Performance Evaluation with PAFAS

State of the Art and Ongoing Work

Speaker: M. R. Di Berardini[1]

[1]Dip. di Matematica e Informatica, Università di Camerino

Paco Meeting, L'Aquila 2-3 marzo 2010

Contents

A Process algebra for Faster Asynchronous Systems

The set of processes is generated by

$$P ::= \mathsf{nil} \mid x \mid \alpha.P \mid \underline{\alpha}.P \mid P + P \mid P \|_A P \mid P[\Phi] \mid \mathsf{rec}\, x.P$$

where α is a basic action (either visible or τ), $A \subseteq \mathbb{A}$ and Φ is a relabeling function

We distinguish between:

- patient actions: actions with time boud 1 - α
- urgent actions: actions with time boud 0 - $\underline{\alpha}$

Our processes are patient

$$\underline{a}.P \stackrel{1}{\not\rightarrow} \qquad \text{but} \qquad \underline{a}.P \,\|_{\{a\}}\, a.\mathsf{nil} \xrightarrow{1} \underline{a}.P \,\|_{\{a\}}\, \underline{a}.\mathsf{nil}$$

Transitional Semantics of PAFAS

- Functional Behaviour

  $P \xrightarrow{\alpha} Q$ P evolves into Q performing the action α

- Refusal Behaviour

  $P \xrightarrow{X}_r Q$ It is a conditional time step of duration 1
  X is a set of actions that are not just waiting for a synchronization

  $P \xrightarrow{1} Q$ whenever $P \xrightarrow{\mathbb{A}}_r Q$ - this is a full time step

The Functional Behaviour of PAFAS-terms

$$\textsc{Act}_p \frac{}{\alpha.P \xrightarrow{\alpha} P} \qquad \textsc{Act}_u \frac{}{\underline{\alpha}.P \xrightarrow{\alpha} P} \qquad \textsc{Sum} \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}$$

$$\textsc{Synch} \frac{\alpha \in A,\ P \xrightarrow{\alpha} P',\ Q \xrightarrow{\alpha} Q'}{P \|_A Q \xrightarrow{\alpha} P' \|_A Q'} \qquad \textsc{Par} \frac{\alpha \notin A,\ P \xrightarrow{\alpha} P'}{P \|_A Q \xrightarrow{\alpha} P' \|_A Q}$$

The other rules are as expected

The Refusal Behaviour of PAFAS-terms

$$\textsc{Nil}_r \frac{}{\mathsf{nil} \xrightarrow{X}_r \mathsf{nil}} \qquad \textsc{Act}_{r1} \frac{}{\alpha.P \xrightarrow{X}_r \underline{\alpha}.P} \qquad \textsc{Act}_{r2} \frac{\alpha \notin X \cup \{\tau\}}{\underline{\alpha}.P \xrightarrow{X}_r \underline{\alpha}.P}$$

$$\textsc{Sum}_r \frac{P \xrightarrow{X}_r P', Q \xrightarrow{X}_r Q'}{P + Q \xrightarrow{X}_r P' + Q'}$$

$$\textsc{Par}_r \frac{P \xrightarrow{X_1}_r P', Q \xrightarrow{X_2}_r Q', X \subseteq (A \cap (X_1 \cup X_2)) \cup ((X_1 \cap X_2) \backslash A)}{P \|_A Q \xrightarrow{X}_r P' \|_A Q'}$$

The other rules are as expected

Notation:

- The set $\mathrm{DL}(P) = \{v \in (\mathbb{A}_\tau \cup \{1\})^* \mid P \overset{v}{\Longrightarrow}\}$ contains the **discrete traces** of P
- The set $\mathrm{RT}(P) = \{v \in (\mathbb{A}_\tau \cup 2^{\mathbb{A}})^* \mid P \overset{\mu}{\Longrightarrow}_r\}$ contains the **refusal traces** of P
- The **refusal transition system** $\mathrm{RTS}(P)$ of P is an arc-labelled graph, an arc being a transition $\xrightarrow{\alpha}$ or $\xrightarrow{X}_r$

Performance Measures

Based on PAFAS, we provide two performance measures:

