triangleright b.(c + x) \) is not read-guarded, and
2. \( x \) is not guarded in \( c + x \)
(Weak) **Fairness of Actions**: an action must be performed whenever countinuosly enabled in a run

**Fair Traces**: sequence of visible actions that occur in a transition sequence with infinitely many 1-step

\[
(a \triangleright b).\text{nil}
\]

\[
\text{FairL}(a \triangleright b) = \{a^i b \mid i \geq 0\}
\]

**Theorem**

If \( P \) is a finite state process with **no read-prefixes** and \( \text{sort}(P) = \{a, b\} \) then \( \text{FairL}(P) \neq \{a^i b \mid i \geq 0\} \)
Mapping $\text{PAFAS}_s$ into $\text{PAFAS}_r$

\[[Q]_s\] is $Q$ where each subterm $\{\mu_1, \ldots, \mu_n\} \triangleright Q_1$ is replaced with $\mu_1 \triangleright \ldots \triangleright \mu_n \triangleright [Q_1]_r$

\[
\left[\{a\} \triangleright (\tau.\{b, c\} \triangleright d \parallel \emptyset e)\right]_s = a \triangleright (\tau. b \triangleright c \triangleright d \parallel \emptyset e)
\]

**Theorem**
For all proper $Q$ in $\text{PAFAS}_s$, $Q$ and $[Q]_r$ are (timed) bisimilar
$P_0 = t_1.P_1 + t_2.P_1$

$P_1 = \{t_5, t_6\} \triangleright (t_3.P_0 + t_4.P_0)$
• We first exhibit a subset of PAFAS$_r$ terms that have an easy translation in PAFAS$_s$ (terms in Read Normal Form, RNF)
• We also discuss how terms that are not in RNF can be normalised and the problems of such a normalisation
Terms in RNF

The set of terms in RNF is the image of the mapping function $\llbracket - \rrbracket_r$

We only consider (sub)terms in $\text{PAFAS}_r$

1. $\mu_1 \triangleright \cdots \triangleright \mu_n \triangleright Q_1$ where $Q_1$ is read-guarded
2. $Q_1 + Q_2$ read-guarded
3. $\text{rec } x.Q$ where $Q$ is rec-proper (as above)

where read-guarded = all read-action prefix operators are in the scope of some action-prefix: $a.b \triangleright c$ but not $b \triangleright c$ or $b \triangleright c + a$
Translating terms in RNF

\([\_]_s\): terms in RNF \(\rightarrow\) proper PAFAS\(_s\)-terms

\([Q]\_s\) is \(Q\) where each subterm \(\mu_1 \triangleright \ldots \triangleright \mu_n \triangleright Q_1\) is replaced with \(\{\nu_1, \ldots, \nu_k\} \triangleright [Q_1]_s\) (here \(\{\nu_1, \ldots, \nu_k\}\) is a “proper” read-set)

\([a \triangleright b \triangleright a \triangleright Q]_s = \{a, b\} \triangleright [Q]_s\)

\([a \triangleright b \triangleright a \triangleright Q]_s = \{a, b\} \triangleright [Q]_s\)

