INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. (in press)
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/dac.881

On the application of forking nodes to product-form queueing networks[‡]

Essia H. Elhafsi[1,*,†], Mart Molle[1] and D. Manjunath[2]

[1]*Department of Computer Science and Engineering, University of California Riverside, Riverside, CA 92521, U.S.A.*
[2]*Department of Electrical Engineering, Indian Institute of Technology, Bombay, Powai Mumbai 400 076, India*

SUMMARY

We define a 'forking node' as a service centre with one input feeding two outputs (each served by its own queue) under the control of an internal path-selection (PS) policy. We assume that both outputs lead to paths through which a packet reaches its final destination. However, the mean downstream delays on the two paths may be different and the PS policy should favour the path with the lower downstream delay. Using simulation, we compare the performance of this system under a variety of random, deterministic, state-dependent PS policies, including threshold-based and join-shortest-queue with bias (JSQ + b). We show that JSQ + b has better performance than the other alternatives. Moreover, if the input process to the forking node is Poisson, standard time series analysis techniques show that its two outputs are very close to being independent Poisson processes. Thus, if we find an accurate and efficient 'offline' analytical performance model for JSQ + b forking node, we can extend the applicability of product-form queueing networks to include such forking nodes. For this reason, we present several ways of modelling the performance of a JSQ + b node, using bounds, and compare their results on example networks. We establish a closed-form expression relating the bias b and the delays of the downstream paths. Copyright © 2007 John Wiley & Sons, Ltd.

Received 2 February 2006; Revised 18 January 2007; Accepted 21 January 2007

KEY WORDS: asymmetric networks; parallel servers; path selection policy; performance bounds; optimization

1. INTRODUCTION

In this paper, we model the performance of networks with routing algorithms that distribute traffic over multiple paths. Multipath routing is important to several application domains. For example, if

*Correspondence to: Essia H. Elhafsi, Department of Computer Science and Engineering, University of California Riverside, Riverside, CA 92521, U.S.A.
†E-mail: essia@cs.ucr.edu
‡This paper is the extended version of the one presented in SPECTS'05 [1].

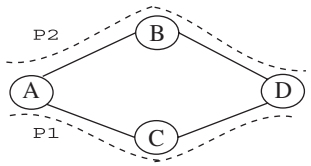

Figure 1. A network in which there are two paths, P_1 and P_2, from the forking node A, to the destination node D. The path-selection algorithm at node A is aware of the mean delay on each path, and uses this information to determine how much traffic to send over each path.

the IP routers within an autonomous system use a link-state routing algorithm (such as OSPF [2]), then traffic to each destination must be equally distributed among the available equal-cost paths. Similarly, if the Flow Deviation Algorithm [3] is used to optimize routing for a given network topology, then traffic will in general be randomly distributed over multiple paths. And, finally, multipath routing can be used to maximize bandwidth for a multi-homed node (such as the border router for an autonomous system with uplinks to several ISPs, or a mobile device that is equipped with both 802.11 and Bluetooth interfaces, say).

Using the terminology of queueing network models, our goal is to study *forking nodes* (such as node A in Figure 1), which are service centres with one input and two outputs. More specifically, we wish to find path-selection (PS) policies for forking nodes that: (i) are easily implementable in real network switches; (ii) provide good performance; and (iii) can be incorporated into tractable analytical network performance models.

In addition to being easy to implement and providing good performance, a good PS policy should also allow us to develop analytically tractable performance models for optimizing the performance of the network. In this context, product-form queueing networks are very attractive since they can provide precise and detailed performance results such as queue length distribution, average response time, resource utilization and throughout, with relatively low computational complexity [4]. Thus, we will focus our attention on the relationship between PS policies and the product-form solution.

Jackson [5] and Gordon and Newel [6] introduced the product-form property for open and closed queueing networks with exponential inter-arrival and service time distributions, while Kleinrock [7] was the first to apply such models to the analysis of packet-switched data networks. They have shown that such queueing networks have a closed-form solution for the equilibrium state probabilities and its structure is the product of factors describing the state of each node in the network.

Definition

Consider a queueing network of J queues where $\pi_i(s_i),\ i = 1, 2, \ldots, J$ is the equilibrium probability that queue i is in state s_i (the number of waiting packets in the queue). Let $\pi(s)$ be the probability that the network is in state $s = (s_1, s_2, \ldots, s_J)$. Then, the network has product-form solution if the equilibrium probability that the network is in state s is expressed by

$$\pi(s) = G \prod_{k=0}^{J} \pi_i(s_i) \tag{1}$$

Here, G is the normalizing constant.

DOI: 10.1002/dac

Muntz [8] investigates the closely related $M \Rightarrow M$ (Markov implies Markov) property, which states that Poisson arrivals imply Poisson departures. Muntz shows that a queueing network has a product-form solution if all nodes in the network satisfy the $M \Rightarrow M$ property. In [9], it is shown that local balance is a necessary and sufficient condition for product-form solution.

The literature with regards to networks having product-form solution is rich [9–12]; the application to computer systems and their widespread queueing disciplines may be credited to Chandy *et al.* [4]. They present an approximate, iterative method for determining performance values of closed queueing networks with first come, first served discipline and general service times. The method is based on an application of Norton's theorem from electrical circuit theory to queueing networks which satisfy local balance.

In [10], queueing network models were extended to open, closed and mixed queueuing networks with several job classes, generally distributed service times and different queueing disciplines. Towsley [13] considers a closed queueing network model with state-dependent policy. The routing function is a rational function of the queue length of various downstream queues. He shows that the introduction of routing will preserve the product-form of the equilibrium distribution if the network with no state dependent policy has product-form. Nelson [14] discusses the mathematics leading to the product-form results and the properties of the stochastic process underlying the network model.

However, most practical queueing problems lead to non-product-form networks such as networks with non-exponentially distributed services times, networks that use routing that depends on the state of other parts of the network, computer systems and networks with asymmetric nodes with simultaneous resource possessions (systems with memory constraints, or with I/O subsystems), models of programs with internal concurrency, and fork-join operations in parallel processing systems. Note that nodes with such features would not be difficult to solve in isolation, but they cause problems when added to a larger multi-queue system because they destroy the product-form property. Thus, approximation such as the flow equivalent server (FES) [15, 16] have been developed to handle such networks. In general, such techniques are approximations that reduce the model to a similar system that has product-form.

Single nodes in isolation, on the other hand, may have certain features that makes it difficult to solve even without trying to integrate them into a multiple-node queueing network model. This includes nodes with a single input linked to two (local) queues through a state-dependent scheduler, a single queue feeding multiple servers where some are not turned on unless the queue size exceeds some threshold. To avoid complex solutions and costly simulation, approximate procedures have been considered. For the simple join the shortest queue problem, Adan, Wessels and Zijm used a technique referred to as the 'compensation approach' to derive a solution. First for symmetric case [17] (queues with the same service rate $\mu_1 = \mu_2$) and later for the asymmetric case [18] ($\mu_1 \neq \mu_2$). The solution is determined to be an infinite sum of product. In [19], a shorter queue model where in addition to the arrival stream of λ that is routed to the shorter queue, two parallel queues are also fed by independent arrival streams. Assuming that the two queues have the same service rate, and using diffusion models, the solution to the steady-state distribution is approximated and is shown to have product-form.

In this paper, we are trying to find a good solution to handle both issues simultaneously, namely a queueing network model where we want to find an accurate and efficient global solution even though some nodes (forking nodes) behave in a way that could be difficult to model in isolation. We approximate the solution to the stationary probabilities of the system by computing upper and lower bounds. We show that these bounds are tight and that the network can be

DOI: 10.1002/dac

approximated to a product-form queueing network. The remainder of this paper is organized as follows.

In Section 2, we present the problem of interest to this work and its underlying assumptions. In Section 3, we consider a forking node that uses separate output queues for each link. We describe a variety of PS algorithms for this system including static policies (such as simple random PS) and some dynamic threshold-type policies (such as join-the-shortest-queue, JSQ). We also introduce JSQ$+b$ a generalization of the JSQ policy. In Section 4, using simulation we show that threshold-type policies perform much better than the alternatives; JSQ$+b$ being the best and random policy the worst. In Section 5, we analyse the output processes generated by the various PS schemes to determine which ones are compatible with the product-form solution based on Muntz' $M \Rightarrow M$ condition. In Section 6, we describe a two-dimensional model of the two-queue JSQ$+b$ model and present bounds for the number in each queue and the mean overall delay. We also consider a threshold-type single-queue system as a one-dimensional approximation to the JSQ$+b$ two-queue model and compare its behaviour to the two-dimensional model. This is similar to the two heterogeneous-server system considered by Koole [20], Lin and Kumar [21] and Viniotis and Ephremides [22], except in our case we have homogeneous servers and heterogeneous 'paths' behind each server. In Section 7, we present simple closed-form expressions for the stationary probabilities of the single-queue model. In Section 8, we study the sensitivity of the overall delay to the bias b. We provide a closed-form expression for optimizing the bias as a function of delay difference between the two downstream paths. In Section 9, we present a simple application of our result to a non-product-form queueing network and compute relevant statistics. Finally, we conclude with a summary.

2. PROBLEM DESCRIPTION

We consider modelling the performance of queueing networks with *forking nodes* such as node A in Figure 1. To simplify the problem, we make the following assumptions about the environment. Packet arrivals to the forking node form a Poisson stream[§] with rate λ. The two outputs from the forking node, which we refer to as LINK 1 and LINK 2, operate at the same speed. Furthermore, both output links lead to paths by which an arriving packet can reach its final destination. However, the remainder of the network transit delays are different for the two paths.

As an example, consider the network given in Figure 1. Assume that all the traffic from A is routed to D. Moreover, assume that there are two paths from A to D, a direct path A–D and a multihop path A–B–D. In this example, all the traffic from A to D has a choice between both paths. However, since packets that use path A–B–D are subject to a store-and-forward delay at the intermediate node B, the remainder of the network transit delay will be much shorter if a packet is routed along the direct path A–D than path A–B–D. Therefore, we are interested in finding a PS policy that reduces the end-to-end delay by favouring the faster path.

