ODE-based general moment approximations for PEPA

R. A. Hayden       J. T. Bradley

Department of Computer Science
Imperial College London

PASTA 2008, Edinburgh
Prefix \((\alpha, r).P\) — action \(\alpha\) happens at rate \(r\) before transitioning to component \(P\)

Choice \(P + Q\) — allows either \(P\) or \(Q\) to occur, choice determined by races

Cooperation \(P \boxtimes L Q\) — For \(\alpha \in L\), if \(P\) and \(Q\) both enable an \(\alpha\)-activity, only then can both make the \(\alpha\)-transition simultaneously at the rate of the slowest

Hiding \(P/L\) — Action types in set \(L\) become the hidden \(\tau\) action, which cannot be cooperated over

Constant \(A \overset{\text{def}}{=} P\) — Assigns the label \(A\) to component \(P\), allows \(P \overset{\text{def}}{=} (\alpha, r).P\) etc.
Prefix \((\alpha, r).P\) — action \(\alpha\) happens at rate \(r\) before transitioning to component \(P\)

Choice \(P + Q\) — allows either \(P\) or \(Q\) to occur, choice determined by races

Cooperation \(P \bigotimes_{L} Q\) — For \(\alpha \in L\), if \(P\) and \(Q\) both enable an \(\alpha\)-activity, only then can both make the \(\alpha\)-transition simultaneously at the rate of the slowest

Hiding \(P/L\) — Action types in set \(L\) become the hidden \(\tau\) action, which cannot be cooperated over

Constant \(A \overset{\text{def}}{=} P\) — Assigns the label \(A\) to component \(P\), allows \(P \overset{\text{def}}{=} (\alpha, r).P\) etc.
Prefix \((\alpha, r).P\) — action \(\alpha\) happens at rate \(r\) before transitioning to component \(P\)

Choice \(P + Q\) — allows either \(P\) or \(Q\) to occur, choice determined by races

Cooperation \(P \triangleleft L Q\) — For \(\alpha \in L\), if \(P\) and \(Q\) both enable an \(\alpha\)-activity, only then can both make the \(\alpha\)-transition simultaneously at the rate of the slowest

Hiding \(P/L\) — Action types in set \(L\) become the hidden \(\tau\) action, which cannot be cooperated over

Constant \(A \overset{\text{def}}{=} P\) — Assigns the label \(A\) to component \(P\), allows \(P \overset{\text{def}}{=} (\alpha, r).P\) etc.
Prefix \((\alpha, r).P\)  — action \(\alpha\) happens at rate \(r\) before transitioning to component \(P\)

Choice \(P + Q\)  — allows either \(P\) or \(Q\) to occur, choice determined by races

Cooperation \(P \parallel_L Q\)  — For \(\alpha \in L\), if \(P\) and \(Q\) both enable an \(\alpha\)-activity, only then can both make the \(\alpha\)-transition simultaneously at the rate of the slowest

Hiding \(P/L\)  — Action types in set \(L\) become the hidden \(\tau\) action, which cannot be cooperated over

Constant \(A \overset{\text{def}}= P\)  — Assigns the label \(A\) to component \(P\), allows \(P \overset{\text{def}}= (\alpha, r).P\) etc.
Prefix \((\alpha, r).P\) — action \(\alpha\) happens at rate \(r\) before transitioning to component \(P\)

Choice \(P + Q\) — allows either \(P\) or \(Q\) to occur, choice determined by races

Cooperation \(P \otimes_{\mathcal{L}} Q\) — For \(\alpha \in \mathcal{L}\), if \(P\) and \(Q\) both enable an \(\alpha\)-activity, only then can both make the \(\alpha\)-transition simultaneously at the rate of the slowest

Hiding \(P/\mathcal{L}\) — Action types in set \(\mathcal{L}\) become the hidden \(\tau\) action, which cannot be cooperated over

Constant \(A \overset{\text{def}}{=} P\) — Assigns the label \(A\) to component \(P\), allows \(P \overset{\text{def}}{=} (\alpha, r).P\) etc.
Prefix \((\alpha, r).P\) — action \(\alpha\) happens at rate \(r\) before transitioning to component \(P\)

Choice \(P + Q\) — allows either \(P\) or \(Q\) to occur, choice determined by races

Cooperation \(P \otimes^L Q\) — For \(\alpha \in L\), if \(P\) and \(Q\) both enable an \(\alpha\)-activity, only then can both make the \(\alpha\)-transition simultaneously at the rate of the slowest

