Partial Evaluation of PEPA Models for Fluid-flow Analysis

Allan Clark, Adam Duguid, Stephen Gilmore and Mirco Tribastone

LFCS, University of Edinburgh

Abstract. We present an application of *partial evaluation* to performance models expressed in the PEPA stochastic process algebra [1]. We partially evaluate the state-space of a PEPA model in order to remove uses of the cooperation and hiding operators and compile an arbitrary sub-model into a single sequential component. This transformation is applied to PEPA models which are not in the correct form for the application of the fluid-flow analysis for PEPA [2]. The result of the transformation is a PEPA model which is amenable to fluid-flow analysis but which is *strongly equivalent* [1] to the input PEPA model and so, by an application of Hillston's theorem, performance results computed from one model are valid for the other. We apply the method to a Markovian model of a key distribution centre used to facilitate secure distribution of cryptographic session keys between remote principals communicating over an insecure network.

1 Introduction

Fluid-flow approximation of PEPA models [2] enables the numerical analysis of models of vast scale using ordinary differential equations (ODEs). The model sizes which can be analysed using transformation into an ODE representation pass effortlessly beyond the maximum attainable using exact discrete-state representations such as continuous-time Markov chains. However, fluid-flow analysis is applicable to PEPA models in a particular form where the model is structured as the cooperation of replicated copies of sequential components. For example, if P, Q and R are sequential PEPA components available in M, N and O replications and $\mathcal{K}$ and $\mathcal{L}$ are cooperation sets then the model

$$P[M] \underset{\mathcal{K}}{\bowtie} \Big(Q[N] \underset{\mathcal{L}}{\bowtie} R[O] \Big)$$

is immediately suitable for fluid-flow analysis but the model

$$P[M] \underset{\mathcal{K}}{\bowtie} \Big((Q \underset{\mathcal{L}}{\bowtie} R)[N] \Big)$$

is not, because of the use of the cooperation operator ($\bowtie$) nested inside the array of N replications. We use partial evaluation to transform the unsuitable model into an equivalent model of the form:

$$P[M] \underset{\mathcal{K}}{\bowtie} QR[N]$$

The new model has a new sequential component QR which exactly respects the interaction between the original Q and R. The new sequential component is generated in such a way that we can recover the states of Q and R from the state of QR. The transformation can be applied compositionally to a model to generate an equivalent which is suitable for fluid-flow analysis without generating the full state-space of the original model. Specifically, the cost of the transformation depends only on the form of the components Q and R and does not depend on the values of M or N.

The original contributions of the present paper are the following.

1. We present a novel application of Hillston's theorem to the partial evaluation of a PEPA model. The theorem [1] guarantees that the strong equivalence relation of PEPA is a congruence for the PEPA process algebra and thus the partially evaluated model is equivalent to the original.
2. We present four styles of analysis of the same model three of which allow the analysis at large scales. The model analysed represents the Needham-Schroeder-Lowe protocol. We compare the performance results obtained using all four analysis methods.

2 Case study: Key Distribution Centres

Key distribution centres enable secure communication between remote principals across an insecure network. The distribution centre acts as a trusted third party, allowing users to register a key with the centre and use a robust cryptographic protocol to establish a secure communication between two principals who have no previous communication history and no secure shared communications channels.

One possible candidate for the chosen cryptographic protocol is the Needham-Schroeder-Lowe protocol [3] which hardens the Needham-Schroeder protocol [4] against replay attacks. The goal of the protocol is to enable secure communication between Alice and Bob. The protocol has five steps, which we describe informally first.

- Alice sends a message to the server requesting a session with Bob.
- The server generates a new session key K_{AB}, encrypted under Alice's registered key, K_{AS}, together with a copy encrypted for Bob.
- Alice forwards the copy on to Bob, who can decrypt it.
- Bob sends a random number (a *nonce*) to Alice, encrypted under the session key.
- Alice makes a small change to the nonce and sends it back to Bob.

The traditional representation of such a protocol is as a narration, setting out more methodicially the information presented above. In the notation used below $X \to Y$ denotes a communication from X to Y, $x_1, \ldots, x_n$ denotes a tuple of n values and $\{x_1, \ldots, x_n\}_K$ denotes a tuple of n values encrypted under the

cryptographic key K.

$$
\begin{aligned}
(\mathit{request})\ 1.\ & A \rightarrow S : A, B, N_A \\
(\mathit{response})\ 2.\ & S \rightarrow A : \{N_A, K_{AB}, B, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}} \\
(\mathit{sendBob})\ 3.\ & A \rightarrow B : \{K_{AB}, A\}_{K_{BS}} \\
(\mathit{sendAlice})\ 4.\ & B \rightarrow A : \{N_{AB}\}_{K_{AB}} \\
(\mathit{confirm})\ 5.\ & A \rightarrow B : \{N_{AB} - 1\}_{K_{AB}}
\end{aligned}
$$

After these five steps are complete Alice and Bob can use the key in a secure session (*usekey*).

