Fast Estimation of Probabilities of Soft Deadline Misses in Layered Software Performance Models

Tao Zheng, Murray Woodside
Dept. of Systems and Computer Engineering, Carleton University, Ottawa Canada
1- (613) 520-5721
{zhengtao | cmw} @sce.carleton.ca

ABSTRACT

Quality of service requirements are normally given in terms of soft deadlines, such as "90% of responses should complete within one second". To estimate the probability of meeting the target delay, one must estimate the distribution of response time, or at least its tail. Exact analytic methods based on state-space analysis suffer from state explosion, and simulation, which is also feasible, is very time consuming. Rapid approximate estimation would be valuable, especially for those cases which do not demand great precision, and which require the exploration of many alternative models.

This work adapts layered queueing analysis, which is highly scalable and provides variance estimates as well as mean values, to estimate soft deadline success rates. It evaluates the use of an approximate Gamma distribution fitted to the mean and variance, and its application to examples of software systems. The evaluation finds that, for a definable set of situations, the tail probabilities over 90% are estimated well within a margin of 1% accuracy, which is useful for practical purposes.

Keywords

Performance Engineering, Software performance, Soft deadlines, Queueing delay distributions, Layered Queuing Network Models

1. INTRODUCTION

For network systems with Quality-of-Service requirements, and for many people specifying performance requirements, it is most natural to specify target response times with a minimum success rate, such as "90% of responses should complete within one second". However practical estimation using analytic models is commonly forced, by limitations of the models, to give mean value results. This paper addresses a simple approximation to estimate the tail part of a distribution.

The exact delay distribution can be found for queueing network models, particularly if jobs do not overtake each other during a response. For open models, early work was done by Wong [19], and fast approximation has been given by Mainkar et al in [12]. For closed models exact solutions have been given by Harrison [9], McKenna [13], and others, and approximations for other structures were described by Salza and Lavenberg [17], Raatikainen [15] and others. However for software systems, *extended* queueing network models are often needed, to which the various approaches described above do not apply.

For general Markov models such as Stochastic Petri nets, delay distribution is a classic passage time problem. Methods for computing it have been described by Muppala et al [14] using direct transient probability solution, and by Dingle et al [4] using Laplace transforms. This work has been extended to Semi-Markov processes by Bradley et al [3]. The approaches based on Petri nets and Markov chains apply to our systems but they do not scale up well enough for many practical problems. Further, it is desirable to be able to treat systems with non-exponential service, which increase the state space of Markov chain models, or require semi-Markov models. However in semi-Markov models all delays must be resampled for every transition in the global state space. When many resources are in use simultaneously in a complex system, this is unrealistic, as there will be many transitions and resamplings during a single resource holding time.

Fast methods to estimate percentiles appear to require queueing models (provided they can be applied), some kind of approximate Mean Value Analysis (MVA), and approximations to the distribution fitted to the moments. Approximate distributions fitted to moments have been described for transition system models, by Gautama and Gemund [7][8], and by Au Yeung et al [1]. These authors used four moments and the Generalized Lambda Distribution [11], which is a flexible function for empirical distributions. However mean value queueing analysis does not easily (if at all) provide four moments. As an alternative we propose the Gamma distribution, which has two parameters and has special cases (such as the exponential and Erlang distribution) which occur frequently in the exact analysis of queues. The layered queueing analysis has been adapted to provide two moments.

This contribution of this work is to evaluate the use of the approximate Gamma distribution, fitted to the mean and variance computed by approximate MVA of the layered queues. It is evaluated in two ways, for accuracy of estimation of distributions found by simulation, and in use to compute the sensitivity of deadline results to many parameters.

WOSP'05, July 12-14, 2005, Palma de Mallorca, Spain.
ISBN 1-59593-087-6/05/0007.

2. LAYERED QUEUEING

The Layered Queueing Network (LQN) model combines queueing analysis with an architectural view of a system and the interactions between its components. An example LQN model for a web server is shown in Figure 1. The architectural aspect shows a set of components called "tasks" (rectangles with heavy borders) that offer services labeled as "entries" (rectangles with normal borders). Tasks are servers, and their entries define classes of service with parameters that define the workload for each class. Processors (ovals), networks and other devices are also servers that are included in this architectural view, and are treated the same way. Figure 1 shows four layers including User.