Hiding \(P/L\) — Action types in set \(L\) become the hidden \(\tau\) action, which cannot be cooperated over

Constant \(A \overset{\text{def}}{=} P\) — Assigns the label \(A\) to component \(P\), allows \(P \overset{\text{def}}{=} (\alpha, r).P\) etc.
A simple client/server model in PEPA

\[ C \overset{\text{def}}{=} (\text{request}, r_{\text{req}}).C_{\text{wait}} \]
\[ C_{\text{wait}} \overset{\text{def}}{=} (\text{data}, r_{\text{data}}).C_{\text{think}} \]
\[ C_{\text{think}} \overset{\text{def}}{=} (\text{think}, r_{\text{think}}).C \]

\[ S \overset{\text{def}}{=} (\text{request}, r_{\text{req}}).S_{\text{get}} + (\text{break}, r_{\text{break}}).S_{\text{broken}} \]
\[ S_{\text{get}} \overset{\text{def}}{=} (\text{data}, r_{\text{data}}).S + (\text{break}, r_{\text{break}}).S_{\text{broken}} \]
\[ S_{\text{broken}} \overset{\text{def}}{=} (\text{reset}, r_{\text{reset}}).S \]

\[ \text{System} \overset{\text{def}}{=} (C \parallel \ldots \parallel C) \overset{\text{ndef}}{=} (S \parallel \ldots \parallel S) \]

\[ 3^{N_C + N_S} \text{ states} \]
A simple client/server model in PEPA

\[ C \overset{\text{def}}{=} (\text{request}, r_{\text{req}}).C_{\text{wait}} \]
\[ C_{\text{wait}} \overset{\text{def}}{=} (\text{data}, r_{\text{data}}).C_{\text{think}} \]
\[ C_{\text{think}} \overset{\text{def}}{=} (\text{think}, r_{\text{think}}).C \]

\[ S \overset{\text{def}}{=} (\text{request}, r_{\text{req}}).S_{\text{get}} \]
\[ + (\text{break}, r_{\text{break}}).S_{\text{broken}} \]

\[ S_{\text{get}} \overset{\text{def}}{=} (\text{data}, r_{\text{data}}).S \]
\[ + (\text{break}, r_{\text{break}}).S_{\text{broken}} \]

\[ S_{\text{broken}} \overset{\text{def}}{=} (\text{reset}, r_{\text{reset}}).S \]

\[ \text{System} \overset{\text{def}}{=} (C \parallel \ldots \parallel C) \]
\[ \begin{array}{c} \{ \text{request, data} \} \end{array} \]
\[ \begin{array}{c} \begin{array}{c} \begin{array}{c} N_C \end{array} \end{array} \end{array} \]
\[ \begin{array}{c} \begin{array}{c} \begin{array}{c} (S \parallel \ldots \parallel S) \end{array} \end{array} \end{array} \]
\[ \begin{array}{c} \begin{array}{c} \begin{array}{c} N_S \end{array} \end{array} \end{array} \]

\[ 3^{N_C+N_S} \text{ states} \]

{rh, jb}@doc.ic.ac.uk
A simple client/server model in PEPA

\[
C \overset{\text{def}}{=} (\text{request}, \text{r}_\text{req}).C_{\text{wait}} \\
C_{\text{wait}} \overset{\text{def}}{=} (\text{data}, \text{r}_\text{data}).C_{\text{think}} \\
C_{\text{think}} \overset{\text{def}}{=} (\text{think}, \text{r}_\text{think}).C \\
\text{System} \overset{\text{def}}{=} (C \parallel \ldots \parallel C) \parallel (S \parallel \ldots \parallel S) \\
S \overset{\text{def}}{=} (\text{request}, \text{r}_\text{req}).S_{\text{get}} \\
\quad + (\text{break}, \text{r}_\text{break}).S_{\text{broken}} \\
S_{\text{get}} \overset{\text{def}}{=} (\text{data}, \text{r}_\text{data}).S \\
\quad + (\text{break}, \text{r}_\text{break}).S_{\text{broken}} \\
S_{\text{broken}} \overset{\text{def}}{=} (\text{reset}, \text{r}_\text{reset}).S \\
\exists^{N_C + N_S} \text{ states}
\]
Traditional ‘solution’ by fluid-analysis (ODEs)

\[ C \overset{\text{def}}{=} (\text{request}, r_{\text{req}}).C_{\text{wait}} \]