A representation of the protocol such as this is adequate for the analysis of the correctness of function of the protocol using a logic such as the BAN logic [5] but it is not suitable for performance analysis. Time is abstracted away in the model above, as it is in classical process algebras. In a *stochastic* process algebra such as PEPA [1] the communication events and the encryption steps have an expected average duration. Performance results such as response time and utilisation can be calculated from a PEPA model of a key distribution centre, as shown in [6] and [7]. Conversely, data is abstracted away in the PEPA model and so it is not suitable for correctness analysis[1].

A PEPA model of a key distribution centre such as the one shown in Figure 1 can be used to produce a finite discrete-state representation of the system with quantified durations associated to each activity.

$$
\begin{aligned}
\mathit{KDC} &\stackrel{def}{=} (\mathit{request}, r_q).\mathit{KDC} + (\mathit{response}, r_p).\mathit{KDC} \\
\\
\mathit{Alice} &\stackrel{def}{=} (\mathit{request}, r_q).(\mathit{response}, r_p).\mathit{Alice}_1 \\
\mathit{Alice}_1 &\stackrel{def}{=} (\mathit{sendBob}, r_B).\mathit{Alice}_2 \\
\mathit{Alice}_2 &\stackrel{def}{=} (\mathit{sendAlice}, \top).(\mathit{confirm}, r_c).\mathit{Alice}_3 \\
\mathit{Alice}_3 &\stackrel{def}{=} (\mathit{usekey}, r_u).\mathit{Alice} \\
\\
\mathit{Bob} &\stackrel{def}{=} (\mathit{sendBob}, \top).\mathit{Bob}_1 \\
\mathit{Bob}_1 &\stackrel{def}{=} (\mathit{sendAlice}, r_A).(\mathit{confirm}, \top).\mathit{Bob}_2 \\
\mathit{Bob}_2 &\stackrel{def}{=} (\mathit{usekey}, \top).\mathit{Bob} \\
\\
\mathit{System} &\stackrel{def}{=} \mathit{KDC} \underset{\mathcal{L}}{\bowtie} \Big((\mathit{Alice} \underset{\mathcal{M}}{\bowtie} \mathit{Bob})[N] \Big) \\
\\
\text{where } \mathcal{L} &= \{\, \mathit{request}, \mathit{response} \,\} \\
\mathcal{M} &= \{\, \mathit{sendBob}, \mathit{sendAlice}, \mathit{confirm}, \mathit{usekey} \,\}
\end{aligned}
$$

Fig. 1. PEPA model of the Key Distribution Centre presented in [6]

[1] The original Needham-Schroeder protocol and the modified Needham-Schroeder-Lowe protocol would have the same representation in PEPA.

The derivation graph underlying this PEPA model can be converted into a Continuous-Time Markov Chain (CTMC) which can be readily solved to find the steady-state distribution over all of the reachable states of the model [8] or analysed to determine transient probability distributions and response-time profiles [9].

Exact discrete-state models of complex systems face the well-known problem of state-space explosion where, as the complexity of the system under study increases, there is an exponential growth in the state-space of the underlying model. Out-of-core [10] and disk-based solution methods [11] allow modellers to tolerate very large state-spaces but at the cost of greater and greater numerical solution times.

As the number of paired principals (N) in the PEPA model increases the machine representation of the probability distribution requires more and more storage and longer and longer computation times to calculate. The size of the state space of the Markovian model is 6^N which grows very rapidly with N. Fortunately we have available in the PEPA Eclipse Plug-in an implementation of the state-space aggregation algorithm for PEPA [12] which allows us to better cope with increases in N. The state-space sizes before and after aggregation are shown in Table 1.

N	Full state-space size	Aggregated state-space size
1	6	6
2	36	21
3	216	56
4	1,296	126
5	7,776	252
6	46,656	462
7	279,936	792
8	1,679,616	1,287
9	10,077,696	2,002
10	60,466,176	3,003
11	362,797,056	4,368
12	2,176,782,336	6,188

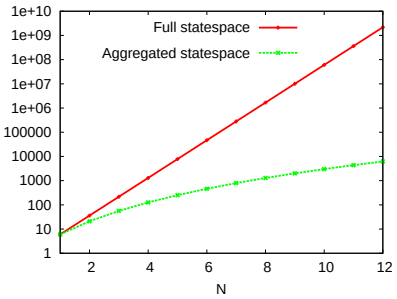


Table 1. Full and aggregated statespace sizes as calculated by the PEPA Eclipse Plugin [8].