We assume that the downstream delay for the path beyond LINK i is d_i, $i = 1, 2$, and that $d_1 \geqslant d_2 \geqslant 0$. Without loss of generality, we can subtract d_2 from the downstream delays for both paths and thus reduce the model to having no extra delay along path 2 and a downstream delay of

[§] We are well aware of the extensive literature about long-range dependence in network traffic. However, it has been shown in [23] that the traffic on highly multiplexed high-speed backbone links looks increasingly like a Poisson process.

DOI: 10.1002/dac

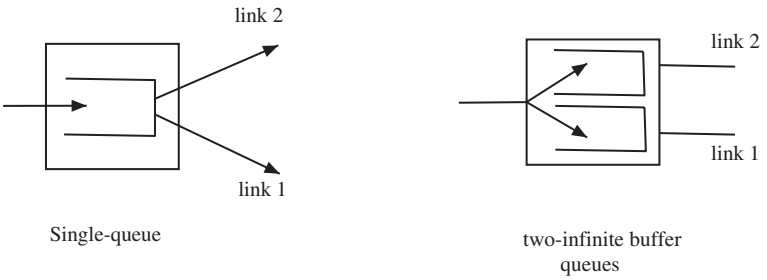

Figure 2. Queueing options at the forking node A.

$d = d_1 - d_2$ along path 1. Throughout this work, we assume that the delay difference is independent of the state of the forking node and hence will be characterized by its mean d.

There are several options for designing the internal structure of a forking node; the node can use a single shared input queue or a separate queue for each output link as shown in Figure 2. If the node uses a single shared input queue, then the PS decisions can be delayed until a packet is at the head of the queue. Conversely, if the node uses separate output queues for each link, then the PS decisions must be made when a packet arrives.

Despite the fact that only the shared queue option can support work conservative scheduling algorithms, network switches generally use separate queues per output port. Single shared queue switches are viewed to have poor performance due to the head-of-the-line blocking [24]. However, recent developments in networking and the dramatic increase of line rates has generated an interest in shared queue switches.

We model the forking node with different queueing structures.

3. PATH SELECTION POLICIES

We consider PS policies that base their packet assignment decisions on a local view of the system, such as the queue lengths at the forking node for each of the available paths. Random, deterministic and state-dependent policies are considered. These policies can be further classified as biased or unbiased. Unbiased policies do not favour one path *versus* the other over the long term. However, if multiple paths to the destination are available, it seems likely that the remainder of the paths downstream delay from the forking node will be different for those alternate paths. Therefore, when the delay difference d is large, which is generally the case, such information should be incorporated in the PS decision as a bias in favour of the path with the lower downstream delay.

We consider the following PS policies.

- *Random policy*: An incoming packet is randomly routed to one of the two queues based on a random coin toss. Biased policy can be obtained by varying the weight of the coin.
- *Round robin*: Incoming packets are routed to the queues deterministically, according to a strict alternation pattern. Biased policy, with rational weight u/v, can be obtained by increasing the cycle length and assigning u out of every v packets to queue 1.
- *JSQ*: An incoming packet is routed to the output link that has a smaller number of waiting packets at its arrival epoch. This greedy algorithm has many optimality properties. It is

DOI: 10.1002/dac

known to be socially optimal [25] over all non-preemptive policies that use knowledge of the distribution (but not the actual value) of each customer's service time, i.e. it minimizes the discounted expected sojourn time of the packets in the system and maximizes the discounted number of jobs to complete their service in any specified time interval [26]. Biased policy is obtained by introducing a queue imbalance threshold, b. Thus, in the JSQ$+b$ policy we route an arriving packet to path 1 if the difference in the instantaneous queue lengths between the two queues exceeds b, i.e. $(j-i)>b$ where i is the length of the queue in front of LINK 1 and j is the length of the queue in front of LINK 2. We refer to b as the PS or routing bias. JSQ$+b$ is known to be optimal [27]; it minimizes the average end-to-end delay of a packet.

- *Virtual waiting time* (*WT*): This scheme is similar to JSQ, except that the PS decision is made on the basis of the actual service times for each customer, and not just their instantaneous queue length. Biased policy can be obtained by sending a new arrival to path 1 if the difference in virtual WT for the two sets of servers exceeds b/μ.

4. COMPARISON OF THE PATH SELECTION POLICIES

We compare the PS policies in terms of the mean end-to-end delay. We simulate[¶] the system for various combinations of d and network loads ρ.

Figure 3 shows that state-dependent policies perform well compared to random and round robin policies when $d=0$. We studied different versions of random and round robin policies as shown in Figure 3, which 'randomly' favour one of the queues. So rather than splitting the traffic between the queues we force only 40% (33%) of the traffic to use one of the queues using random (round robin) scheme. As a result, their performance decreases drastically especially at high network loads. Based on Figure 3, JSQ policy performs better than the alternative policies when the paths have the same downstream delay.

Figure 4, for $d=20/\mu$ where $1/\mu=10^{-3}$ ms and the network load ρ between 0 and 1, shows that random policy has the worst delay followed by round robin. We also tested different versions of random and round robin policies. For biased random policy, we force 40% of the traffic to use the slower path. For round robin, we impose that about 33% of the traffic uses the slower path. This improved their performance especially for low loads. Biased policies, however, have a better performance at all network loads. JSQ$+b$ and the virtual WT policies were found to have the best performance. Similar results are obtained for other values of d. Note that for biased policies, we compute the average overall delay using the optimal routing bias ($b=b^*$) obtained by simulation.

Similarly, Figure 4 shows that JSQ$+b$ policy provides the best performance over all PS policies when our system model includes some additional path-independent delay beyond the forking node. This improvement is clearly visible in Figure 5 which shows the mean delay as a function of b at $\rho=0.5, 0.7$, for several values of the extra delay d behind queue 1. Moreover, Figure 5 shows that the delay curves seem to be a convex function in b. Note that in all cases, the minimum delay occurs for some $b>0$. Moreover, the optimum value of b/μ is always much smaller than d, and becomes even smaller as ρ increases. It is interesting to note that the optimal solution to the standard parallel queue PS problem ($d=0$) is a 'greedy' policy, i.e. JSQ. However, when we generalize the problem by adding some extra delay behind one queue ($d\geqslant 0$) then the 'greedy'

[¶]We used Csim 19 [28] for our simulations.

DOI: 10.1002/dac

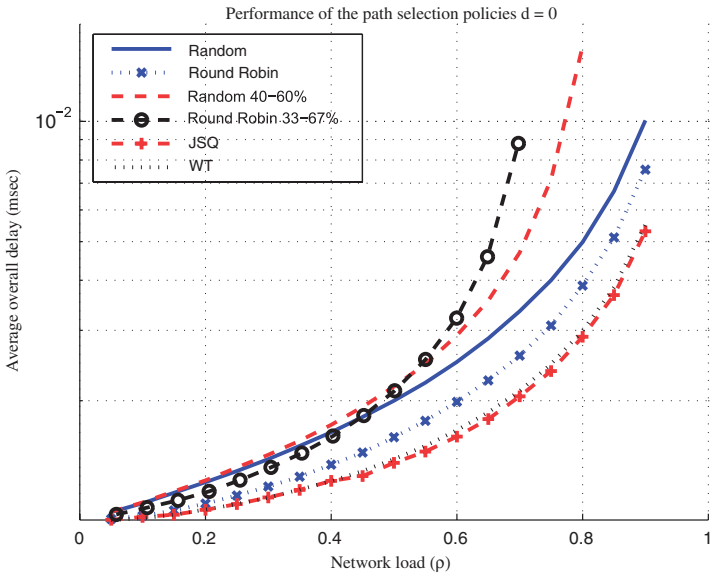

Figure 3. Performance of unbiased policies when the downstream delay behind the queues is the same ($d = 0$, $1/\mu = 10^{-3}$ ms).

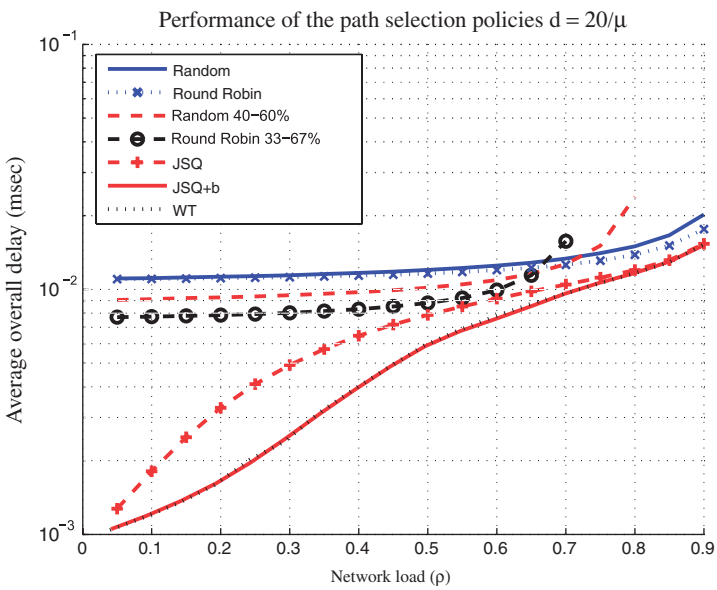

Figure 4. Comparison of several PS policies. The mean delay in a two-queue system when there is some extra delay behind one of the queues ($d = 20/\mu$, $1/\mu = 10^{-3}$ ms).