1. A faster-than preorder relation
   This is a timed variation of the testing preorder
   P is faster than P if it is successful in more environments than Q, or in the same environments within a shorter time
2. A performance function
   Gives the worst-case time a process need to satisfy a given user behaviour (test)

The Testing Preorder

- A **timed test** is a pair (O, D) where:
  - O is a test process (perform the success action ω)
  - $D \in \mathbb{N}_0$ is a time bound
- P **must** (O, D) if for each $v \in \mathrm{DL}(P \| O)$ with duration $\zeta(v) > D$ contains some ω
- $P \sqsupseteq Q$ if, for each (O, D), Q **must** (O, D) implies P **must** (O, D)

Theorem (Characterization of the testing preorder):

$$P \sqsupseteq Q \text{ if and only if } \mathrm{RT}(P) \subseteq \mathrm{RT}(Q)$$

Performance Function

The *performance function* $p(P, O)$ gives the worst-case time the process P needs to satisfy O

$$p(P, O) = \sup\{\zeta(v) \mid v \in \mathrm{DL}(P \parallel O) \text{ and } v \text{ does not contain } \omega\}$$

Proposition (Quantitative formulation of the testing preorder):

$$P \sqsupseteq Q \text{ if and only if } p(P, O) \leq p(Q, O) \text{ for all tests } O$$

Response Performance

- Consider

$$\text{Seq} = \text{rec}\, x.\ \underline{in}.\tau.out.x, \text{ and}$$

$$\text{Pipe} = (\text{rec}\, x.\ \underline{in}.s.x \parallel_{\{s\}} \text{rec}\, x.\ s.out.x)/s$$

- One would expect that Pipe is faster than Seq, but this is not true
- Pipe **must** $(in.in.\underline{\omega}, 1)$, but Seq $\not\!\!\textbf{must}(in.in.\underline{\omega}, 1)$
- The expectation that Pipe is faster than Seq is based on some assumption about the users
- We want to compare these processes w.r.t. their ability to answer a given number of requests as fast as possible

Response Performance

We compare processes w.r.t. the class of users $\mathcal{U} = \{U_n \mid n \geq 1\}$

$$U_1 = \underline{in}.\underline{out}.\underline{\omega}$$
$$U_{n+1} = U_n \parallel_\omega \underline{in}.\underline{out}.\underline{\omega}$$

Given this class of users, the **response performance function** is a function $rp_P : \mathbb{N} \longrightarrow \mathbb{N}_0 \cup \{\infty\}$ such that

$$rp_P(n) = \sup\{\zeta(v) \mid v \in \mathrm{DL}(P \parallel U_n) \text{ and } v \text{ does not contain } \omega\}$$

A process P is a **response process** if:

- it can only perform *in* and *out* as visible actions,
- it is always able to perform the required number of *out* actions and never performs too many

Ex: $\mathrm{Seq} = \mathrm{rec}\ x.\ \underline{in}.\tau.out.x$, but also Pipe

Response Performance Calculation

In [CV2005], it ha been shown how to calcate the response performance for a response process starting from the refusal transition system RTS(P)

- Remove from RTS(P) all conditional time steps (and all processes that are not reachable anymore) that cannot partecipate in a full time when considering the behavior of $P \parallel U_n$ – rRTS(P)

Response Performance Calculation

In [CV2005], it ha been shown how to calcate the response performance for a response process starting from the refusal transition system RTS(P)

- Remove from RTS(P) all conditional time steps (and all processes that are not reachable anymore) that cannot partecipate in a full time when considering the behavior of $P \parallel U_n$ – rRTS(P)
- Consider a special subset of paths in rRTS(P) called n-critical paths $v \in DL(P \parallel U_n)$ that does not contain ω iff there is an n-critical path w of P such $\zeta(v) = \zeta(w)$

$$
\begin{aligned}
rp_P(n) &= \sup\{\zeta(v) \mid v \in \mathrm{DL}(P \parallel U_n) \text{ and } v \text{ not contain } \omega\} \\
&= \sup\{\zeta(w) \mid w \text{ is an } n\text{-critical path of } P\}
\end{aligned}
$$