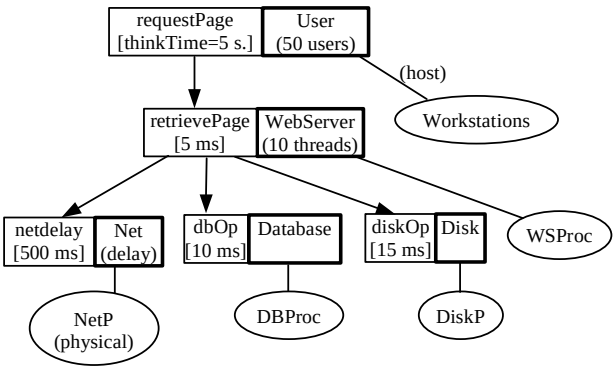

Figure 1 A LQN model for a web server

Directed arcs in Figure 1 represent requests for service. A request from one entry to another may return a reply to the requester (a synchronous request), or be forwarded to another entry for later reply, or may not return any reply (an asynchronous request). All the requests in Figure 1 are synchronous, indicated by a solid-headed arrow and meaning that the requester waits for a reply. The task resource is held by the transaction for this time, so the blocking time becomes part of the service time of the entry. Requests queue to obtain the server, and then perform the operation defined by the entry they have called.

The applied workload is modeled by a task which does not accept requests and which has a multiplicity equal to the number of users, such as the User task in Figure 1. The response time in this system is the time for a User request to queue and obtain service from the WebServer. The WebServer service time is the holding time of one thread until the final acknowledgement, and includes the lower services. A long retrievePage service time can produce saturation at the WebServer task if all threads are busy, even if the web server processor is unsaturated; this is called a "software bottleneck".

Analytic approximations have been developed to speed up the analysis of large systems and large sets of alternative models. Analytic LQN solvers, such as LQNS ([5][6][16][20]), construct queueing submodels for clusters of servers at different layers and apply a fixed-point iteration to the submodels, to find a steady-state solution for delays and resource utilizations. Within each layer submodel, LQNS applies standard MVA approximations such as the Schweitzer approximation (see e.g. Bolch et al [2]) to solve the model, with additional special approximations to deal with non-exponential service with multiclass FIFO queues, multiservers, server holding times that continue after the reply (so-called "second phases"), and parallel branches within a service [5].

The analytic approximations have been evaluated against simulation in [5] and in many applied studies. The general experience is that errors are less than 5% in throughputs and 10% in most delays, which makes the approximations useful in practice for searching a design space with many cases to be tested. Simulation is still useful for checking accuracy, and for cases where the approximations fail.

Most LQN servers at higher layers display markedly non-exponential service times, depending on the demands they make to their lower levels. This is natural enough if a service time includes random numbers of requests to lower services that have different response times, since a mixture of distributions like this tends to have a high coefficient of variation CV, defined as:

$$CV^2 = \text{variance}/\text{mean}^2$$

compared to an exponential distribution (which has CV = 1).

Variance Estimation

In general, if we denote a service time as the random variable *s*, its variance is determined as follows. We suppose it is made up of random numbers of requests to *n* different lower level classes of service, plus its host processor (called class 0).

For the *i*th of these service classes, let the random service delay be x_i with mean X_i and variance VXi. The *j*th sample of the class-*i* delay occurs in the service time sum as x_{ij} which can be matched with the host service time x_{i0} just after it, giving a delay term $(x_{ij} + x_{i0})$, with *j* running from 1 to y_i. The number y_i is itself random with mean Y_i and variance VY_i. Thus:

- $s = x_0 + \sum_{i=1}^{n} \sum_{j=1}^{y_i} (x_{ij} + x_{i0})$

If y_i and x_{ij} are independent random variables, then classical probability theory gives

- $E\{ x_{ij} + x_{i0} \} = X_i + X_0$
- $Var\{(x_{ij} + x_{i0})\} = (VX_i + VX_0)$
- $E\{s\} = S = X_0 + \sum_{i=1}^{n} Y_i * X_i$
- $Var\{s\} = VS$

$$= VX_0 + \sum_{i=1}^{n} \left(Y_i (VX_i + VX_0) + VY_i (X_i + X_0)^2 \right)$$

Two special cases are considered in the current version of the LQNS solver. The "deterministic service" case has a constant (integer) number of requests to each lower service class which gives $VY_i = 0$ and $VS = VX_0 + \Sigma_i Y_i (VX_i + VX_0)$.

