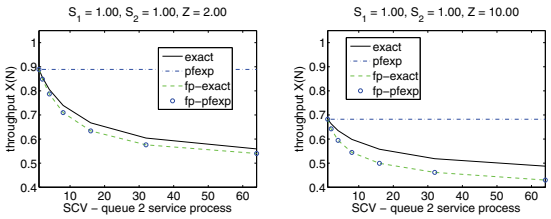
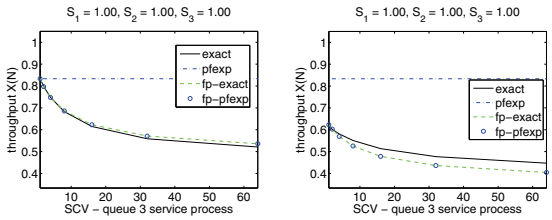


(a) exp+ph, $N = 10$ (b) exp+erl, $N = 20$

Figure 8: Sensitivity to delay stations; the model has $M = 2$ queues and a delay station.



(a) exp+exp+ph, $N = 10$ (b) exp+ph+ph, $N = 10$

Figure 9: Sensitivity to choice of distribution at resource 1; the model has $M = 3$ queues.

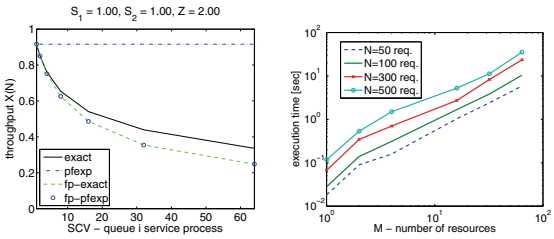
exponential product-form solution. Such a difference from the case in Figure 9(a) may be explained by observing that, when more queues are PH, the request-flows between resources tend to deviate more markedly from a Poisson process. Such a situation cannot be handled by exponential networks[6], whereas our fixed-point method accounts for it effectively, thanks to the scaled input processes $(\boldsymbol{\alpha}_j, \rho_j \boldsymbol{T}_j)$. To test this conjecture, we repeated the experiment in Figure 9(b), replacing the scaled input processes $(\boldsymbol{\alpha}_j, \rho_j \boldsymbol{T}_j)$ by an exponential inter-arrival time with the same mean. Thus, the caudal characteristics η_k are for $M/PH/1$ queues rather than $PH/PH/1$ queues. It is found that the fixed point solution error grows to 37.69% for $SCV = 64$, so it is about 600% larger than with the $PH/PH/1$ approach. For the model in Figure 9(a), the degradation is similar, at 32.9%. These additional experiments provide robust evidence that the proposed approach is effective in describing the distribution of inter-departure times from resources.

Finally, Figure 10 shows sensitivity experiments similar to the ones developed in Figure 5. As before, no major deviations from the exact solution are observed. As with the cases of $M = 2$ resources, for $M = 3$, the execution times for fp-exact and fp-pfexp in all experiments were less than 1 second.

6.3 Large intractable models

We now illustrate the accuracy and scalability of the fixed-point method on models that are intractable by direct solution of the Markov chain. Figure 11(a) shows results for a model with $N = 100$ requests and $M = 10$ resources. The number of states in the underlying Markov process is 4.366×10^{15}, which is clearly intractable. All service times at the queueing resources follow an identical PH(2) distribution with the specified SCV. The exact solution is computed by simulation, using the Java Modelling Tools suite simulator [2], configured with the independent replication method and 95% confidence intervals. Notice that utilization is identical for all resources and equal to the throughput since $S_k = 1$, for $1 \leq k \leq M$. Results indicate that the method is very accurate for $SCV \leq 10$ when the network is more heavily-loaded, whilst it gives slightly larger errors than in the previous examples at large $SCVs$. Overall, the trend is captured fairly well, especially when compared to the pfexp method. We have performed additional experiments and observed that as the bottleneck load grows, performance appears to be captured better at large SCV. Thus, it appears that the proposed approximation tends to perform better at medium/high loads. This is consistent with the discussion in Section 3, since under light load, the different decay rate at queue-length 1 (i.e., $\rho(1-\eta)/(1-\rho)$ as opposed to η) may reasonably become dominant.

[6]Notice that in closed exponential networks the flows are not Poisson either. However, empirical observations suggest that their variability is usually not too far from $SCV = 1$.