Use of the aggregation algorithm allows us to tolerate larger state-spaces but this too will reach a limit and at that point we will need to rely on other techniques to analyse the model. In this paper we make use of three techniques; fluid-flow analysis, stochastic simulation and reduction to a closed queueing network, to allow us to continue to analyse the model past the limit on the number of clients imposed by the state-space explosion problem on the CTMC analysis. For two of the techniques the model must first be partially evaluated and this is discussed in the next section. Following this we detail the analysis by each method.

3 Partial Evaluation

Our model of the key distribution centre shown in Figure 1 exhibits synchronisation nested within an array and is thus in its present form unsuitable for fluid-flow analysis. However we can apply the partial evaluation technique described in the introduction transforming the synchronisation: $(Alice \underset{\mathcal{M}}{\bowtie} Bob)$ into an equivalent component: *AliceBob*.

This is achieved by considering this synchronisation as an entire model and deriving the entire state-space of this smaller model. Deriving the entire state-space of this synchronisation transforms multiple (in this case two) synchronised components into a single sequential component. The number of states in the new sequential component depends only on the form and synchronised activies of the involved components and not on any part of the larger model containing the original cooperation. In particular if the synchronisation occurs nested within an array – as is the case with our model – the partial evalation is the same regardless of the size of the array.

$$KDC \stackrel{def}{=} (request, r_q).KDC + (response, r_p).KDC$$

$$\begin{aligned}
AliceBob &\stackrel{def}{=} (request, r_q).AliceBob_1 \\
AliceBob_1 &\stackrel{def}{=} (response, r_p).AliceBob_2 \\
AliceBob_2 &\stackrel{def}{=} (sendBob, r_B).AliceBob_3 \\
AliceBob_3 &\stackrel{def}{=} (sendAlice, r_A).AliceBob_4 \\
AliceBob_4 &\stackrel{def}{=} (confirm, r_c).AliceBob_5 \\
AliceBob_5 &\stackrel{def}{=} (usekey, r_u).AliceBob
\end{aligned}$$

$$System \stackrel{def}{=} KDC \underset{\mathcal{L}}{\bowtie} AliceBob[N]$$

$$\text{where } \mathcal{L} = \{ request, response \}$$

Fig. 2. Partially evaluated PEPA model of the Key Distribution Centre

When we apply partial evaluation to the PEPA model shown in Figure 1 we obtain the model in Figure 2. There is a one-to-one correspondence between the states of the original synchronised components $(Alice \underset{\mathcal{M}}{\bowtie} Bob)$ and the sequential component *AliceBob* as indicated in Table 2. In turn there is a one-to-one correspondence between the states of the original PEPA model and the states of the partially-evaluated PEPA model.

4 Analysis

After the partial evaluation of the key distribution centre PEPA model we have four distinct forms of analysis which we may apply:

Original PEPA model		Partially-evaluated PEPA model
$Alice$	Bob	$AliceBob$
$(response, r_p).Alice_1$	Bob	$AliceBob_1$
$Alice_1$	Bob_1	$AliceBob_2$
$Alice_2$	Bob_1	$AliceBob_3$
$(confirm, r_c).Alice_3$	$(confirm, \top).Bob_2$	$AliceBob_4$
$Alice_3$	Bob_2	$AliceBob_5$

Table 2. Bisimilar states in the original and partially-evaluated PEPA models.

1. exact discrete-state analysis by solving the underlying continuous-time Markov chain;
2. manually reduce and approximate the model to a closed queueing system as is done in [6];
3. approximate discrete-state analysis by stochastic simulation of the underlying continuous-time Markov chain; and
4. approximate continuous-state analysis by numerical integration of the underlying fluid-flow differential equations.

We perform each of these in turn on the model of the key distribution centre.

4.1 Markovian analysis

The first programme of analysis which we undertake is to examine the probability density function (pdf) and cumulative density function (cdf) for a passage through the system behaviour. For this analysis we have used the International PEPA Compiler (ipc) [9] tool suite.

The measurement which we make is from the end of an occurrence of the *request* activity to the end of a *confirm* activity. This measures the time it takes to restart a session and gives us our notion of response time for this system. Note though that in this passage only the *response* activity cooperates with the server the other activities are performed within a single *AliceBob* component and hence the delay from these activities is unaffected by the number of clients in the system. This is an important performance metric since sessions may need to be restarted frequently. This is necessary because after a period of continued use a session key should not be considered safe due to too many ciphertexts being available to an attacker.