The "random service" case has a geometrically distributed number of requests to each lower service class, which gives the variance VY_i in terms of the mean, as $VY_i = Y_i(Y_i + 1)$. Then:

$$VS = VX_0 + \sum_{i=1}^{n} \left(Y_i (VX_i + VX_0) + Y_i (Y_i + 1)(X_i + X_0)^2 \right)$$

The variance of the response time of the *i*th service/class combination includes contention at the server, so it is the sum of its waiting time w_i and its service time s_i :

$$x_i = w_i + s_i$$

The approximation used here makes two assumptions, motivated by gaps in the information about the joint distribution of s_i and w_i . First, they are assumed to be independent, which is true of many single queues, but not in general of networks; second, w_i is assumed to be exponentially distributed, which appears to be a good assumption for many queues.

Under these two assumptions, $VW_i = W_i^2$ and we can use:

$$VX_i = W_i^2 + VS_i$$

This completes the information needed to compute the variance of the service time of a higher-layer entry, from the queueing results and the mean and variance of its lower-layer server entries.

3. THE APPROXIMATE DISTRIBUTION

The Gamma distribution is a two-parameter distribution with the density function:

$$f(x) = \frac{x^{\alpha-1}e^{-x/\theta}}{\Gamma(\alpha)\theta^{\alpha}}$$

and the moments:

$$mean = \alpha\theta\ ,\ variance = \alpha\theta^2$$

Special cases include the exponential (when CV = 1) and the Erlang-k, when $CV^2 = 1/k$.

The parameters of an approximating Gamma distribution can be fitted to a random variate using its mean and variance, as

$$\theta = variance/mean,\quad \alpha = mean^2/variance$$

In layered systems the response times are frequently "hyper-exponential" in the sense that $CV^2 > 1$, as discussed above. There also exist cases, typically with a fixed numbers of requests to one kind of operation which itself is relatively deterministic, in which CV^2 may be less than 1.

3.1 The Fitted Distribution and its Accuracy

The goodness of fit of the approximate Gamma distribution is demonstrated in Figure 2 for the web server system shown in Figure 1. Figure 2 shows the estimated cumulative probabilities based on a histogram with 20 cells, captured from a simulation, as the diamond-shaped points labeled p_i.

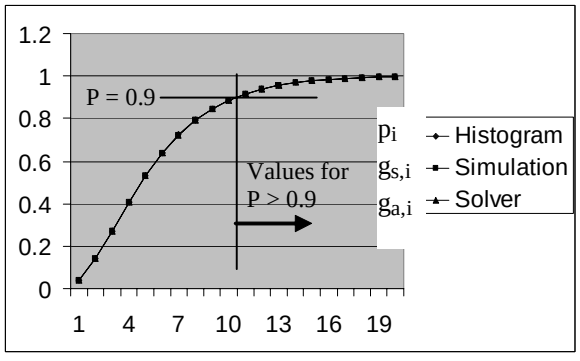

Figure 2 Comparison of the Fitted and Measured Distributions

The approximation is shown by two fitted curves which are virtually identical, one (labelled $g_{s,i}$) fitted to the mean and variance of the simulation results and the other (labelled $g_{a,i}$) to the analytic solver results.

The former shows that the Gamma approximation gives an excellent fit to the measured distribution, while the latter includes the additional errors introduced by using the approximate moments from the analytic solution (which are negligible in this example). The goal of this work is to evaluate $g_{a,i}$, but the simulation-based fit will help distinguish the source of errors, in cases where they are significant.

Figure 3 plots the absolute errors for both approximations, against the value of p_i. The errors are $e_{s,i} = g_{s,i} - p_i$ (the diamonds) and $e_{a,i} = g_{a,i} - p_i$ (the squares). Sometimes the former are larger, and sometimes the latter. Larger errors occur for smaller p_i, (and this appears to be commonly true). Since we are most concerned with the tail of the distribution and systems with large probabilities of achieving their target delay, we concentrate on errors for $p_i > 0.9$. In both figures a line is drawn to indicate those cases, and the largest absolute error for $p_i > 0.9$ is denoted M(0.9|s) or M(0.9|a), indicated on the Figure 3.