\[ C_{\text{wait}} \overset{\text{def}}{=} (\text{data}, r_{\text{data}}).C_{\text{think}} \]

\[ C_{\text{think}} \overset{\text{def}}{=} (\text{think}, r_{\text{think}}).C \]

\[ S \overset{\text{def}}{=} (\text{request}, r_{\text{req}}).S_{\text{get}} + (\text{break}, r_{\text{break}}).S_{\text{broken}} \]

\[ S_{\text{get}} \overset{\text{def}}{=} (\text{data}, r_{\text{data}}).S + (\text{break}, r_{\text{break}}).S_{\text{broken}} \]

\[ S_{\text{broken}} \overset{\text{def}}{=} (\text{reset}, r_{\text{reset}}).S \]

\[
\text{System} \overset{\text{def}}{=} \left( C \parallel \ldots \parallel C \right)_{N_C} \begin{array}{c} \{ \text{request, data} \} \end{array} \left( S \parallel \ldots \parallel S \right)_{N_S}
\]

At time \( t \), let \( n_C(t) \) count the number of \( C \) components etc.

- \( C \rightarrow C_{\text{wait}} \) at rate \( r_{\text{req}} \times \min(n_C(t), n_S(t)) \)
- \( C_{\text{think}} \rightarrow C \) at rate \( r_{\text{think}} \times n_{C_{\text{think}}}(t) \)

So approximate \( \mathbb{E}[n.C(t)] \) by \( v.C(t) \), defined by:

\[
\frac{dv.C(t)}{dt} = -r_{\text{req}} \times \min(v_C(t), v_S(t)) + r_{\text{think}} \times v_{C_{\text{think}}}(t) \text{ etc.}
\]
Traditional ‘solution’ by fluid-analysis (ODEs)

\[
C \overset{\text{def}}{=} (\text{request, } r_{\text{req}}).C_{\text{wait}} \\
C_{\text{wait}} \overset{\text{def}}{=} (\text{data, } r_{\text{data}}).C_{\text{think}} \\
C_{\text{think}} \overset{\text{def}}{=} (\text{think, } r_{\text{think}}).C \\
\]

\[
S \overset{\text{def}}{=} (\text{request, } r_{\text{req}}).S_{\text{get}} \\
\quad + (\text{break, } r_{\text{break}}).S_{\text{broken}} \\
S_{\text{get}} \overset{\text{def}}{=} (\text{data, } r_{\text{data}}).S \\
\quad + (\text{break, } r_{\text{break}}).S_{\text{broken}} \\
S_{\text{broken}} \overset{\text{def}}{=} (\text{reset, } r_{\text{reset}}).S \\
\]

\[
\text{System} \overset{\text{def}}{=} (C \parallel \ldots \parallel C) \overset{\{\text{request, data}\}}{\otimes} N_C \\
\quad \overset{\{\text{solutions}\}}{\otimes} (S \parallel \ldots \parallel S) \overset{N_S}{\otimes} \\
\]

At time \( t \), let \( n_C(t) \) count the number of \( C \) components etc.

\begin{itemize}
  \item \( C \rightarrow C_{\text{wait}} \) at rate \( r_{\text{req}} \times \min(n_C(t), n_S(t)) \)
  \item \( C_{\text{think}} \rightarrow C \) at rate \( r_{\text{think}} \times n_{C_{\text{think}}}(t) \)
\end{itemize}

So approximate \( E[n.(t)] \) by \( v.(t) \), defined by:

\[
\frac{dv_C(t)}{dt} = -r_{\text{req}} \times \min(v_C(t), v_S(t)) + r_{\text{think}} \times v_{C_{\text{think}}}(t) \text{ etc.}
\]
Fluid-analysis example results (clients)

- Client (ODEs)
- Client (SS)
- Client waiting (ODEs)
- Client waiting (SS)
- Client thinking (ODEs)
- Client thinking (SS)

Number of active components vs. Time, t
Current limitations of this approach

- Usually interpreted as an approximation to the expectation of the component counts. But what about variance (for example)?
- Limited knowledge of the quality of this approximation wrt. underlying CTMC
- Works better in some situations than in others — hard to predict in general

This work is concerned with improving the situation as far as these three issues are concerned.
From CTMC to existing fluid-analysis