We vary the rate r_u (the rate of the *usekey* activity). This essentially varies the duration of a session (*usekey* is the session). We are measuring between the *request* and *confirm* activities as performed by only the *first* Alice/Bob pairing because otherwise we have essentially a meaningless measurement. The ability to isolate and probe a particular Alice/Bob pair is given to us by the use of *location-aware* stochastic probes [13]. The results are shown in Figure 3.

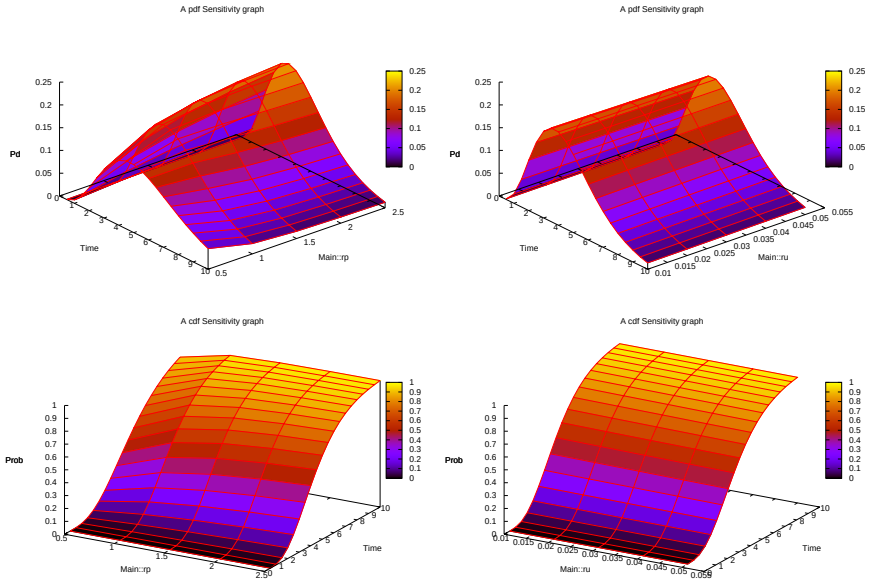


Fig. 3. The top two graphs depict the probability density function (pdf) and the bottom two graphs the cumulative density function (cdf) of the passage from *usekey* to *confirm* for a particular Alice/Bob pair. In the graphs on the left we vary the rate at which the server responds while in the graphs on the right we vary the rate at which session keys are consumed and hence the rate at which requests arrive at the server.

Because we are using the full numerical solution technique we are limited in the number of clients that may participate in the model. Because of the low number of clients the centre is able to cope with demand very well and varying the rate at which keys are used – and therefore the rate at which requests arrive – does not have a significant impact on the passage of interest. This further motivates us to apply large state-space size analysis techniques.

The numerical solution of the underlying Markov chain has allowed us to obtain response-time quantiles. However we can also use the Markov chain to compute the steady-state probability distribution, i.e. the long-term probability of being in each state. This information allows us to compute throughput and utilisation which for this model will give us the average number of clients waiting to be processed by the server and the average response-time. The results are computed using the PEPA Eclipse Plug-in [8] and are shown together in comparison to the same measures as computed using the other three analysis techniques in Section 4.3.

4.2 Analytical Solution

We have seen that it is possible to transform the original model into one suitable for analysis using ordinary differential equations and stochastic simulation while being certain that we are analysing an equivalent model. Another technique is to continue simplifying the model until we have one which may be solved analytically. We may lose the exact correspondence between the original model and simplified model however if we are careful in our transformations we may still relate the performance measurements obtained from the simplified model to the original model.

In the case of the key distribution centre it is possible to reduce the model to that of a simple closed queueing system. This simpler model has one queue station representing the key distribution centre and each client performing an exponential delay after being serviced before returning to the queue. This technique is described in detail in [6]. This represents only an approximate solution because in the original model each client pair performs a sequence of activities before returning to the queue. This sequence of (exponentially delayed) activities will give rise to an Erlang distributed delay.

From this closed queueing system we can compute the average number of clients waiting in the queue and from this the average response time. As mentioned above these results are shown for all analysis techniques in Section 4.3.

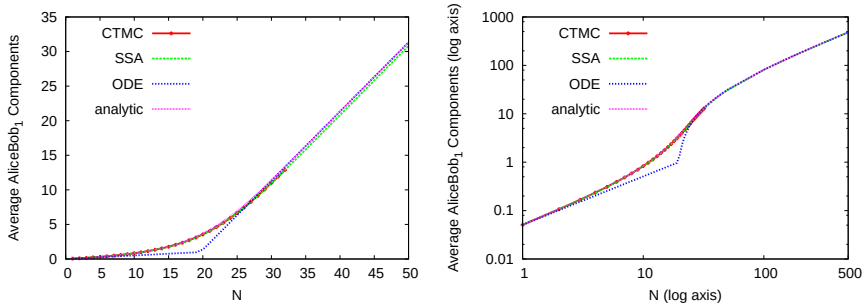
4.3 Simulation and fluid-flow analysis

The work in the early part of this paper was concerned with transforming the PEPA model of the key distribution centre into a form which was suitable for ODE analysis. We reap the benefits of this work here because we can efficiently compute mean trajectories through the model state space for large-scale models.