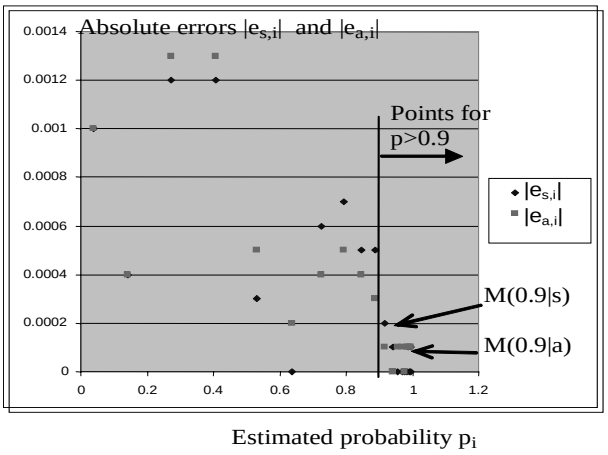

Figure 3 the Magnitude of errors for different probabilities

3.2 "Overloaded" servers

The Gamma distribution has a density function with a single maximum and a long tail. It is not appropriate for approximating densities of bounded random variables, or densities with multiple peaks, such as the response time distribution in Figure 4. This type of distribution can appear in a service system if it contains (at least one level down from the entry server) a saturated server with a large fixed population of customers, and one response contains a variable number of requests to this server. The saturated server has a long queue of nearly constant length (nearly all its customers), so the central limit theorem makes its response time roughly normally distributed (corresponding to one peak). Multiple visits per response give multiple peaks. The distribution function has a series of steps at the peaks, with plateaus between. It clearly cannot be approximated very well by a Gamma distribution.

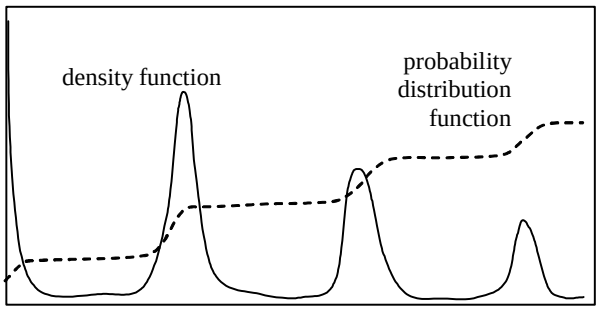

Figure 4. Response distribution including multiple requests to a "overloaded" server, with decreasing probability.

The cases with a large nearly constant queue length, which are seen to be unsuitable for the present approximation, will be called "overloaded" servers in the remainder of this paper. As a rule of thumb, an "overloaded" server has:

- utilization over 99%,
- more than 15 clients, or if it is a multiserver, more than 15 times as many clients as servers.

The seriousness of this limitation is reduced by the fact that a system with such a saturated queue is probably not a candidate for a "good" configuration either.

3.3 Approximation Errors

If the distribution is to be approximated, we should consider what is a reasonable and useful accuracy of the approximation, in the most interesting range of probabilities from 0.9 – 1.0.

Our target for this work was to achieve errors that are usually less than ± 0.01 in the success or miss probability. We have evaluated errors across many cases as RMS (root-mean-square) errors, which we compare to 0.01. The comparison is very system-dependent.

The accuracy of the error estimate also depends on the accuracy of the estimates of p_i. The 95% confidence intervals for an estimated probability value p, based on a binomial distribution for the counts below the cell boundary, are $\pm 1.96\sqrt{p(1-p)/N}$ [10]. The simulations for this system had more than 2 million responses, giving confidence intervals on all the estimated probabilities of less than ± 0.0007, and for probabilities over 0.9, less than ± 0.0004.

4. EVALUATION OF ACCURACY OF THE GAMMA APPROXIMATION

To evaluate the approximation accuracy on a wide range of cases, several sample system topologies were set up, and a large set of parameter variations were constructed using systematic variations and random choices of some parameters.

4.1 Web Server Example

The first example is the web server model in Figure 1. It represents a class of network service systems for which QoS specifications with soft deadlines are becoming increasingly important.