(a) 10×ph, $N = 100$ (b)

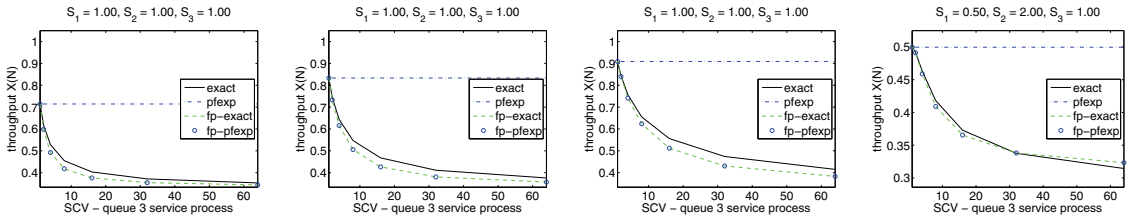
Figure 11: Sensitivity to model size.

Figure 11(b) shows computation times for large models with N up to 500 requests and networks of increasing size, M. In all cases, the fixed-point method can solve the model in a few seconds on a laptop computer. In particular, the execution times scale very efficiently with the population N, thanks to the $O(N)$ complexity of the normalizing constant computation.

7. SERVICE MANAGEMENT EXAMPLE

Finally, we introduce a case study of practical interest for the proposed class of performance models. Consider an application that defines a set of enterprise Java beans (EJBs) to deal with the business logic. Each EJB acquires data from a pool of C connection objects that represent entry-points for web services and database resources, collectively referred to as *data sources*. Assume that there are D data sources and there exist one or more dedicated connection objects for each data source. Thus $C \geq D$ and we denote by C_d the number of connection objects for data source d, $1 \leq d \leq D$. Assume also that queueing delays for outstanding calls at data sources are negligible and that each connection object stores the pending calls it will serve in a first-in-first-out buffer.

We consider the problem of allocating residual bandwidth at run-time by instantiation of new connection objects. Indeed, as the number of connection objects grows, more data can be fetched in parallel per unit of time from data sources and thus increase the application throughput if data acquisition is the performance bottleneck. However, a physical bandwidth limit exists which calls for deciding which data source to prioritize. For simplicity, we focus here on identify-



(a) ph+ph+ph, $N = 5$ (b) ph+ph+ph, $N = 10$ (c) ph+ph+ph, $N = 20$ (d) ph+ph+ph, $N = 10$

Figure 10: Sensitivity to values of mean service time and to the population size; the model has $M = 3$ queues.

ing the data source d^* that would be able to exploit best the greatest portion of the residual bandwidth, and thus acquire more data per time unit.

The decision problem may be tackled by studying a closed queueing network with N circulating requests representing the number of calls issued by EJBs to the connection objects. Define Z_d to be the average time that elapses between completion of a call to data source d and the successive arrival of a new call to that data source. Such a delay may be due to several factors, such as gaps in the arrival stream of requests, time taken by EJBs to process the business logic, level of parallelism for software worker threads used by the application. We model the system as a cyclic network with a first-come-first-served queue, representing the new connection object, a processor-sharing queue, representing available bandwidth, and a delay server with exponential rate Z_d^{-1}. Let R_d be the random variable for the response time of data source d and let $E[B_d]$ be the average data size in bits of a response from data source d over the network. Further, denote by μ the network bandwidth in bits per second and by $\rho_{net,d}$ the current network utilization due to calls to data source d. The average time to transfer data from source d may be estimated as $S_{net,d} = E[B_d]\mu^{-1}$. Since we consider single class models and the network utilization is available at run-time, we consider the scaled quantity $S^*_{net,d} = ((C_d + 1)/C_d)S_{net,d}(1 - \sum_{i \neq d} \rho_{net,i})^{-1}$. Here the utilization scaling factor accounts for the delay due to shared network bandwidth under the assumption that class d will not affect the bandwidth allocated to class $i \neq d$. Instead, the factor $((C_d + 1)/C_d)$ estimates the extra demand placed on the network by a new connection for source d. The example is based on two web service time-traces from the wsdream dataset, having respectively $E[R_1] = 412.43ms$, $SCV[R_1] = 22.35$, $SKEW[R_1] = 9.96$ and $E[R_2] = 661.00ms$, $SCV[R_2] = 0.50$, $SKEW[R_2] = 9.30$, which are fitted by PH(2)s. Thus, data source 1 has high-variability in its response while data source 2 has low variability. The other parameters used in the experiments are given in Table 1.

Notice that source $d = 1$ has the highest demand on network bandwidth and the smallest delay Z_d; thus the choice $d^* = 1$ seems natural. However, we find that low variability makes the choice $d^* = 2$ a better one. Simulation and analytical results are shown in Figure 12. Execution time for the fixed-point algorithm is just $4ms$ per experiment, as opposed to simulation that takes about $10s$ to converge. As we can see, the exponential network model pfexp, which cannot represent high-variability, predicts that an additional connection object would roughly provide the same bandwidth utilization for both data sources, with a slight preference for

$D = 2$	$C_1 = 1$	$C_2 = 1$
d	1	2
N	10	10
Z	$400ms$	$430ms$
$S_{net,d}$	$250ms$	$100ms$
$\rho_{net,d}$	0.5242	0.1518

Table 1: Model parameters.