In this section we perform those analyses with the partially-evaluated model. The graphs in Figure 4 show the average number of $AliceBob_1$ components in the system varying as we increase N – the total number of Alice and Bob pairs in the system. This gives us the average number of clients that are waiting to be served by the key distribution center as a function of the total number of clients. Results are shown for all four analysis methods though the numerical solution via a Markov chain has results only up to $N = 32$, the limit before solving the Markov chain becomes too expensive. The graph on the left highlights the small values of $N \leq 50$ and the graph on the right depicts all values of $N \leq 500$.

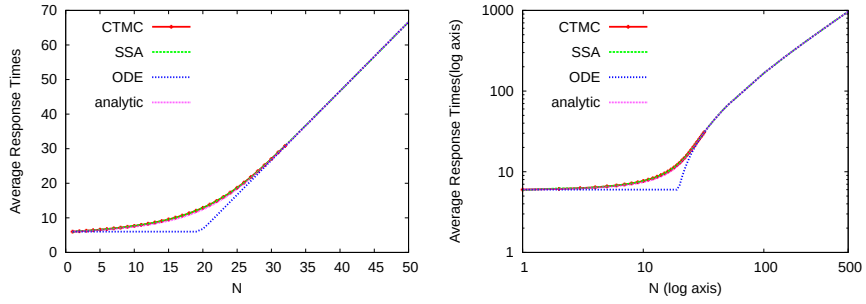
Similarly the graphs in Figure 5 show average response-time as computed using all four analysis methods. Once again the Markov chain solution is limited to values of $N \leq 32$ and correspondingly the graph on the left highlights the smaller values of N while the graph on the right allows N to increase to 500.

Encouragingly we see that as N increases the agreement between the measurements improves. At very small values of N the ODE differs from the other methods, whilst the queueing method, stochastic simulation and Markov chain method continue to show good agreement for all measured values of N. This provides us with confidence that the stochastic simulation analysis is providing



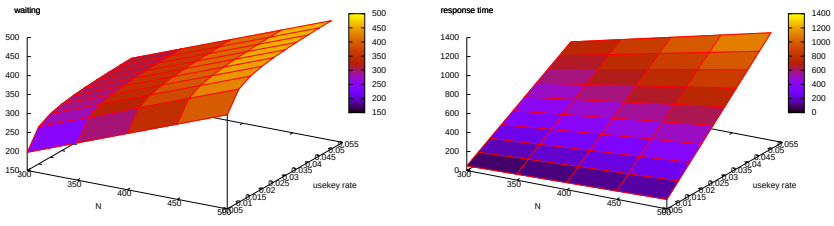
(a) (b)

Fig. 4. The average number of AliceBob1 components varied as the number of client pairs (N) is increased, using all of the separate analysis techniques.



(a) (b)

Fig. 5. The average response times measured as the number of clients is increased.



(a) (b)

Fig. 6. The sensitivity at varying values of N of the rate of key consumption (the rate of the *usekey* activity) as measured by the ODE analysis.

accurate results and hence we use this to compare how well our ODE analysis is performing at large values of N where it is not possible to compare with the Markov chain solution. We are also pleased to note that the queueing and ODE methods begin to show agreement with the Markov chain solution (and therfore also the simulation results) before the limit of the Markov chain solution. In the following section we give a more detailed comparison of the results.

Sensitivity Analysis Recall from Section 4.1 that we performed sensitivity analysis for small values of N using the Markovian solution method of analysis. We found that varying the rate at which the server responds affects the response time as one would expect. However varying the rate at which keys are consumed – and therefore the rate at which requests arrive at the server – did not significantly affect the response time. We reasoned that this was because N was so low that whenever a client made a request there was very likely to be no other clients already waiting in the queue even as we increased the rate of key consumption. To achieve any noticable effect the rate that the clients use the key must be set unrealistically high.