DOI: 10.1002/dac

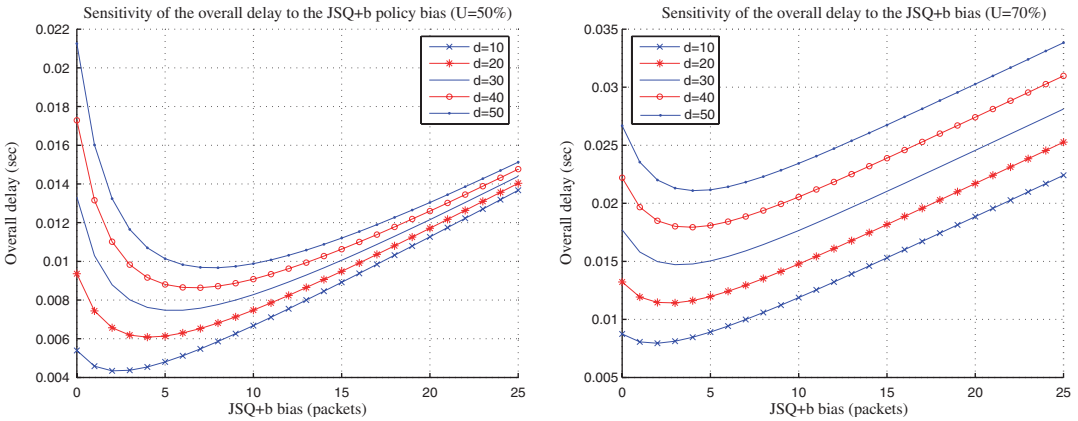

Figure 5. Reducing the mean delay *via* biased PS policies when there is some extra delay behind Q_1. Network load $\rho = 50, 70\%$, $(d = (10, 20, 30, 50)/\mu)$, $1/\mu = 10^{-3}$ ms.

policy, i.e. JSQ + b with $b/\mu = d$, is clearly *not* optimal in this case. Our objective in the next section is to provide an expression for the 'optimal' PS bias as a function of d.

5. RELATION TO PRODUCT-FORM

We use the $M \Rightarrow M$ property, due to Muntz [8], to determine whether forking nodes with various PS policies can be incorporated into a product-form queueing network model. A service centre satisfies the $M \Rightarrow M$ (Markov implies Markov) property if feeding it a Poisson arrival process causes it to produce a Poisson departure process. If every service centre in a queueing network satisfies the $M \Rightarrow M$ property, then we have a sufficient condition for showing that its solution has a product-form. Based on this result, we simulated a forking node and applied time-series analysis to its two output processes to test for goodness of fit between the observed inter-departure times distribution and the exponential distribution, and evaluated the autocorrelation and cross-correlation functions (as a test for independence). Note that this methodology is *not* intended as a rigorous proof that the product-form does or does not hold for the PS policies we studied. Instead, our goal is to decide whether or not we can use a product-form solution to approximate the performance of a network that includes forking nodes.

For each of the PS policies, we simulate the system with various utilizations ($\rho = 10$–90%) and different path downstream delays (d). For the threshold based PS policies, we study the system using the optimal PS threshold, b^*, that we obtain by simulation. We study the distribution of packet inter-departure times from both queues (Q_1 and Q_2) and show that it can be described by an exponential distribution for certain PS schemes. Furthermore, we show that the inter-departure times are independent for other PS policies. For some policies such as the round robin, the results do not show that product-form holds.

5.1. *Inter-departure times distribution*

To show that the inter-departure times distribution from both Q_1 and Q_2 is exponential, we use the complementary cumulative distribution function (CCDF). The CCDF is defined as

DOI: 10.1002/dac

$F^{c}(t) = 1 - F(t)$, where $F(t)$ is the cumulative distribution function. The CCDF of an exponential distribution with mean $1/\lambda$ is

$$F^{c}(t) = e^{-\lambda t}, \quad t \geqslant 0$$

The CCDF of packet inter-departure times is a straight line when the Y-axis is plotted in log scale, which corresponds to an exponential distribution.

The CCDF of inter-departure times distribution from Q_1 and Q_2 for a network 70% utilized and using the threshold based and random PS schemes are given in Figure 6(a) and (b), respectively. We tested networks with various loads and linear least-square fitting shows that the CCDFs for all network utilization we tested can be described by an exponential distribution with confidence over 95% for all PS policies with the exception of the round robin scheme.

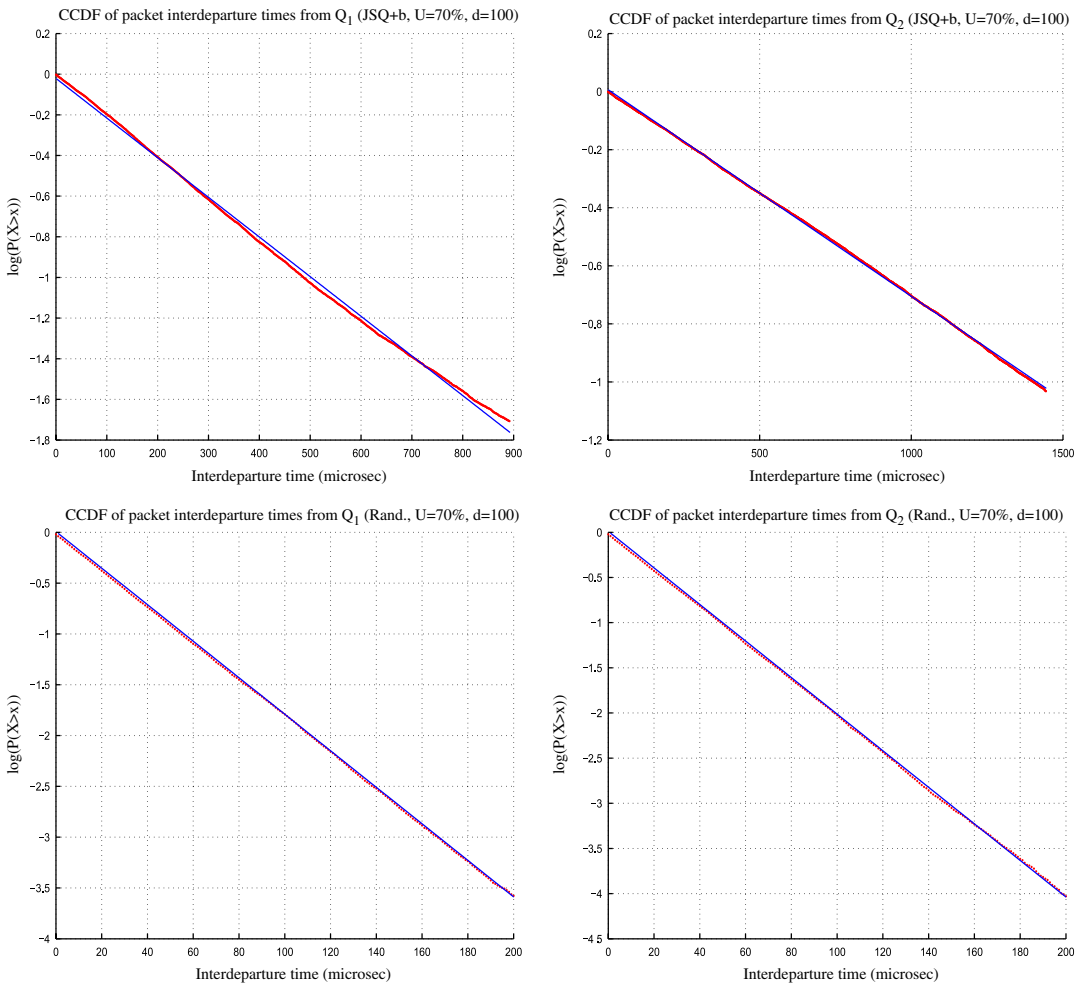

Figure 6. CCDF of packet inter-departure times from Q_1 and Q_2. The distributions can be well approximated by an exponential distribution: $\rho = 70\%$, $d = 100/\mu$. (b) CCDF of packet inter-departure times—random policy.

DOI: 10.1002/dac

Table I gives a more detailed approximation of the inter-departure times distribution from each queue, for various network utilizations, to an exponential distribution in terms of confidence levels (%).

5.2. Correlation in inter-departure times

The inter-departure times correlation is captured by the autocorrelation function (ACF), $\rho_{\text{acf}}(k)$ which measures the dependency between a series X_t and a shifted version of itself X_{t+k}:

$$\rho_{\text{acf}}(k) = \frac{E[(X_t - \mu_X)(X_{t+k} - \mu_X)]}{\sigma_X^2}$$

where μ_X and σ_X are the sample mean and standard deviation, respectively.

The correlation of inter-departure times between queues is also captured by the cross-correlation (XCF) which measures the dependency between two different series $X_{1,t}$ and $X_{2,t}$ shifted

$$\rho_{\text{xcf}}(k) = \frac{E[(X_{1,t} - \mu_{X_{1,t}})(X_{2,(t+k)} - \mu_{X_{2,t}})]}{\sigma_{X_{1,t}}\sigma_{X_{2,t}}}$$

where $\mu_{X_{1,t}}$ and $\sigma_{X_{1,t}}$ are $X_{1,t}$'s mean and standard deviation, respectively, and $\mu_{X_{2,t}}$ and $\sigma_{X_{2,t}}$ are $X_{2,t}$'s mean and standard deviation, respectively.

The closer the $\rho_{\text{acf}}(k)$ and $\rho_{\text{xcf}}(k)$ are to zero, the better is the assumption of independence. For various values of the downstream delay d and for all PS policies, we computed the ACF for packet inter-departure times, from Q_1 and Q_2. We also computed the queues cross correlation (XCF). These statistics were computed for 200 lags for over 50 000 consecutive packet inter-departure times.

Figure 7 gives plots of the ACF, of the packet inter-departure times using threshold PS, from Q_1 and Q_2 for a downstream delay d, $d = 100/\mu$. It also shows the XCF of packet inter-departure times from both queues. Similar plots are given in Figure 7 for the random policy.

We computed the error, that is the fraction of data outside of the 95% confidence interval bounds, for the auto correlation of the inter-departure times from Q_1, Q_2 and for the cross-correlation of the inter-departure times from both queues. In all cases, except for the round robin, the error is less than or equal to 5% (Table II), so with a 95% confidence level, our simulation results show that the autocorrelations are very close to zero. Note that in certain cases, not enough data are available to compute the error (due to low load in the system or not enough packets join a particular queue, in this case Q_1).

5.3. Satisfaction of the $M \Rightarrow M$ property

We showed that for a variety of network utilizations the inter-departure times are exponentially distributed with a confidence level of at least 95%, with the exception of the round robin policy. We also showed that for most policies, the inter-departures from each queue and cross-correlations also pass the test for independence. We conclude that it is very likely that the departure processes from both queues are (or almost are) Poisson. We cannot generalize for sure that the $M \Rightarrow M$ property holds for all policies. However, we can claim with confidence that a network using any of the threshold PS policies we tested can be approximated to having a product-form solution. Furthermore, the threshold PS under test have a much higher performance than random PS that has product-form. Therefore, we will base our further analysis of the system on JSQ + b PS policy.