216 combinations of 3 parameters (the number of users, the threading level of WebServer and the squared coefficient of variation of the CPU execution) were evaluated by simulation and by the analytic approximations.

As we are most interested in the tail of the distribution, errors were found for probability values above 0.9. The RMS and maximum errors across the 216 cases are shown in Table 1, denoted as RMS(0.9|s), RMS(0.9|a) , M(0.9|s) and M(0.9|a).

For cases in which the server utilization is less than 99% (that is, not "overloaded" in the sense defined above), the Gamma approximation was very good. Ignoring the "overloaded" cases, the maximum error in the tail is less than 0.005, and the RMS errors are less than 0.002 from solver-based data. The errors of the simulation-based data are even smaller.

Table 1 Summary of the WebServer case results for different server utilizations, over 216 cases

WebServer Utilization	cases	RMS (0.9|s)	RMS (0.9|a)	M(0.9|s)	M(0.9|a)
0.0 --0.6	162	2.31E-4	2.61E-4	7.0E-4	0.0012
0.6 --0.9	24	2.84E-4	7.24E-4	8.0E-4	0.0043
0.9 --0.95	6	3.75E-4	0.0015	6.0E-4	0.0043
0.95--0.99	6	6.54E-4	0.0016	0.0010	0.0042
0.99--1.00 (overloaded)	18	0.0037	0.0573	0.0075	0.105

4.2 Large "interconnected" example

This example represents a complex web of relationships among

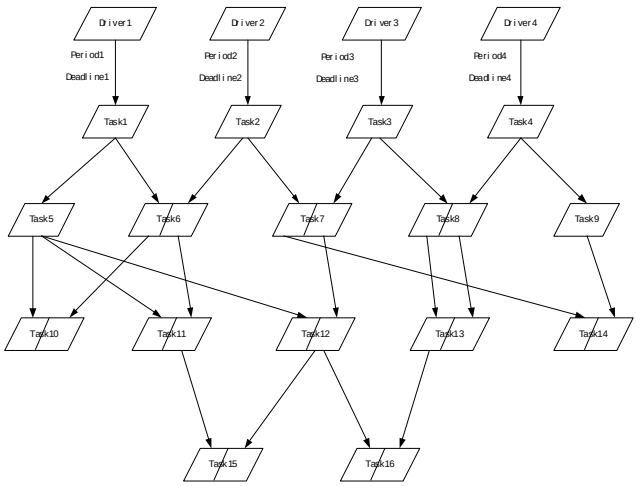

Figure 5 Large "interconnected" example

16 single-threaded tasks in four layers, shown in Figure 5, with randomly generated parameters. Delays for each task are made up of complex combinations of lower-layer delays. The single-threaded tasks do not impose enough workload on lower layers to create any single bottleneck, and utilizations are moderate.

The structure of this example is similar to that used in an example in [21], but here it has all blocking interactions.

210 cases were generated, with various parameters including a range of the squared coefficients of variation of the processor-execution demand of the tasks, ranging from 0.5 to 2.0.

The approximation errors for the Gamma distribution are shown in Table 2. The fit to the moments found by simulation is excellent, which again indicates that the approximation is appropriate for these distributions. The fit to the analytic moments is less good, and the larger errors are due to the errors in approximating the variance in the analytic solver. However, the analytic RMS(0.9) errors are better than 0.01, and the maximum errors are less than 0.03 for the tail of the distribution.

Table 2 Summarized results of the "complex web" example

Maximum Utilization	cases	RMS (0.9|s)	RMS (0.9|a)	M(0.9|s)	M(0.9|a)
0.0 --0.6	164	0.0019	0.0086	0.0088	0.0292
0.6 --0.9	46	0.0022	0.0101	0.007	0.0263

In summary, the Gamma distribution fits the distributions well enough, but the analytic approximation to the variance is barely good enough. This is a challenge to improve the calculation of the variance.

5. EXPERIENCE FOR EXPLORATION OF SYSTEM PROPERTIES

An important use of a model is for optimization or to explore the sensitivity of performance measures to parameters. Important parameters include the CPU demand of operations, processor speed and multiplicity, number of storage operations, process replication or threading parameters, and network latencies. A straightforward way to find a sensitivity value is to change a parameter by a small amount, solve the model again, and observe the change in the performance measures (in the present work, in the target delay miss rate). In a large system this may require hundreds of solution runs.