We have repeated this sensitivity analysis at higher levels of N in order to understand the influence of varying the rate of key consumption. The graphs in figure Figure 6 show the effect that varying this rate has on the number of clients waiting (left) and the response time (right). Here we can see a significant effect caused by changing the rate at which requests arrive at the server. Having done this we can conclude that although the Markovian solution allows for very accurate results, if the number of clients is unrealistically low, any conclusion obtained from sensitivity analysis of the Markovian method cannot be assumed to apply at larger values of N. Thus our analysis using ordinary differential equations and stochastic simulation is a necessary endeavour.

5 Comparison

In the previous section we compared the results from the four methods of analysis, noting disagreement at low values of N ($N <\approx 25$) between the ODE analysis and other methods. For higher values of N the data indicated better agreement over all four methods. In this section we look closer at the differences between the different analyses.

Taking the Markovian analysis as our yardstick we can compare how well the other three analysis methods perform for values of $N \leq 32$. The graphs in Figure 7 show the error in the measured average number of waiting clients. The graphs in Figure 8 show the error as compared with the Markovian solution in the measured response time. In both sets of graphs we depict in the left graph the absolute error, as the difference between the given analysis method and Markovian solution, while in the right hand graph this value is given relative to the number of clients in the system. This is because, particularly in the case of average number of clients waiting, an error of 5 is more significant when there are only 15 clients in the system as when there are over 100. In the case of the

response-time error normalising by the number of clients does not make sense so we instead divide the error by the response-time as calculated by the Markov chain solution.

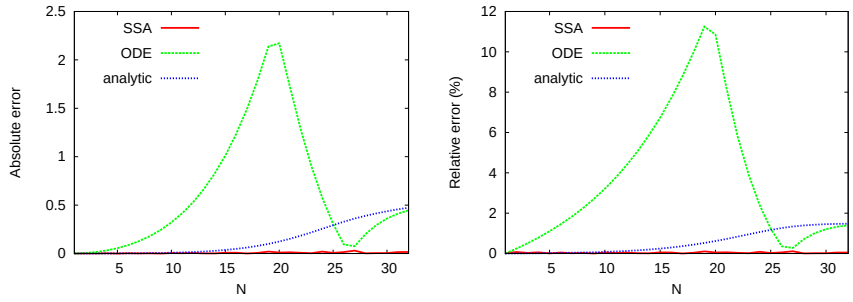


Fig. 7. Each graph plots the difference in measured average number of waiting clients for the given analysis method against the Markov chain solution as the number of clients N is increased. The graph on the left (a) depicts the absolute error while the graph on the right (b) depicts the error relative to the number of clients.

The disagreement seen in Figures 4 and 5 can clearly be seen here, peaking at approximately 12% around $N = 20$ for the ODE analysis. The SSA analysis shows very good agreement with the CTMC, shown by a maximum error of 0.1% for $N = 1 \ldots 32$. What the previous graphs failed to show was the minor, but increasing error between the CTMC and analytical analysis. What is unclear from Figure 7 is whether the error between the CTMC and analytical approaches has peaked or not. If we assume the level of error between CTMC and SSA were to remain constant, as we did in Section 4.3, then we can be confident comparing the SSA and the analytical method. This comparison shows an error of no more than 1.4% for $N = 1 \ldots 500$. Looking at response times (Figure 8) we see that for lower values of N the error can be as high as 50% and a peak of 4.5% for the analytical method.

So far comparison has been at steady-state; however SSA and ODE analysis both allow comparison for any value of t. Figure 9 shows the time-series data for the number of waiting clients ($AliceBob_1$), and Figure 10 the absolute and relative error for $N = 5 \ldots 50$. Figure 11 shows the absolute and relative error for $N = 50 \ldots 300$.

What is clear from these graphs is that the reported error for the steady-state is not the peak error seen. From Figure 7 we could see an error of approximately 12% when $N \approx 20$, whereas the peak value seen from this sample of N is closer to 15% when $N = 5$. Figure 11 shows that the peak error observed decreases as N increases, showing a peak error of approximately 3.5% for $N = 300$, and a steady-state error of 0.2%. If the graph was extended to show $N = 500$ the

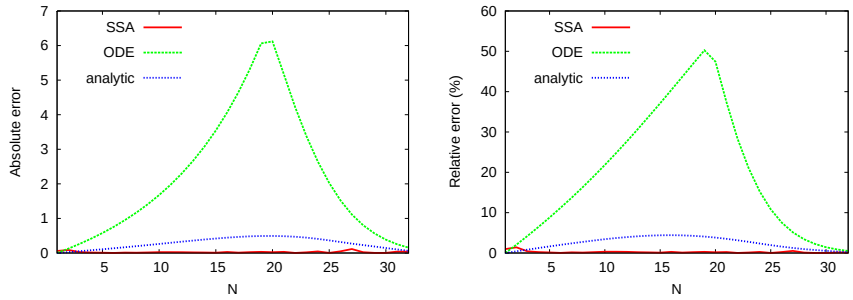


Fig. 8. Each graph plots the difference in measured response time for the given analysis method against the Markov chain solution as the number of clients N is increased. The graph on the left (a) depicts the absolute error while the graph on the right (b) depicts the error relative to the response-time as calculated using the Markov chain solution.