Considering client/server example again, we can show e.g.:

\[ \text{CTMC} \Rightarrow \frac{d\mathbb{E}[n_C(t)]}{dt} = -r_{req} \times \mathbb{E}[\min(n_C(t), n_S(t))] + r_{think} \times \mathbb{E}[n_{C_{\text{think}}}(t)] \]

Applying the approximation \( \mathbb{E}[\min(\cdot, \cdot)] \approx \min(\mathbb{E}[\cdot], \mathbb{E}[\cdot]) \) gives:

\[ \frac{d\mathbb{E}[n_C(t)]}{dt} \approx -r_{req} \times \min(\mathbb{E}[n_C(t)], \mathbb{E}[n_S(t)]) + r_{think} \times \mathbb{E}[n_{C_{\text{think}}}(t)] \]

In general, existing fluid-analysis depends only on application of this approximation for all models where the ODEs do not involve rational functions.
Switch points

Points at which $\min(\cdot, \cdot)$ terms in the ODEs ‘change sides’

- Far away from them, approximation will be good
- Close to them, potentially much worse
- How far away/close depends on variability
Heterogeneous cooperation rates

\[ E[\min(n_C(t)r_1, n_S(t)r_2)] \approx \min(E[n_C(t)]r_1, E[n_S(t)]r_2) \]

\[ r_1 = 1, \ r_2 = 50 \]

\[ E[C] = 8 \]
\[ E[S] \times 50 = 25 \]
\[ \min(E[C], E[S] \times 50) = 8 \]
\[ E[\min(C, S \times 50)] = 4 \]

\[ r_1 = r_2 = 1 \]

\[ E[C] = 8 \]
\[ E[S] = 8 \]
\[ \min(E[C], E[S]) = 8 \]
\[ E[\min(C, S)] \approx 7.5 \]
Effectively passive example

The graph shows the number of active components over time for different scenarios:
- **Client (ODEs)**
- **Client (SS)**
- **Client waiting (ODEs)**
- **Client waiting (SS)**
- **Client thinking (ODEs)**
- **Client thinking (SS)**

The x-axis represents time, t, ranging from 0 to 100, and the y-axis represents the number of active components, which ranges from 0 to 20.
From CTMC to new fluid-analyses

Consider client/server example again, we can show e.g. :

\[
\frac{d\mathbb{E}[n_C^2(t)]}{dt} = \mathbb{E}[\min(n_C(t), n_S(t))]r_{req} - 2\mathbb{E}[\min(n_C^2(t), n_C(t)n_S(t))]r_{req} \\
+ \mathbb{E}[n_{Ct}(t)]r_{think} + 2\mathbb{E}[n_C(t)n_{Ct}(t)]r_{think}
\]

Applying the same approximation \( \mathbb{E}[\min(\cdot, \cdot)] \approx \min(\mathbb{E}[\cdot], \mathbb{E}[\cdot]) \) and writing e.g. \( v_{C\cdot S}(t) \) for the approximation to \( \mathbb{E}[n_C(t)n_S(t)] \) etc. then yields:

\[
\frac{dv_{C_2^2}(t)}{dt} = \min(v_C(t), v_S(t))r_{req} - 2\min(v_{C_2^2}(t), v_{C\cdot S}(t))r_{req} \\
+ v_{Ct}(t)r_{think} + 2v_{C \cdot C_t}(t)r_{think}
\]

We can repeat this programme for all 27 first and second order moments.
Variance approximations for client/server example

- Client (ODEs)
- Client (SS)
- Client waiting (ODEs)
- Client waiting (SS)
- Client thinking (ODEs)
- Client thinking (SS)
Observations

- Even after 100,000 independent replications, stochastic simulation still exhibited visible fluctuations, ODE-analysis thus even more valuable in the case of variance etc.?

- Same ideas work for moments of any order of the component counting processes, relying only on the approximation $\mathbb{E}[\min(\cdot, \cdot)] \approx \min(\mathbb{E}[\cdot], \mathbb{E}[\cdot])$
Conclusions

- We have exhibited the nature of the existing fluid-analysis for PEPA models, presenting insight into how the quality of analysis varies with model structure and parameters.

- We have derived naturally ODE-approximations for arbitrary higher order moments, providing access to previously inaccessible quantities such as variance and skewness.
Future work

- The case of fluid-analysis involving ODEs with rational functions can be treated with more complicated underlying approximations — such models arise in passage-time computations (see our other talk)...

- Can we use the extra knowledge of the distribution of the component counts to improve the original first order fluid-analysis?
Questions?