DOI: 10.1002/dac

Table I. Confidence level to approximate the interdeparture times to an exponential distribution for all PS policies.

		CI ($d = 20/\mu$)		CI ($d = 100/\mu$)	
Policy	ρ	Q_1	Q_2	Q_1	Q_2
Virtual	10	—*	96.0	—	94.8
waiting	20	—	95.2	—	95.6
time	30	—	97.9	—	95.4
	40	97.7	94.4	—	94.9
	50	98.3	96.4	97.7	96.6
	60	99.5	95.9	96.0	94.3
	70	98.6	94.6	98.8	95.0
	80	98.4	97.5	99.5	95.1
	90	98.6	95.6	98.4	95.6
JSQ + b	10	—	95.2	—	94.8
	20	—	97.3	—	95.0
	30	—	97.3	—	95.2
	40	97.5	96.7	—	95.4
	50	98.1	97.4	99.1	95.0
	60	99.6	96.3	98.4	96.9
	70	98.9	97.0	98.6	95.8
	80	98.2	97.1	98.7	95.1
	90	98.3	96.6	99.2	95.0
JSQ	10	98.0	95.7	97.3	95.7
	20	96.6	96.4	97.7	97.6
	30	98.4	96.9	98.8	96.2
	40	98.6	96.6	98.1	95.6
	50	99.2	97.2	98.7	97.2
	60	98.0	95.2	97.5	95.3
	70	97.9	95.1	99.2	96.7
	80	98.7	96.8	96.8	95.0
	90	98.0	97.9	98.5	95.0
Random	10	97.7	97.2	98.1	98.0
	20	95.5	95.0	97.9	96.6
	30	95.1	97.0	96.5	96.0
	40	96.8	97.2	96.8	96.5
	50	96.9	96.3	97.7	96.4
	60	97.4	96.5	95.8	95.5
	70	95.2	96.7	97.1	98.2
	80	96.5	96.2	97.1	95.4
	90	97.4	95.9	96.5	95.9
Round	10	93.0	89.8	94.2	93.2
robin	20	97.0	95.2	90.6	92.6
	30	94.8	93.6	90.1	90.1
	40	96.2	96.9	93.3	87.8
	50	92.4	97.3	91.2	95.3
	60	95.9	91.8	91.7	93.0
	70	95.5	96.0	93.5	95.5
	80	94.7	92.5	96.7	97.2
	90	94.9	96.9	91.9	97.5

*Not enough data to compute statistics.

DOI: 10.1002/dac

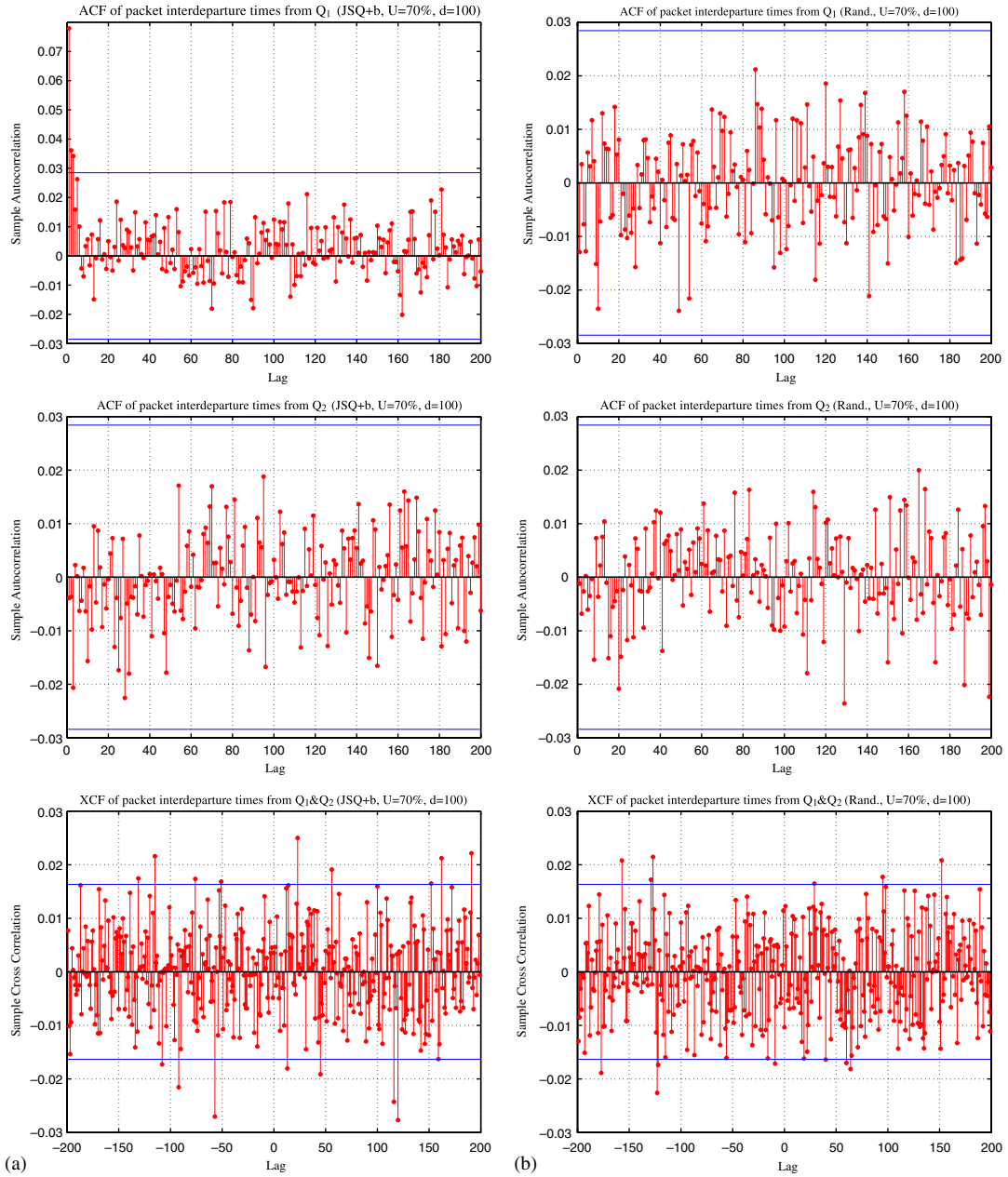

Figure 7. Autocorrelation function of packet inter-departure times from Q_1, Q_2 and sample cross-correlation (left to right). All the correlation coefficients are within the 95% confidence intervals except for a small number of coefficients. $\rho = 70\%$ and $d = 100/\mu$: (a) correlation—JSQ + b policy and (b) correlation—random policy.

DOI: 10.1002/dac

Table II. Error (%) in the autocorrelation of the interdeparture times from Q_1 (ε_1) and from Q_2 (ε_2) and in the cross-correlation (ε_c).

		$d = 20/\mu$			$d = 100/\mu$		
Policy	ρ	ε_1	ε_2	ε_c	ε_1	ε_2	ε_c
Virtual	10	−1*	0	−1	−1	0	−1
waiting	20	−1	0	−1	−1	0	−1
time	30	0	0	0	−1	0	−1
	40	0	0	0	−1	0	−1
	50	0	0	0	0	0	0
	60	0.5	0	0	0	0	0
	70	0.5	0	0	0.5	0	0
	80	0	0	3.0	1.5	0	1.5
	90	0	0	4.5	0.5	0	1.7
JSQ + b	10	−1	0	−1	−1	0	−1
	20	−1	1.2	−1	−1	0.5	−1
	30	0	2.1	0	−1	0	−1
	40	0	0	0	0	0	2.2
	50	0	0.7	0	0	0	0
	60	0	0.9	0	0	0	0
	70	0	0	0.2	0	0	2.0
	80	0	0	1.5	0	0	1.0
	90	0.5	0	3.2	0	2.9	2.0
JSQ	10	0	0.5	0	0	0	0
	20	0	0.5	0	0	0.5	0
	30	0.5	0.5	0	0	0.5	0
	40	0	0.5	0	0	0.5	0
	50	0	1	0	0	0.5	0
	60	0	1	1.5	0	1	1.0
	70	0.5	1	1.0	0	0.5	1.7
	80	0	0.5	2.0	0.5	0.5	3.0
	90	0	0.5	3.0	0.5	0	3.0
Random	10	0	0	0	0	0	2.7
	20	0	0	2.2	0	0	2.5
	30	0	0	2.2	0	0.5	2.0
	40	0	0.5	3.7	0	0	3.2
	50	0	0	2.7	0	0	2.0
	60	0	0	3.2	0.5	0.5	2.7
	70	0	0	4.5	0	0	2.5
	80	0	0	2.7	0	0	1.0
	90	0	0	4.0	0	0	1.7
Round	10	0	0	3.2	0	0	4.2
robin	20	0.5	0	5.7	0.5	0	6.5
	30	0	0.5	5.7	0	0.5	5.7
	40	0.5	0.5	3.7	0.5	0.5	7.0
	50	0.5	0.5	4.0	1	1	5.5
	60	0.5	0.5	6.2	1	0.5	7.0
	70	0.5	1.5	7.2	0.5	1.5	7.2
	80	0	0.5	6.7	0.5	0.5	6.0
	90	0	0	6.2	0	0	6.0

*Not enough data to compute statistics.

DOI: 10.1002/dac

6. MODELLING THE JSQ + b FORKING NODE

The promising results of Section 5 indicate that a forking node with the JSQ + b PS policy can be incorporated into product-form queueing networks without compromising the accuracy of the product-form solution algorithm. Therefore, in this section we consider several alternatives to model the local behaviour of a forking node.

The first approach would be to find an exact closed-form expression for the stationary probabilities. Unfortunately, even though the simpler JSQ (without bias) has been studied since at least 1958, explicit formulas for its stationary probabilities have not yet been obtained [29]. We were able to obtain simple expressions for the stationary probabilities in the case of heavy traffic [30]. Since we need the stationary probabilities over all operating conditions, here we consider several methods for finding good approximations rather than exact solutions.