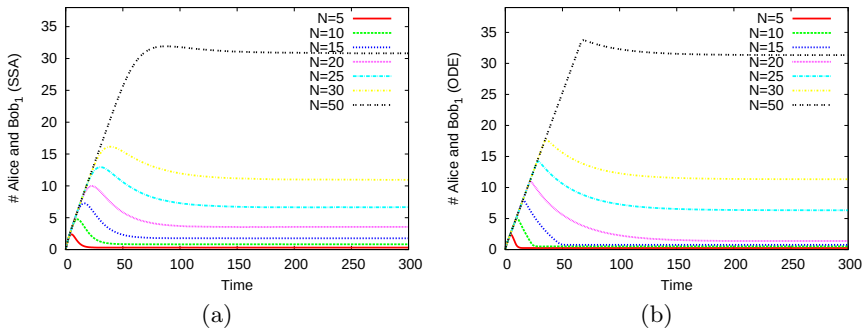
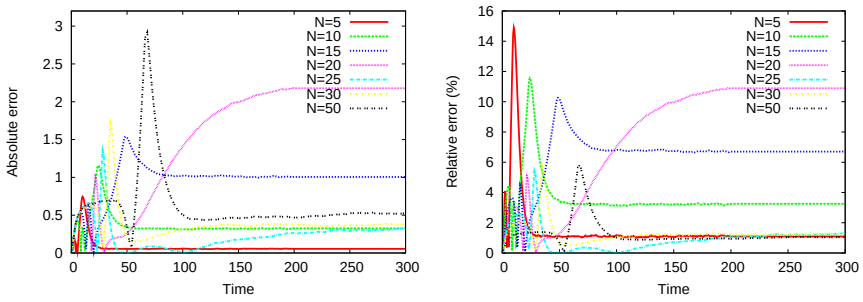


Fig. 9. Time-series data for SSA and ODE, graph (a) and (b) respectively, for various values of N.

peak error seen would be approximately 3% and a steady-state error of less than 0.1%.

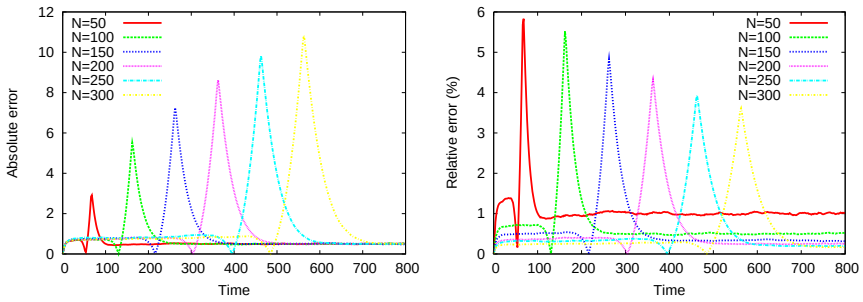
6 Conclusions

By applying partial evaluation of the Markovian state-space of a PEPA model we have been able to transform a model unsuitable for fluid-flow analysis into one for which fluid-flow analysis is immediately applicable. Fluid-flow analysis allows us to examine the dynamics of models of large-scale at low computational cost. This has provided us with four possible analysis methods for this model each with a distinct set of advantages and disadvantages:



(a) (b)

Fig. 10. Graphs for absolute error (a) and relative error (b) for $N = 5 \ldots 50$.



(a) (b)

Fig. 11. Graphs for absolute error (a) and relative error (b) for $N = 50 \ldots 300$.

CTMC Compiling the model to the CTMC allows for the most detailed analysis of the model. We can obtain passage-time quantiles which none of the other analysis methods yet support. The disadvantage though is clear - the CTMC representation suffers from the well known state-space explosion problem. We can mitigate this to some degree using state-space aggregation. In this particular example we could cope with values of N up to 32.

Analytical Approximation By first manually reducing the model to an equivalent closed queueing network we can cope with values of N much larger than for the CTMC based analysis. We can comfortably cope with values of N over a thousand. The disadvantage of such an approach is that it is available to only a specific kind of model, namely those which may be reduced to a closed queueing system. Additionally such a transformation is model dependent and therefore must be done manually. In each case the modeller must work to show that the simplified model is indeed equivalent to the original model or that the approximation is close enough.