Here we consider the sensitivity of the deadline miss probability to a range of parameters in a substantial model of a trunk-to-trunk telephone switch [18].

The model was used in [18] to investigate the effectiveness of analytic modeling for rapid solution of large models, using mean response times. The LQN is summarized in Figure 6, showing 23 tasks representing a voice-over-packet switch which connects lower level switches together. There is a Call Connection agent for an originating switch, a gateway for it, a core switch (represented very abstractly), and a terminal gateway.

When a soft deadline measure of call connection within 15 sec. was applied to the model, the base case had a success rate of 82%. The sensitivity of this success rate to the model parameters was sought, in order to find the most promising way to improve the success rate.

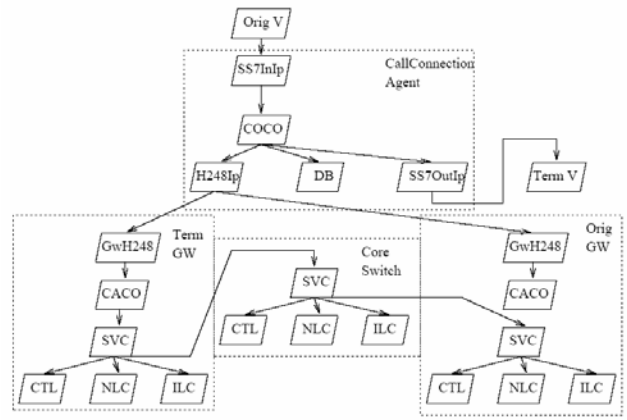

Figure 6 A LQN for a Trunking Gateway Voice-over-Packet Switch

The execution demands and threading levels of the tasks were perturbed for 66 model parameters, giving a total of 67 model solutions. The normalized sensitivities to the parameters were calculated, being the ratio of fractional change in performance, divided by fractional perturbation in the parameter. Thus for a parameter value Y that gave a base success probability P, the sensitivity of P to Y is defined as:

$$S_{PY} = (\Delta P/P) /(\Delta Y/Y)$$

From the 67 model solutions, the sensitivities were found and then ordered by value, to give a sensitivity map of the parameter space. The most sensitive parameters are the ones that give the greatest promise for performance improvement, so this is a valuable tool for discovering performance opportunities, and for evaluating some kinds of changes before making an effort to implement them.

Space prevents a full description, but among the 66 S_{PY} values, the sensitivity to the threading level of the COCO (ConnectionControl) task had the largest value, of 0.27. This predicts that an additional thread (over the initial 10 threads) will improve the success probability by about 2.7%. Other threading sensitivities and execution demand sensitivities are much smaller.

This is consistent with the analysis in the thesis [18] which identified the thread pool size of the COCO task as a bottleneck, and confirmed it by adding threads. This demonstrates the use of sensitivity analysis for diagnosis.

The value of the analytic solver (compared to simulation), for this evaluation, is indicated by the time taken to find the solutions. Simulation to give a 1% accuracy in the mean response time required more than 5 minutes per run, while the entire 67 analytic solutions required about 5 minutes in total, a 60 – to – 1 advantage.

6. CONCLUSIONS

This paper has described a simple approximation to the tail part of the probability distribution of delay in a type of Extended Queueing Network model called Layered Queueing Networks (LQNs). It found that the approximation is usable (giving errors less than 0.01) for systems structured like network service systems, which are not "overloaded". This approximation opens the door to the use of fast MVA approximations for distributed

software systems that have delay requirements on soft deadlines, rather than mean values.

This work shows that more accurate variance approximations for MVA could give immediate gains in QoS calculations, as the most important source of error in the approximate distribution appears to be the variance calculation. This points a direction for research on iterative MVA solution algorithms, such as a Schweitzer algorithm for variance.

7. ACKNOWLEDGMENTS

This research was supported by grants from NSERC, the Natural Sciences and Engineering Research Council of Canada, and from IBM.

8. REFERENCES

[1] S. W. M. AuYeung, N. J. Dingle, and W. J. Knottenbelt, "Efficient Approximation of Response Time Densities and Quantiles in Stochastic Models," in *Fourth Int. Workshop on Software and Performance (WOSP 04)*, Redwood City, CA, Jan. 2004, pp. 151-155.