6.1. Two-dimensional state space

Let $Q_i(t)$ be the number of packets in queue i $(i = 1, 2)$ at time t, including the packets being transmitted (if any). In this case, we can use $\mathbf{Q}(t) := [Q_1(t), Q_2(t)]$ to represent the state of the system at time t. Clearly, $\mathbf{Q}(t)$ evolves as a two-dimensional continuous-time Markov chain (CTMC) in $\mathscr{Z}^{+2}$, with state transitions as shown in Figure 8. In particular, because of the JSQ + b PS policy, an arriving packet will be directed to LINK 2's queue iff $Q_2(t) - Q_1(t) \leqslant b$. We call the line defined by $Q_2(t) - Q_1(t) = b$ in the state space of $\mathbf{Q}$ its *attractor line*. Note that for all states that are *not* on this line, a packet arrival transition always moves the state towards the *attractor line*. Thus, for all states to the left of the *attractor line*, an arrival is routed to LINK 2. Similarly, for all states to the right of the *attractor line*, an arrival is routed to LINK 1.

If we assume that $\rho \equiv \lambda/2\mu < 1$, then the Markov chain $\mathbf{Q}(t)$ will be ergodic. Let π_{q_1,q_2} denote the stationary probability of $\mathbf{Q}(t) = [q_1, q_2]$. Given the difficulties of the analysis of the basic

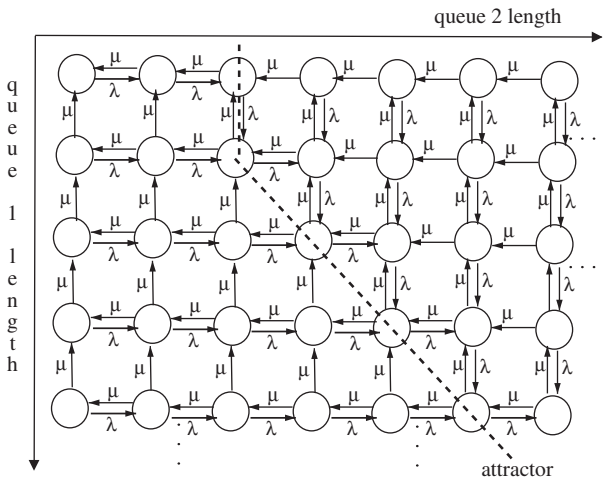

Figure 8. The transition diagram for the CTMC $\mathbf{Q}(t)$. The *attractor line* is shown for $b = 1$.

DOI: 10.1002/dac

JSQ system, we do not anticipate finding an easy solution to the more-general JSQ + b system. Hence, we now focus on the problem of finding good approximations, rather than an exact solution.

6.2. Approximate solution via two-dimensional bounds

Following the method of [31, 32], separate Markov chains can be constructed whose solutions can be shown to, respectively, upper and lower bound the stationary probabilities. In [31, 32], it is shown that suitable modifications of the transitions at the edges of the state spaces can in fact yield tight upper and lower bounds of the stationary probabilities. These stationary probabilities in turn have a closed-form expression that can be easily computed. Performance statistics like mean and variance of the system are also amenable to closed-form and computable solutions. The Markov chains over the truncated state space form a quasi-birth–death (QBD) process that can be solved using matrix geometric techniques. We refer to this bounding method as the 'Two-queue bounding procedure'.

We followed the procedure in [32] to compute bounds of the stationary probabilities and consequently bounds of the number in the queues and in the system, which are shown in Figure 9. The bounds are very tight at low and average loads.

We also compute bounds of the mean overall delay based on a model we discuss in Section 6. The results are shown in Figure 10. Note that the bounds approximate the actual delay very well, especially for low and average network loads.

6.3. Approximating the system by a single-queue model

Due to the complexity of the two-dimensional system described in the previous subsection, we now consider the alternative strategy of approximating the system by a simpler one with similar performance, rather than approximating the solution to the original system. In this case, we consider a single-queue model shared by two links and serviced by a threshold PS policy. Each link has a single server, and all the service rates are identical and equal to μ. The path through LINK i is assumed to have a downstream delay of d_i, $i = 1, 2$, where $d_1 \geqslant d_2 \geqslant 0$. We assume that the queue has an infinite buffer capacity and that the arrival process is Poisson with rate λ. New arrivals join the queue and are scheduled for transmission as follows.

- If LINK 2 is free, a packet is transmitted on this link.
- If LINK 2 is busy but LINK 1 is free, then a packet is transmitted on LINK 1 only if the queue length is greater than the threshold b.

The CTMC given in Figure 11 describes the transition behaviour of the system. Let $Q(t)$ denote the system state at time t where $Q(t)$ takes on values as follows:

$$Q(t) = \begin{cases} 0 & \text{queue is empty} \\ i,\ i \geqslant (b+1) & i \text{ in system; both links busy} \\ i,\ 0 < i \leqslant b & i \text{ in system; LINK 2 busy} \\ i',\ 1 < i' \leqslant b & i \text{ in system; both links busy} \\ i',\ i' = 1 & i \text{ in system; LINK 1 busy} \end{cases}$$

DOI: 10.1002/dac

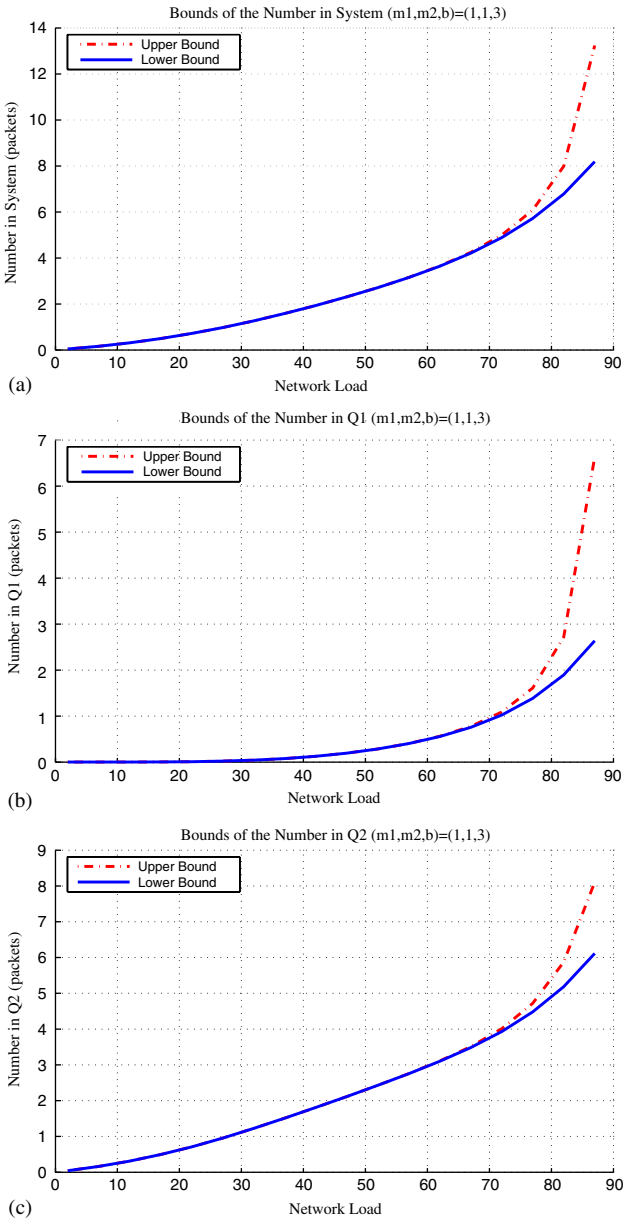

Figure 9. Bounds on the number in the queues and in the system for the two queue infinite buffer model when the PS threshold is $b=3$ and $d=50/\mu$: (a) number in the system; (b) number in Q_1; and (c) number in Q_2.

DOI: 10.1002/dac

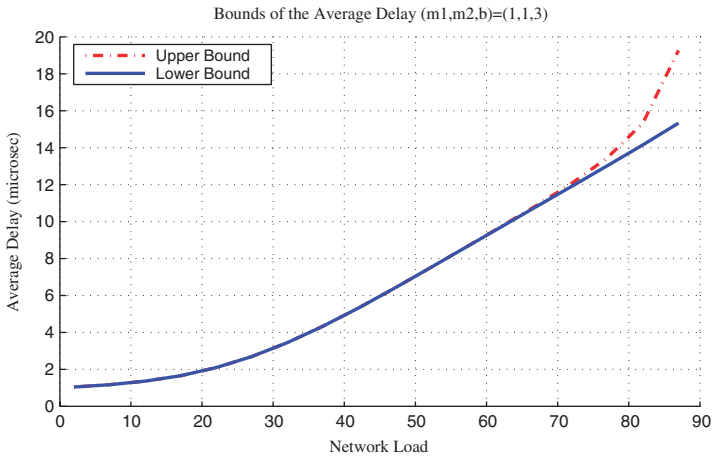

Figure 10. Bounds of the mean delay in the system for two infinite buffer queues when the PS threshold is $b = 3$ and $d = 50/\mu$.

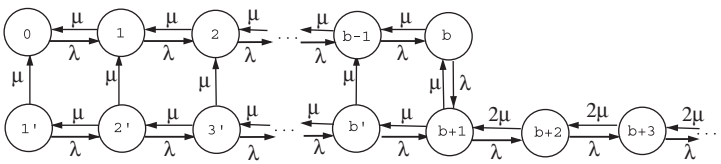

Figure 11. The CTMC for the single-queue model with PS threshold b.

6.4. *Comparing single-queue versus two-queue results*

Before we describe efficient solution techniques for the single-queue model, we must first demonstrate that approximating the system in this way gives us performance results of sufficient accuracy in comparison to the two-dimensional model. Thus, we now compare the results of these two models to see how well they agree in terms of predicted mean delay and the optimal PS bias.