Response Performance Calculation

In [CV2005], it ha been shown how to calcate the response performance for a response process starting from the refusal transition system RTS(P)

- Remove from RTS(P) all conditional time steps (and all processes that are not reachable anymore) that cannot partecipate in a full time when considering the behavior of $P \parallel U_n$ – rRTS(P)
- Consider a special subset of paths in rRTS(P) called n-critical paths $v \in DL(P \parallel U_n)$ that does not contain ω iff there is an n-critical path w of P such $\zeta(v) = \zeta(w)$

$$
\begin{aligned}
rp_P(n) &= \sup\{\zeta(v) \mid v \in \mathrm{DL}(P \parallel U_n) \text{ and } v \text{ not contain } \omega\} \\
&= \sup\{\zeta(w) \mid w \text{ is an } n\text{-critical path of } P\}
\end{aligned}
$$

- Worst-Cycles: **Catastrophic Cycles** and **Bad Cycles**

Catastrophic Cycles

- A cycle in rRTS(P) is **catastrophic** if it contains a positive number of time steps but no *in*'s and no *out*'s
- Let $P = in.\mathrm{rec}\, x.(\tau.x + out)$

$$P \xrightarrow{in} Q = \mathrm{rec}\, x.(\tau.x + out) \xrightarrow{\mathbb{A}}_r \mathrm{rec}\, x.(\underline{\tau}.x + \underline{out}) \xrightarrow{\tau} Q$$

- Along this cycle time increases without limits, but no 'useful' actions are performed
- As a consequence, $rp_P(n) = \infty$ for each n

Bad cycles

- For a P without catastrophic cycles, we consider cycles that may be reached from P by a path where all time steps are full and which themselves contains only full time steps.
- The **average performance** of such a cycle is

$$\frac{\text{the number of time steps}}{\text{the number of the } in\text{'s}}$$

- We call a cycle **bad** if it is a cycle of maximal average performance in rRTS(P)

An Example

rRTS(Seq)

$$\mathsf{Seq} \xrightarrow{in} \tau.out.\mathsf{Seq} \xrightarrow{\tau,\ \mathbb{A}\tau} out.\mathsf{Seq} \xrightarrow{\mathbb{A}} \underline{out}.\mathsf{Seq}$$

$$out.\mathsf{Seq} \xrightarrow{out} \mathsf{Seq} \qquad \underline{out}.\mathsf{Seq} \xrightarrow{out} \mathsf{Seq}$$

Figure: A cycle with average performance 1

An Example

rRTS(Seq)

$$\begin{tikzcd}
\mathsf{Seq} \arrow[r, "in"] & \tau.out.\mathsf{Seq} \arrow[r, "{\tau,\ \mathbb{A}\tau}"] & out.\mathsf{Seq} \arrow[r, "\mathbb{A}"] \arrow[ll, bend left, "out"] & \underline{out}.\mathsf{Seq} \arrow[lll, bend right, "out"']
\end{tikzcd}$$

Figure: A cycle with average performance 2

Theorem (Bad cycles theorem) [CV2005] : For P without catastrophic cycles, the response performance of P is asymptotically linear and its asymptotic factor is the average performance of a bad cycle

$$rp_{\mathsf{Seq}} = 2n + \Theta(1)$$

FASE (Faster Asynchronous Systems Evaluation)

FASE is a tool that support us in the evaluation of the evaluation of the response performance function. Main features:

- **Catastrophic-Cycles Detection**

  The problem has been reduced to that of finding shortest paths in a weighted modification of rRTS(P)

  Floyd-Warshall algorithm - $O(N^3)$

- **Bad-Cycles Detection**

  The problem has been reduced to that of finding a cycle with minimal average throughput (inverse of the average performance)

  Karp's algorithm - $O(N^3)$

- ***n*-critical path with maximal duration**

  Currently, FASE adopts an exhaustive research on the state space of rRTS(P) to find the maximal n-critical for a given n

FASE (Faster Asynchronous Systems Evaluation)

FASE is a tool that support us in the evaluation of the evaluation of the response performance function. Main features:

- **Catastrophic-Cycles Detection**
- **Bad-Cycles Detection**

  The algorithms for Catastrophic-Cycles and Bad-Cycles Detection have been improved