[2] S. G. G. Bolch, H. de Meer, K. S. Trivedi, *Queueing Networks and Markov Chains.* John Wiley and Sons, 1998

[3] J.T. Bradley, N. J. Dingle, P. G. Harrison, and W. J. Knottenbelt, "Distributed Computation of Transient State Distributions and Passage Time Quantiles in Large Semi-Markov Models", Future Generation Computer Systems, 2004

[4] N. J. Dingle, P. G. Harrison, and W. J. Knottenbelt, Response Time Densities in Generalised Stochastic Petri Net Models, *in 3rd Int. Workshop on Software and Performance (WOSP 2002*), Rome, July, 2002, pp. 46-54

[5] G. Franks, *Performance Analysis of Distributed Server Systems*, PhD, Dept. of Systems and Computer Engineering, Carleton, Ottawa, Canada, 2000

[6] G. Franks, S. Majumdar, J. Neilson, D. Petriu, J. Rolia, and M. Woodside, "Performance Analysis of Distributed Server Systems", Proc. Sixth International Conference on Software Quality (6ICSQ), Ottawa, Ontario, 1996, pp. 15-26.

[7] H. Gautama and A.J.C. van Gemund, "Static Performance Prediction of Data-Dependent Programs," in *Proceedings of the Second International Workshop on Software and Performance (WOSP2000)*, Ottawa, Canada, September 17-20, 2000, pp. 216-226.

[8] H. Gautama and A. J. C. v. Gemund, "Symbolic Performance Prediction of Speculative Parallel Programs," *Parallel Processing Letters*, vol. 13, no. 4 pp. 513-524, Dec 2003.

[9] P. G. Harrison, "The Distribution of Cycle Times in Tree-Like Networks of Queues," *Computer Journal*, vol. 27, no. 1 pp. 27-36.

[10] R. Jain, *The Art of Computer Systems Performance Analysis*. John Wiley & Sons Inc., 1991

[11] Z. Karian and E. Dudewicz, *Fitting Statistical Distributions: The Generalized Lambda Distribution and Generalized Bootstrap Methods*. CRC Press, Boca Raton, 2000.

[12] V. Mainkar, K. S. Trivedi, S. Woolet, "Fast Approximate Computation of Response Time Distribution in Open Markovian Network of Queues", *7th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation*, May 3-6, 1994 Vienna, Austria

[13] J. McKenna, "Asymptotic expansion of the sojourn time distribution functions of jobs in closed product-form queueing networks," *J. of ACM*, vol. 34, pp. 985-1003, 1987

[14] J. Muppala, K. S. Trivedi, V. Mainkar, and V. Kulkarni, "Numerical Computation of Response Time Distributions Using Stochastic Reward Nets," *Annals of Operations Research*, vol. 48, pp. 155-184, 1994.

[15] K. Raatikainen, "Approximating Response Time Distributions," in *Proc. ACM Sigmetrics 89*, Berkeley, CA, 1989, pp. 190-199.

[16] Rolia, J.R., Sevcik, K. The method of layers. *IEEE Transactions on Software Engineering*, Vol. 21, No. 8, pp. 689-700, 1995.

[17] S. Salza, S.S. Lavenberg, "Approximating Response Time Distributions in Closed Queueing Network Models of Computer Performance", *Proc. Performance '81*, F.J. Kylstra (ed), North-Holland Publishing Company, 1981

[18] P. Tregunno, *Practical Analysis of Software Bottlenecks*, M.A.Sc thesis, Dept. of Systems and Computer Engineering, Carleton University, May 2003.

[19] Wong, J. W., "Distribution of End-to-End Delay in Message-Switched Networks," *Computer Networks*, Vol. 2, 1978.

[20] Woodside, C.M. " Throughput calculation for basic stochastic rendezvous networks". *Performance Evaluation*, Vol. 9, No. 2, pp. 143-160, 1989.

[21] T. Zheng and M. Woodside, "Heuristic Optimization of Scheduling and Allocation for Distributed Systems with Soft Deadlines," in *Proc. 13th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation (TOOLS 03*), Urbana, USA, Sept. 2003