We define T to be the average end-to-end delay from the forking node to the target destination. Let f be the proportion of time that LINK 1 will be idle. Then f can be expressed as follows:

$$f = \begin{cases} \sum_{i=0}^{b+1} \pi_i & \text{single-queue model} \\ \sum_{j=0}^{\infty} \pi_{0,j} & \text{two-queue model} \end{cases} \tag{2}$$

Note that $(1 - f)\mu$ is the mean departure rate along the slower path. On the other hand, the combined departure rate through both paths will be λ if the system is ergodic. Thus, a fraction

DOI: 10.1002/dac

$(1-f)\mu/\lambda$ of the packets entering the forking node will be subject to the additional downstream delay on the slower path.

We let N be the average number of packets within the forking node, including the ones in service. Using Little's law, we have the following overall delay model:

$$T = \frac{N}{\lambda} + (1-f)\frac{\mu}{\lambda}d \tag{3}$$

Numerically we compute the steady-state probabilities of both models by brute force (i.e. $\pi = \pi P$, where P is the transition matrix of the corresponding CTMC) to get the probability LINK 1 is idle. Based on Equation (3), we compute the overall mean delay for various network loads ($\rho = 50, 70$ and 80%).

Figure 12 gives a plot of both models overall mean delay as a function of the downstream delay. The results show that for high loads and high d, the delays are very close. For low loads and low values of d, the delays are not as close as for high loads but the gap is acceptable. In all cases, the single-queue delays are lower than the two-queue delays, which is expected.

We also tested both system on how well they estimate the optimum routing bias (Figure 13). We use Equation (3) to find the optimum PS bias (b^* that minimizes the mean delay) for various network loads. Though Equation (3) does show a direct relationship between T and b, however, there is a dependency since f is a function of b.

Figure 14 presents a plot of both systems estimates of b^*. It is clear that the estimates are very close; in the worst case they are off by one packet. Note that when the delay is zero, it is clear that the optimal routing bias is also zero. In this case, packets will be subjected to the

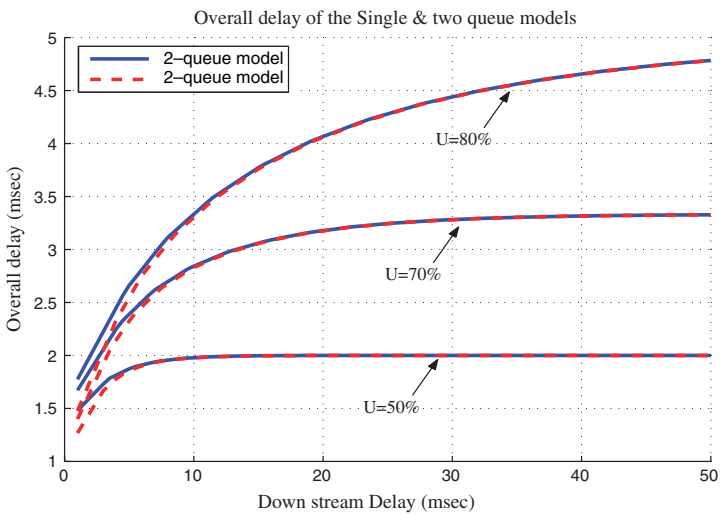

Figure 12. Comparative results: overall mean delay of the single queue and the two-queue models ($\rho = 50, 70$ and 80%).

DOI: 10.1002/dac

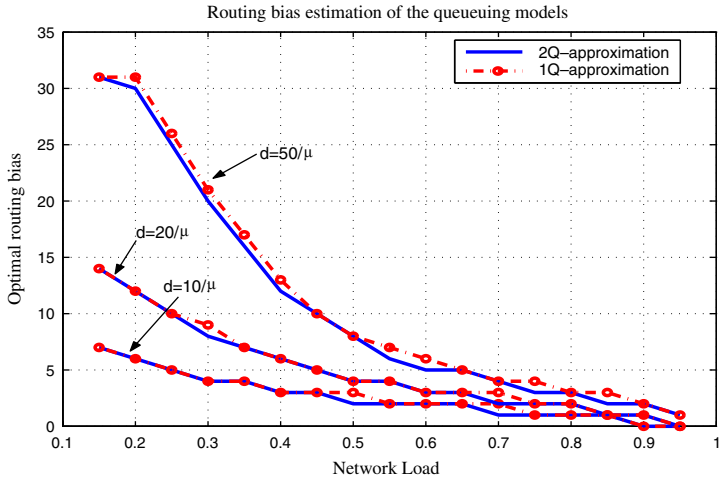

Figure 13. Comparative results: optimum routing bias estimation of the single-queue and the two-queue models ($\rho = 50$ and 80%).

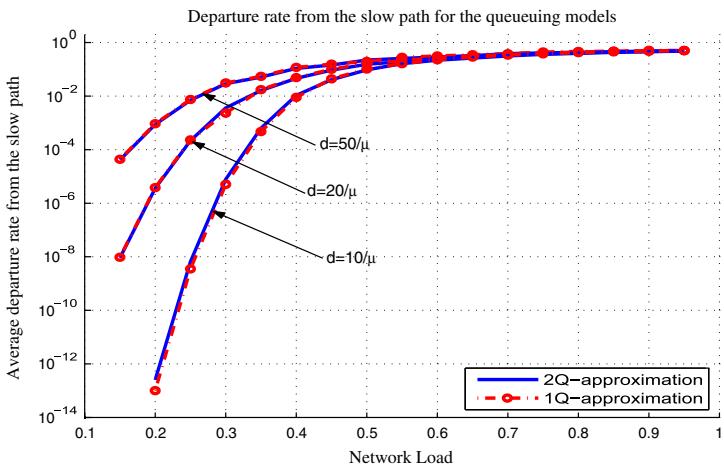

Figure 14. Comparative results: departure rate from the slow path estimation of the single-queue and the two-queue models ($\rho = 50$ and 80%).

same delay joining either queues. As a result, the single-queue system can be modelled as an $M/M/2$ queueing system. For the two-queue system, arrivals will be evenly split between both queues.

DOI: 10.1002/dac

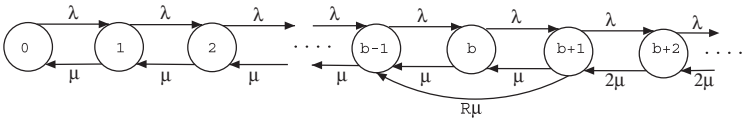

Figure 15. The truncated CTMC to obtain bounds for the number in the system and the mean delay for the single-queue model. When $R=0$, the CTMC provides upper bounds and when $R=1$ it provides lower bounds. The PS threshold is b.

7. APPROXIMATE SOLUTION TO THE SINGLE-QUEUE MODEL

Lin and Kumar [21] solved a similar problem and found complex expressions for the stationary probabilities. Therefore, we are looking for approximations or, preferably, tight bounds with simpler expressions that provide better insight and allow for easy computation. Further, a closed-form expression for the mean number in the system and hence the mean delay seems elusive. In the following we derive tight lower and upper bounds for the stationary probabilities of the CTMC given in Figure 11. For $\rho \equiv \lambda/2\mu < 1$, the Markov chain is ergodic. We refer to this bounding method as the 'Single-queue bounding procedure'.

7.1. Upper bounds on the number in the system

We truncate the state space such that when there are $b+1$ packets in the system, a departure from LINK 2 will not be allowed (i.e. the same packet will be retransmitted over LINK 2) until a departure from LINK 1 occurs. Following the arguments of [32], this provides a sample path based upper bound for the number in the system. Solving the global balance equations of the Markov chain given in Figure 15 when $R=0$ gives an upper bound on the stationary probabilities, $\{\overline{\pi}_i\}$, and consequently an upper bound on the number in the system and the overall mean delay. $\{\overline{\pi}_i\}$ are expressed as follows:

$$\overline{\pi}_i = \begin{cases} \overline{\pi}_0 (2\rho)^i & \text{for } 1 \leqslant i \leqslant b+1 \\ \overline{\pi}_0 2^{b+1} \rho^i & \text{for } i \geqslant b+2 \end{cases}$$

where

$$\overline{\pi}_0 = \frac{(1-\rho)(1-2\rho)}{1-\rho(1+(2\rho)^{b+1})}$$

The upper bound, $\overline{N}$, of the mean queue length is

$$\overline{N} = \overline{\pi}_0 \left[(2\rho)^{b+1} \left(\frac{(b+1)\beta + 1}{\beta^2} + \frac{2((b+1)\alpha - 1)}{\alpha^2} \right) + \frac{2}{\alpha} \right] \rho$$

where $\alpha = 2\rho - 1$ and $\beta = 1 - \rho$.

7.2. Lower bounds on the number in the system

We truncate the state space such that when there are $b+1$ in the queue, a departure from LINK 2 will force a simultaneous departure from LINK 1. The CTMC of the system is given in Figure 15

DOI: 10.1002/dac

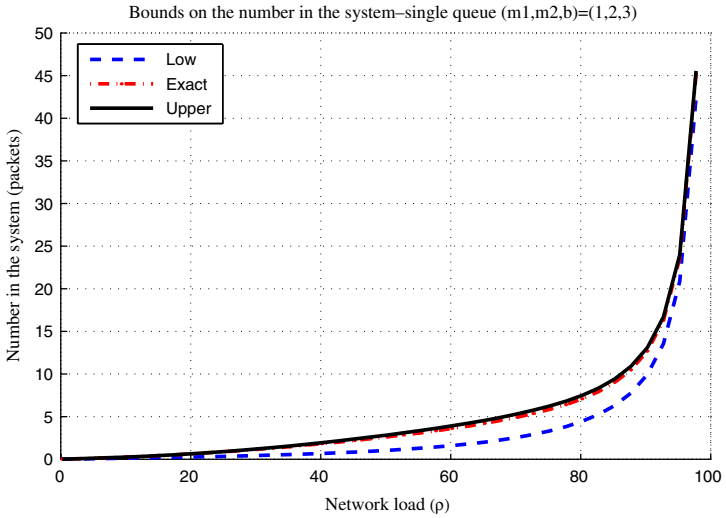

Figure 16. Bounds on the number of packets in the single-queue model. The PS bias $b=3$, $d=50/\mu$.

when $R=1$. The solution of the stationary probabilities of the Markov chain, following the argument in [32], provides lower bounds on the stationary probabilities of the original system and consequently lower bounds on the number in the system and the overall mean delay.