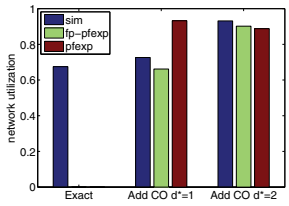


Figure 12: Application example results.

$d = 1$. Conversely fp-pfexp, the fixed-point algorithm initialized with the pfexp solution, correctly predicts that a benefit can be achieved only if the additional connection object is for data source $d = 2$. This is because the high variance of data source 1 would often block the line of requests queueing at the connection object buffer.

Summarizing, this small, but realistic, example shows that the proposed class of models may return surprising – but correct – decisions compared to those suggested by commonly used exponential models. Such predictions are obtained in negligible time and so are compatible with application to run-time decision problems of far greater complexity than that of this exemple.

8. CONCLUSION

We have presented a class of product-form expressions that approximate a diverse range of closed queueing networks with resources having generally distributed processing times. When the skewness of the distribution is not too large (e.g., $SKEW \leq 12$), it was found that the accuracy of the approximation is excellent at all levels of variability, as characterized by the second moment. A fixed-point algorithm that obtains such approximate solutions cheaply in terms of both time and space requirements has been implemented and its application to run-time service management has been illustrated. Future work will focus on the generalization of the proposed method to multi-class workloads as well as load-dependent and multi-server resources.

9. REFERENCES

[1] S. Asmussen and F. Koole. Marked point processes as limits of Markovian arrival streams. *J. Appl. Prob.*, 30:365–372, 1993.

[2] M. Bertoli, G. Casale, and G. Serazzi. User-friendly approach to capacity planning studies with Java Modelling Tools. In *Proc. of SIMUTools 2009*, pages 1–9. ACM, 2009.

[3] D. Bini, B. Meini, S. Steffé, and B. Van Houdt. Structured Markov chains solver: software tools. In *Proc. of SMCTOOLS Workshop*. ACM, 2006, `http://win.ua.ac.be/~vanhoudt/`.

[4] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi. *Queueing Networks and Markov Chains*. 2nd ed., Wiley and Sons, 2006.

[5] J. P. Buzen. Computational algorithms for closed queueing networks with exponential servers. *Comm. of the ACM*, 16(9):527–531, 1973.

[6] G. Casale. A note on stable flow-equivalent aggregation in closed networks. *Queueing Systems*, 60(3-4):193–202, 2008.

[7] G. Casale, N. Mi, and E. Smirni. Bound analysis of closed queueing networks with workload burstiness. In *Proc. of ACM SIGMETRICS 2008*, pp. 13–24. ACM Press, 2008.

[8] G. Casale, N. Mi, L. Cherkasova, and E. Smirni. Dealing with burstiness in multi-tier applications: new models and their parameterization. *IEEE Trans. Software Eng.*, to appears.

[9] G. Casale, M. Tribastone. Fluid Analysis of Queueing in Two-Stage Random Environments. in *Proc. of QEST*, Aachen, Germany, Sep 2011.

[10] P. Courtois. Decomposability, instabilities, and saturation in multiprogramming systems. *Comm. of the ACM*, 18(7):371–377, 1975.

[11] D. L. Eager, D. Sorin, and M. K. Vernon. AMVA techniques for high service time variability. In *Proc. of ACM SIGMETRICS*, pages 217–228. ACM Press, 2000.

[12] E. Gelenbe and I. Mitrani. *Analysis and Synthesis of Computer Systems*. Academic Press, London, 1980.

[13] A. Heindl. *Traffic-Based Decomposition of General Queueing Networks with Correlated Input Processes*. Ph.D. Thesis, Shaker Verlag, Aachen, 2001.

[14] A. Heindl, G. Horvath, and K.Gross. Explicit Inverse Characterizations of Acyclic MAPs of Second Order. Proc. of *EPEW*, Springer LNCS 4054, 108–122, 2006.

[15] H. Kobayashi. Application of the diffusion approximation to queueing networks I: equilibrium queue distributions. *Journal of the ACM*, 21(2):316–328, 1974.

[16] E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik. *Quantitative System Performance*. Prentice-Hall, 1984.

[17] T. Marian, M. Balakrishnan, K. Birman, and R. van Renesse. Tempest: Soft state replication in the service tier. In *DSN*, pages 227–236. IEEE Computer Society, 2008.

[18] N. Mi, Q. Zhang, A. Riska, E. Smirni, and E. Riedel. Performance impacts of autocorrelated flows in multi-tiered systems. *Perform. Eval.*, 64(9-12):1082–1101, 2007.