Simulation We can use stochastic simulation to analyse our model. This again allows us to cope with larger state space sizes. The drawback in this case is that our results will only ever be an approximation to the true results. As our accuracy requirements increase so do the number of simulations which must be run and thus the computation time.

Fluid-flow The approach we have used in this paper is to partially-evaluate the model into a form suitable for translation into ordinary differential equations. This has similar advantages to the reduction to a closed queueing network. However the partial evaluation of the parallel sub-components is a well defined transformation which may be automated. Moreover all partially-evaluated models are known to be equivalent to their original models. The disadvantage of this approach is that for the solution to the ODEs to be accurate we require a large number of components. That is the model cannot be used for small values of N. This is in direct contrast to the CTMC method which can be used effectively for small values of N but cannot cope with larger values of N.

We believe that the combination of CTMC analysis for small values of N and ODE analysis – via partial evaluation of parallel sub-components where required – forms an important partnership in the analysis of large scale parallel models. The CTMC analysis can be used not only for analysis for small values of N where the ODE analysis is inappropriate but also to gain greater insight into the properties of the model since the CTMC analysis permits such analyses as the computation of passage-time quantiles. Meanwhile analysis for large values of N can be obtained through fluid-flow analysis.

In doing such a combination of analyses the modeller will likely look for the crossover point. The crossover point is the value of N at which the ODE analysis agrees with the CTMC analysis. In general we would like this value to be lower than the upper bound on N for the CTMC analysis. Otherwise the modeller must guess at the crossover point though stochastic simulation can be used to provide some assurance. Through our use of aggregation of the state-space of the model we are hopeful that many models fall into the former category.

Acknowledgements: This work has been partially sponsored by the project SENSORIA, IST-2005-016004.

References

1. J. Hillston. *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996.
2. J. Hillston. Fluid flow approximation of PEPA models. In *Proceedings of the Second International Conference on the Quantitative Evaluation of Systems*, pages 33–43, Torino, Italy, September 2005. IEEE Computer Society Press.
3. G. Lowe. An attack on the Needham-Schroeder public key authentication protocol. *Information Processing Letters*, 56(3):131–136, November 1995.
4. R. Needham and M. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, December 1978.

5. M. Burrows, M. Abadi, and R.M. Needham. A logic of authentication. *ACM Transactions on Computing Systems*, 8(1):18–36, February 1990.
6. Yishi Zhao and Nigel Thomas. Approximate solution of a PEPA model of a key distribution centre. In S. Kounev, I. Gorton, and K. Sachs, editors, *Performance Evaluation: Metrics, Models and Benchmarks. SPEC International Performance Evaluation Workshop (SIPEW 2008)*, LNCS 5119, pages 44–57, Darmstadt, Germany, 2008. Springer.
7. Yishi Zhao and Nigel Thomas. Fluid flow analysis of a model of a secure key distribution centre. In A. Argent-Katwala, N.J. Dingle, and U. Harder, editors, *Proceedings of the 24th UK Performance Engineering Workshop*, pages 160–171, Imperial College London, July 2008.
8. Mirco Tribastone. The PEPA Plug-in Project. In Mor Harchol-Balter, Marta Kwiatkowska, and Miklos Telek, editors, *Proceedings of the 4th International Conference on the Quantitative Evaluation of SysTems (QEST)*, pages 53–54. IEEE, September 2007.
9. Allan Clark. The ipclib PEPA Library. In Mor Harchol-Balter, Marta Kwiatkowska, and Miklos Telek, editors, *Proceedings of the 4th International Conference on the Quantitative Evaluation of SysTems (QEST)*, pages 55–56. IEEE, September 2007.
10. M. Kwiatkowska, R. Mehmood, G. Norman, and D. Parker. A symbolic out-of-core solution method for Markov models. In *Proc. Workshop on Parallel and Distributed Model Checking (PDMC'02)*, volume 68.4 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2002.
11. W.J. Knottenbelt and P.G. Harrison. Distributed disk-based solution techniques for large Markov models. In *Proc. 3rd International Workshop on the Numerical Solution of Markov Chains (NSMC '99)*, pages 58–75, Zaragoza, Spain, September 1999.
12. S. Gilmore, J. Hillston, and M. Ribaudo. An efficient algorithm for aggregating PEPA models. *IEEE Transactions on Software Engineering*, 27(5):449–464, May 2001.
13. Ashok Argent-Katwala, Jeremy Bradley, Allan Clark, and Stephen Gilmore. Location-aware quality of service measurements for service-level agreements. In G. Barthe and C. Fournet, editors, *Proceedings of the Third International Conference on Trustworthy Global Computing (TGC'07)*, volume 4912 of *LNCS*, pages 222–239. Springer-Verlag, 2008.