Let $\{\underline{\pi}_i\}$, be the stationary probabilities of the Markov chain (Figure 15 when $R=1$), $\{\underline{\pi}_i\}$ are expressed as

$$\underline{\pi}_i = \begin{cases} \underline{\pi}_0 (2\rho)^i & \text{for } 0 \leqslant i \leqslant b-1 \\ \underline{\pi}_0 \left(\dfrac{2^{b-1}}{1+\rho} \right) \rho^{i-1} & \text{for } i \geqslant b \end{cases}$$

where

$$\underline{\pi}_0 = \left(\frac{1-(2\rho)^b}{1-2\rho} + \frac{(2\rho)^{b-1}}{1-\rho^2} \right)^{-1}$$

The lower bound, $\underline{N}$ of the mean queue length is

$$\underline{N} = \underline{\pi}_0 \left[\frac{2\rho - (2\rho)^{b+1}}{(1-2\rho)^2} - \frac{b(2\rho)^b}{1-2\rho} + \frac{(2\rho)^{b-1}}{1-\rho^2} \left(b + \frac{\rho}{1-\rho} \right) \right]$$

To assess the tightness of the bounds, in Figure 16 we plot $\overline{N}$ and $\underline{N}$ as a function of ρ. The bounds are very tight for low and high network loads.

7.3. Bounds on the mean overall delay

Based on the computed upper and lower bound values of the stationary probabilities, we compute upper and lower bounds on the mean departure rate from LINK 1 ($\overline{f}$, $\underline{f}$). Having computed

DOI: 10.1002/dac

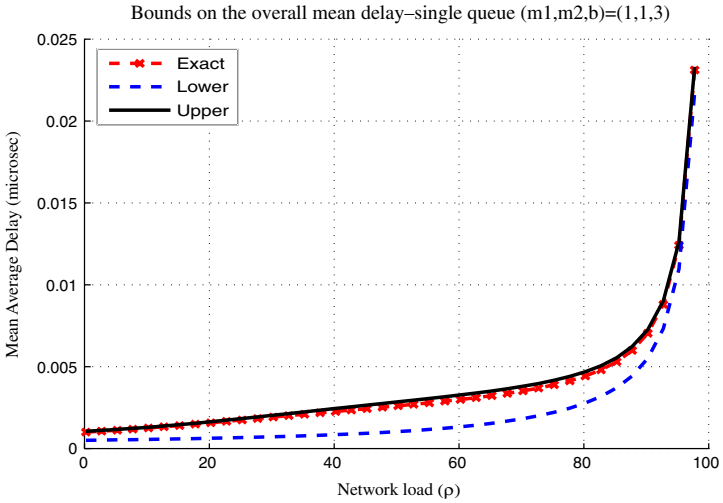

Figure 17. Bounds on the mean delay in the single-queue model. The PS bias $b = 3$, $d = 50/\mu$.

upper and lower bound on the number in the system $(\overline{N}, \underline{N})$, we use Equation (3) and combine similar bounds to obtain bounds on the mean overall delay. Figure 17 shows a plot of the delay bounds. As in the case of the number in the system, the bounds are very tight for high network loads.

8. CHOOSING THE PS BIAS TO MATCH THE DOWNSTREAM DELAY DIFFERENCE

As shown in Figure 5, we can improve the overall network performance by adjusting the routing threshold b according to the downstream delay difference d. If d is large, then we would want to send more packets on the faster path than on the slower one so we set b to a large value. When d is small, however, it does not make much difference to send a large number of packets on the faster path, so we set b to a small value.

Consequently, the optimal value for b must be an increasing function of d. However, our results also show that the 'greedy' value of $b = \mu d$ is too large. Moreover, the optimal value of b is a decreasing function of the load. Therefore, providing a method for choosing a good value for b is an important part of our work.

In general, the forking node can be modelled as a two-dimensional Markov chain, as we discuss in Section 6 (see Figure 8). To estimate the optimal PS threshold, however, we will use the truncated state space we introduced in Section 7. For the purpose of this analysis, we use the truncated state space which Markov chain is given by Figure 15 when $R = 1$.

Now consider, the effect of a change in the PS bias from b to $b + 1$ on the delay T (given by Equation (3)) for a given λ and d. For this analysis, we use the superscript (b) or $(b + 1)$ in our notation to differentiate between the instances where the bias is b and when it is $b + 1$. By increasing the bias, we are sending less packets to queue 1 to avoid its downstream delay.

DOI: 10.1002/dac

On the other hand, we are increasing the delay of packets joining queue 2. Therefore, if the cost of joining queue 1 is larger than the cost of joining queue 2, then it is beneficial to increase the bias to avoid joining queue 1 since the reward from reducing congestion in queue 1 out weights the penalty from increasing congestion in queue 2.

Thus, if we let $\Delta T = T^{(b+1)} - T^{(b)}$, then the value of b for which $\Delta T = 0$ is the 'optimal' b for a given d and λ.

Applying our new notation to Equations (3), (2) and (4) and after simplifications and algebraic manipulation, ΔT can be expressed as

$$\Delta T = \frac{1}{\lambda}(\Delta N + \Delta f d\mu)$$

where

$$\Delta N = N^{(b+1)} - N^{(b)}$$

$$\Delta f = f^{(b)} - f^{(b+1)}$$

As a result, we get the following expression relating d and b:

$$d = \frac{1}{\mu}\frac{\Delta N}{\Delta f} \tag{4}$$

Given the complexity of the expression relating d to b (Equation (4)), it is difficult to show that $b<d$ (assuming $\mu = 1$) through a straightforward analysis. An asymptotic evaluation of the expression shows that $d = O(\lambda^b)$. Figure 18 gives a plot of d *versus* b for various network loads ρ along with a 45° line plot of $b = d$. All the curves are above the 45° line thus, $b<d$.

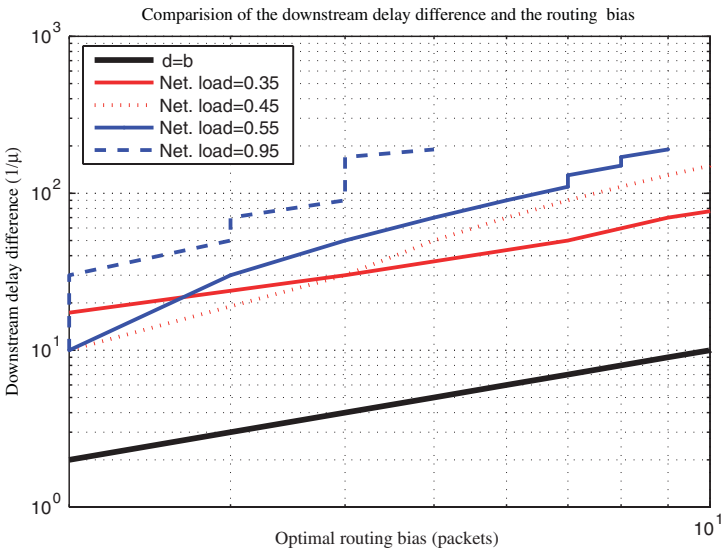

Figure 18. Optimal routing bias for various downstream delays and network loads.

DOI: 10.1002/dac

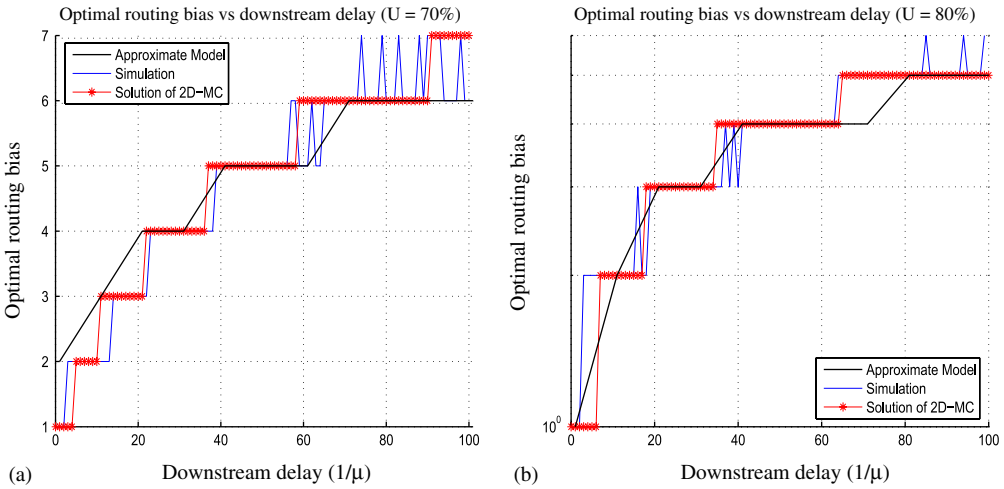

Figure 19. Approximation of the routing bias using simulation, solution to the Markov chain (Figure 8) and the *bd-11 approximation method*: (a) $\rho = 0.70$ and (b) $\rho = 0.80$.

It is interesting to compare this result to the one obtained in [33], where it is shown that $d = b$, i.e. using the terminology in [33], the greedy local optimal policy of join-the-minimum-cost-queue is also the global social optimal under linear costs. The result that $b = d$, however, is true under the assumption of heavy traffic ($\rho \to 1$).

Figure 19 shows how well our approximation of the relationship of b and d fits the results obtained by simulation, which we consider exact and by solving the two-dimensional Markov chain describing the behaviour of the actual system that we describe in Section 6.

9. APPLICATION TO QUEUEING NETWORKS

In this section, we validate our finding that forking nodes can be included in product-form queueing network without compromising the accuracy of the solution method. Therefore, we incorporate our solution method for a JSQ + b forking node in isolation into the product-form solution for two simple open queueing networks that include some JSQ + b forking nodes (shown in Figure 20). However, we assume that the remainder of the network satisfies the conditions for a product-form solution. The simpler example represents a pure feed forward network, where the branching probability at a forking node (which is related to its queue size and hence input rate) does not affect its own input rate. The second example network includes a PS policy with feedback loop containing a forking node. In this example, the networks are non-product-form because of forking nodes A and B, which use output queueing in combination with state-dependent JSQ + b policy. We assume that all the nodes run at the same speed of μ except for C and E which run at a speed of 2μ. We use the 'Single-queue bounding procedure' and the 'Two-queue bounding procedure' described in Sections 7 and 6, respectively, to model the forking nodes. To validate the accuracy of our results, we also simulated the system.