[19] M. F. Neuts. The caudal characteristic curve of queues. *Adv. Appl. Prob.*, 18:221–254, 1986.

[20] M. F. Neuts. *Structured Stochastic Matrices of M/G/1 Type and Their Applications*. Marcel Dekker, New York, 1989.

[21] B. Pittel. Closed exponential networks of queues with saturation: the Jackson-type stationary distribution and its asymptotic analysis. *Math. Oper. Res.*, 4:357–378, 1979.

[22] M. Reiser. A queueing network analysis of computer communication networks with window flow control. *IEEE Trans. on Communications*, 27(8):1199–1209, 1979.

[23] A. Riska and E. Smirni. MAMsolver: A matrix analytic methods tool. In *Proc. of TOOLS, pp. 205–211*. Springer-Verlag, 2002, `http://www.cs.wm.edu/MAMSolver/`.

[24] A. Riska, V. Diev, and E. Smirni. An EM-based technique for approximating long-tailed data sets with PH distributions. *Perform. Eval.*, 55(1-2):147–164, Jan. 2004.

[25] R. Sadre and B. R. Haverkort. Fifiqueues: Fixed-point analysis of queueing networks with finite-buffer stations. In *Proc. of MMB*, pages 77–80, 1999.

[26] W. Whitt. The queueing network analyzer. *The bell system tech. journal*, 62(9):2779–2815, Nov. 1983.

[27] J. Zahorjan, E. D. Lazowska, and R. L. Garner. A decomposition approach to modelling high service time variability. *Perform. Eval.*, 3:35–54, 1983.

[28] Z. Zheng and M. R. Lyu. Collaborative reliability prediction of service-oriented systems. In *Proc. of ACM ICSE*, pp. 35–44, May 2010.

APPENDIX

A. ASYMPTOTIC DISTRIBUTIONS

The exact asymptotic distribution of the conditional distribution in a $PH/PH/1$ queue is known to exist from equation (2), i.e.,

$$\begin{aligned}\widetilde{\boldsymbol{\theta}} &= \boldsymbol{\theta} = \lim_{n\to\infty} \frac{\boldsymbol{\pi}(n)}{\boldsymbol{\pi}(n)\mathbf{1}} \\ &= \lim_{n\to\infty} \frac{\boldsymbol{\pi}(1)\boldsymbol{R}^{n-1}}{\boldsymbol{\pi}(1)\boldsymbol{R}^{n-1}\mathbf{1}} \\ &= \lim_{n\to\infty} \frac{\boldsymbol{\pi}(1)\eta^{n-1}\boldsymbol{\Pi}}{\boldsymbol{\pi}(1)\eta^{n-1}\mathbf{1}} \\ &= \frac{\boldsymbol{\pi}(1)\boldsymbol{r}}{\boldsymbol{\pi}(1)\mathbf{1}}\boldsymbol{\ell}\end{aligned}$$

Thus, $\boldsymbol{\theta}$ is a left-Perron-eigenvector of $\boldsymbol{R}$ (parallel to $\boldsymbol{\ell}$) and so, by the previous assumption, $\boldsymbol{\theta}\boldsymbol{R} = \boldsymbol{\theta}\eta\boldsymbol{\Pi} = \eta\boldsymbol{\theta}$. Thus $\boldsymbol{\theta}\boldsymbol{\Pi} = \boldsymbol{\theta}$. Notice that $\boldsymbol{\Pi}$ is the spectral projector for the $\boldsymbol{R}$ matrix associated with the *exact* solution $\boldsymbol{\pi}(n)$.

B. PROOF OF PROPOSITION 1

The normalizing constant may be written as

$$G(M,N) = \sum_{S}\prod_{i=1}^{M}[(1-\rho_i)(1-\delta(n_i)) + \rho_i(1-\eta_i)\eta_i^{n_i-1}\delta(n_i)]$$

where $\delta(n_i) = 1$ if $n_i \geq 1$, 0 otherwise. Observe now that

$$\begin{aligned}G(M,N) = &\sum_{(n_1,\ldots,n_M):n_M=0} (1-\rho_M)\prod_{i=1}^{M-1} x_i(n_i) \\ &+ \sum_{(n_1,\ldots,n_M):n_M\geq 1} \rho_M(1-\eta_M)\eta_M^{n_M-1}\prod_{i=1}^{M-1} x_i(n_i) \quad (12)\end{aligned}$$

where $x_i(n_i) = (1-\rho_i)(1-\delta(n_i)) + \rho_i(1-\eta_i)\eta_i^{n_i-1}$. Factoring out of the two summations $(1-\rho_M)$ and ρ_M, respectively, the right-hand side terms are, by definition, $G(M-1,N)$ and $G^{aux}(M,N-1)$, respectively. The terminations conditions also follow immediately by definition of G and G^{aux}. □