**Theorem**
For each \(Q\) in RNF, \(Q\) and \([Q]_s\) are (timed) bisimilar
Terms not in RNF

Basic Idea

Use laws to rewrite a term not in RNF into a bisimilar one in RNF

Ex: \((a ⊳ b) + c \sim a ⊳ (b + c)\)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>(\mu ⊳ (\nu ⊳ Q) \sim \nu ⊳ (\mu ⊳ Q))</td>
</tr>
<tr>
<td>L2</td>
<td>(\alpha ⊳ (\mu ⊳ Q) \sim \mu ⊳ Q, \alpha ⊳ (\mu ⊳ Q) \sim \alpha ⊳ Q) provided that (\mu \in {\alpha, \alpha})</td>
</tr>
<tr>
<td>L3</td>
<td>((\mu ⊳ Q) + R \sim \mu ⊳ (Q + R))</td>
</tr>
<tr>
<td>L4</td>
<td>(a ⊳ (Q_1 \parallel_A Q_2) \sim ((a ⊳ Q_1) \parallel_A {a} (a ⊳ Q_2)), a ⊳ (Q_1 \parallel_A Q_2) \sim ((a ⊳ Q_1) \parallel_A {a} (a ⊳ Q_2))) prov. that (a \notin \text{sort}(Q))</td>
</tr>
<tr>
<td>L5</td>
<td>((\alpha ⊳ Q)[\Phi] \sim \Phi(\alpha) ⊳ (Q[\Phi]), (\alpha ⊳ Q)[\Phi] \sim \Phi(\alpha) ⊳ (Q[\Phi]))</td>
</tr>
<tr>
<td>L6</td>
<td>((Q[\Phi])[\Psi] \sim Q[\Psi \circ \Phi])</td>
</tr>
<tr>
<td>L7</td>
<td>(\text{rec } x.Q \sim Q{\text{rec } x.Q/x})</td>
</tr>
</tbody>
</table>
The idea of the translation into RNF is to perform rewriting by induction on the term size:

- action-prefix, parallel composition and relabeling preserve RNF
- read-prefixes $\mu \triangleright Q$ can be dealt with distributing $\mu$ among $Q$’s components

Ex: $a \triangleright (b \triangleright c \parallel \emptyset d) \sim a \triangleright b \triangleright c \parallel a \triangleright d$

- choice and recursion still pose an unsolved problem
Normalising choices

How to normalise $Q + (R_1 \parallel_A R_2)$?

(1) if $Q$ is deterministic (Ex: $Q = a.b + c.d$) we can use the law

$$Q + (R_1 \parallel_A R_2) \sim (Q + R_1) \parallel_{A \cup \text{sort}(Q)} (Q + R_2)$$

But if $Q = a.b + a.d$ (non-deterministic):

- $(Q + R_1) \parallel_{\{a,b,c,d\}} (Q + R_2) \xrightarrow{a} b \parallel_{\{a,b,c,d\}} d$ while

- $Q + (R_1 \parallel_A R_2) \xrightarrow{a} b$ or in $c$
Normalising choices

How to normalise $Q + (R_1 \ parallel_A R_2)$?

(2) replace the secon copy of $Q$ with the “top-part” of $Q$

The top-part of $Q = a.b + a.d$ is $a$ and

$$Q + (R_1 \ parallel_A R_2) \sim (Q + R_1) \ parallel_{A \cup \{a\}} (a + R_2)$$

The top-part of $Q = a \rhd b.c$ is $a \rhd b$ and

$$Q + (R_1 \ parallel_A R_2) \sim (Q + R_1) \ parallel_{A \cup \{a,b\}} (a \rhd b + R_2)$$
We need a suitable expansion law

But, what is the top-part of $Q = a \parallel \emptyset b$? (note that $Q \not\sim a.b + b.a$)

- $Q \xrightarrow{1} a \parallel \emptyset b \xrightarrow{a} \text{nil} \parallel \emptyset b$,
- $a.b + b.a \xrightarrow{1} a.b + b.a \xrightarrow{a} b$

To solve this problem, a proper expansion law is needed

But even for the standard PAFAS this law is unknown
Conclusion and Future Work

We have studied two different ways to enhance PAFAS with non-blocking reading actions.

We also study the expressiveness of these read operators:

- it is still an open problem if PAFAS$_r$ is more expressive than PAFAS$_s$.

In the future:

- we will try to complete this translation; this is related to finding an expansion law for generic PAFAS$_r$ (and PAFAS) terms.
- This expansion law should also provide us with an axiomatisation for the full PAFAS language.
- We plan to use read prefixes for modelling systems and comparing their efficiency or proving them correct under the progress assumption.
Thank you for your attention