- ***n*-critical path with maximal duration**

  Currently, FASE adopts an exhaustive research on the state space of rRTS(P) to find the maximal n-critical for a given n

Relationship among our Performance Measures

- $P \sqsupseteq Q$ implies $rp_P \leq rp_Q$, but
- $rp_P \leq rp_Q$ does not necessarily imply that $P \sqsupseteq Q$

F. Corradini, M. R. Di Berardini, W. Vogler. PAFAS at Work: Comparing the Worst-Case Efficiency of Three Buffer Implementations. **APAQS 2001**: pp. 231-240.

F. Buti, M. Callisto, F. Corradini, M. Di Berardini, and W. Vogler. Evaluating the efficiency of asynchronous systems with fase. In pre-proceedings of QFM 2009, pp. 101-106.

Relationship among our Performance Measures

- $P \sqsupseteq Q$ implies $rp_P \leq rp_Q$, but
- $rp_P \leq rp_Q$ does not necessarily imply that $P \sqsupseteq Q$

F. Corradini, M. R. Di Berardini, W. Vogler. PAFAS at Work: Comparing the Worst-Case Efficiency of Three Buffer Implementations. **APAQS 2001**: pp. 231-240.

F. Buti, M. Callisto, F. Corradini, M. Di Berardini, and W. Vogler. Evaluating the efficiency of asynchronous systems with fase. In pre-proceedings of QFM 2009, pp. 101-106.

- The response performance function only considers a **restricted class of test environments**

Relationship among our Performance Measures

- $P \sqsupseteq Q$ implies $rp_P \leq rp_Q$, but
- $rp_P \leq rp_Q$ does not necessarily imply that $P \sqsupseteq Q$

  - F. Corradini, M. R. Di Berardini, W. Vogler. PAFAS at Work: Comparing the Worst-Case Efficiency of Three Buffer Implementations. **APAQS 2001**: pp. 231-240.
  - F. Buti, M. Callisto, F. Corradini, M. Di Berardini, and W. Vogler. Evaluating the efficiency of asynchronous systems with fase. In pre-proceedings of QFM 2009, pp. 101-106.

- The response performance function only considers a **restricted class of test environments**
- We can reasonably expect that if we use such a restricted class of environments also when comparing processes efficiency, (i.e., if we introduce a new efficiency preorder $\sqsupseteq_{io}$) then

$$P \sqsupseteq_{io} Q \text{ iff } rp_P \leq rp_Q$$

The Input/Output Testing Preorder

- A **i/o timed test** is a pair (U_n, D) where:
  - U_n is a test process in $\mathcal{U}$
  - $D \in \mathbb{N}_0$ is a time bound
- If P is a response process we say that P **must** (U_n, D) if for each $v \in \mathrm{DL}(P \parallel U_n)$ with duration $\zeta(v) > D$ contains some ω
- $P \sqsupseteq_{io} Q$ if, $n \in \mathbb{N}$ and for each $D \in \mathbb{N}_0$, Q **must** (U_n, D) implies P **must** (U_n, D)

A First Result

Theorem: $P \sqsupseteq_{io} Q$ if and only if $rp_P \leq rp_Q$

Sketch of the proof

$P \sqsupseteq_{io} Q$ iff, for each $n \in \mathbb{N}$,

$$\sup\{\zeta v \in \mathrm{DL}(\tau.P \parallel U_n) \text{ that does not contain } \omega\} \leq \sup\{\zeta v \in \mathrm{DL}(\tau.Q \parallel U_n) \text{ that does not contain } \omega\}$$

iff

$$\sup\{\zeta(w) \mid w \text{ is an } n\text{-critical path of } P \leq \sup\{\zeta(w) \mid w \text{ is an } n\text{-critical path of } Q\}$$

iff $rp_P(n) \leq rp_Q(n)$

Future Work

- It still missing a finite characterization of the preorder $\sqsupseteq_{io}$
- This finite characterization may also be useful to improve the algorithm that calculate the n-critical path with maximal duration
- Istantiating this framework in the timed automata setting
- Possibly with a finite characterization

Thank you for your attention