DOI: 10.1002/dac

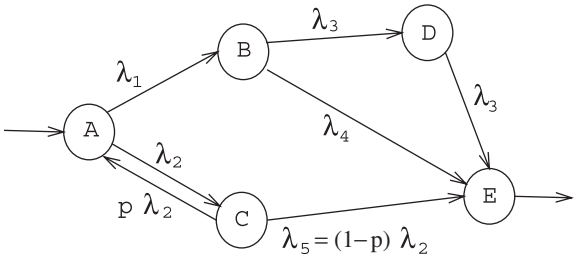

Figure 20. A network with two forking nodes, A and B and several paths, from the source node A, to the destination node, E. When $p>0$, we have a feedback loop network.

9.1. Computational method

Starting with the assumption that the network is product-form, the first step is to find the flow at each link. Normally this step should be easy because we have an open queueing network. However, the PS of packets at each forking node is state-dependent so the output PS probabilities depend on the input packet arrival rate.

For the feed forward network, the flow assignment at each link is done by invoking the 'Single-queue bounding procedure' and the 'Two-queue bounding procedure' at the forking nodes and using standard solution algorithm for open product-form networks. However, for the feedback loop network, the state-dependent PS by JSQ $+ b$ nodes may create problems for the standard solution algorithm. The algorithm first calculates the flows on each link and then computes the associated queue size, whereas the local behaviour of a JSQ $+ b$ node changes the flow assignment to its output links as a function of its queue sizes. Thus, networks in which there is feedback paths through one or more JSQ $+ b$ nodes lead to a cyclic dependency where we need the flow entering the JSQ $+ b$ node before we can calculate its queue sizes, but we need the queue sizes at the JSQ $+ b$ node before we can determine the flow in the feedback path leading to its own input.

We describe the following two approaches to find the flows in each link for both networks.

- For the feed forward network ($p = 0$ in Figure 20): At the forking nodes A and B, we invoke the 'Single-queue bounding procedure' and the 'Two-queue bounding procedure' to estimate the queue length at node A and node B, N_A and N_B, and the departure rates λ_1, λ_2 from A's two output links AB and AC, respectively and λ_3 and λ_4 from B's two output links BD and BE, respectively. In the case of the two-queue forking nodes, N_{AB} and N_{AC} refer to the individual queue lengths at node A. Similarly, N_{BD} and N_{BE} refer to the individual queue lengths at node B. Here, we denote by λ_5 the departure rate from C's output link. We then calculate the mean number in system for each service centre using standard results for product-form queues.
- For the feedback loop network ($p>0$ in Figure 20), we start with an initial guess of the departure rates at each node. We then invoke our 'Single-queue bounding procedure' and 'Two-queue bounding procedure' to estimate the queue length at nodes A and B, and the departure rates from A's two output links and from B's two output links. If these calculated departure rates from the forking nodes do not match our previously estimated flows, then we update the flows on each link and recompute the number in system for each service centre.

DOI: 10.1002/dac

Algorithm 1 illustrates the computation steps in more detail for a general network. The network is composed of L links and K nodes that are a mixture of forking and regular nodes. Let Λ be a flow (departure rate) matrix and R a routing matrix.

Algorithm 1

Product-form solution in a feedback loop network with forking nodes.

```
1:   solve [R^T − I]Λ = 0 for Λ /* using Gaussian elimination */
2:   repeat
3:     R_0 ← R
4:     for i = 1 → L do
5:       Forking ← False
6:       for j = 1 → L do
7:         if i and j are output links of a forking node leading to links say, i' and j'
7:           respectively. →
8:           (N_i, N_j, T_i, T_j, λ_i, λ_j) = Two-queue-bounding-procedure (Λ)
9:           /* An alternative is to invoke the Single-queue-bounding-procedure(Λ)*/
10:          R_ii' ← λ_i/(λ_i+λ_j)
11:          R_jj' ← λ_j/(λ_i+λ_j)
12:          Forking ← True
13:        fi
14:      od
15:      if Forking == False →
16:        ρ_i ⇐ λ_i/μ_i /* μ_i the service rate at the ith queue */
17:        N_i ⇐ ρ_i/(1−ρ_i)
18:        T_i ⇐ N_i/λ_i
19:      fi
20:    od
21:  until ‖ R − R_0 ‖< ε
```

Finally, we compute the overall number in the system N as the sum of the number in the individual nodes, and apply Little's law to obtain the mean delay T.

9.2. Numerical results

Recall that our analysis considered two different approximation strategies for modelling a JSQ$+b$ forking node: (i) *approximating the solution to the exact two-dimensional system* by truncating its state space to leave us with upper and lower bounds that are more easily computed; and (ii) *approximating the system as a much simpler one-dimensional threshold queue model* which has similar behaviour but is much easier to solve (at least to give us upper and lower bounds). For completeness, the results given in Tables III and IV are designed to illustrate the effects of both approximation strategies. In all cases, we use the same parameters for the underlying queueing network model. However, for all results given in Table III we have approximated each forking node as the one-dimensional threshold queue from strategy (ii), whereas in Table IV we have represented each forking node by the 'correct' two-dimensional model.

DOI: 10.1002/dac

Table III. Statistics for the network of Figure 20 when A and B are single-queue forking nodes with JSQ + 2 policy.

Statistics	Simulation	Approximation 1	Approximation 2
(a) *Feed forward network*			
N	9.30	8.876	10.952
N_A	3.42	3.144	4.767
N_B	0.904	0.8247	1.044
N_C	1.240	1.142	0.896
N_D	0.713	0.7656	1.2456
N_E	3.006	3.0	3.0
λ_1	0.3171	0.2891	0.3698
λ_2	0.6828	0.7108	0.6301
λ_3	0.0579	0.0367	0.0250
λ_4	0.2591	0.2525	0.3448
λ_5	0.6828	0.7108	0.6301
$T(1/\mu)$	0.006209	0.00592	0.00730
(b) *Feedback loop network*			
N	10.664	9.9452	12.734
N_A	4.188	4.08	6.105
N_B	1.2316	0.892	1.279
N_C	1.309	1.022	0.8070
N_D	0.9387	0.9512	1.5429
N_E	2.999	3.0	3.0
λ_1	0.351	0.325	0.4045
λ_2	0.649	0.675	0.5955
λ_3	0.0728	0.044	0.0330
λ_4	0.3465	0.281	0.3715
λ_5	0.649	0.674	0.5954
$T(1/\mu)$	0.0071	0.00663	0.00848

Note: Comparison of the simulation of the exact PS policy with the one-dimensional bounds from Section 7: $\rho = 75\%$ and $p = 1/8$.

In Approximations 1 and 2 in Tables III, we use the 'Single-queue bounding procedure' to compute, respectively, lower and upper bounds on the number at the forking nodes A and B. Similarly, in Approximations 3 and 4 in Table IV, we use the 'Two-queue bounding procedure' to compute, respectively, lower and upper bounds on the number at the forking nodes. The discrepancy between the analytical and simulation results *within each table* shows the approximation error that results from assuming the product-form solution algorithm holds in a queueing network where some of the nodes use state-dependent PS (i.e. single-queue with threshold queueing in Table III, and two-queues with JSQ + b policy Table IV). Conversely, the discrepancy *between tables* shows the approximation error from modelling JSQ + b policy as a one-dimensional threshold queueing system.

The results given in Tables III and IV show that the estimated number in the system for the entire network computed *via* both methods are in good agreement, and they also fit very well the 'reference value' obtained by simulation. The number in the forking nodes computed using both methods bound the actual values provided by the simulator. Similar results are obtained for the overall mean delay.

DOI: 10.1002/dac

Table IV. Statistics for the network of Figure 20 when A and B are two-queue forking nodes with JSQ + 2 policy.

Statistics	Simulation	Approximation 3	Approximation 4
(a) *Feed forward network*			
N	9.40	9.05	10.94
N_A	4.49	4.206	6.049
N_{AB}	1.51	1.236	4.20
N_{AC}	2.98	2.97	1.849
N_B	1.0245	0.8995	1.033
N_{BD}	0.0995	0.0435	0.0607
N_{BE}	0.925	0.856	0.9723
N_C	0.780	0.903	0.803
N_D	0.104	0.0417	0.0579
N_E	3.01	3.0	3.0
λ_1	0.3904	0.3671	0.4059
λ_2	0.6096	0.6329	0.5941
λ_3	0.0496	0.0267	0.0365
λ_4	0.3408	0.3404	0.3694
λ_5	0.6096	0.6329	0.5941
$T(1/\mu)$	0.006265	0.006033	0.00729
(b) *Feedback loop network*			
N	11.09	10.393	12.67
N_A	5.85	5.42	7.51
N_{AB}	2.133	1.49	2.63
N_{AC}	3.72	3.92	4.87
N_B	1.217	1.01	1.218
N_{BD}	0.137	0.057	0.0894
N_{BE}	1.08	0.955	1.128
N_C	0.885	0.9389	0.862
N_D	0.144	0.0205	0.0805
N_E	3.01	3.0	3.0
λ_1	0.452	0.3826	0.435
λ_2	0.5479	0.6173	0.5651
λ_3	0.0669	0.033	0.0497
λ_4	0.385	0.349	0.3851
λ_5	0.5479	0.6173	0.5651
$T(1/\mu)$	0.00741	0.0069	0.00838

Note: Comparison of the simulation of the exact PS policy with the matrix geometric bounds from Section 6: $\rho = 75\%$ and $p = 1/8$.

These results support our hypothesis that a queueing network that is non-product-form because it contains forking nodes with JSQ + b policy can be very well approximated by the solution obtained from the product-form solution method.

10. CONCLUSION

In this paper, we have focused our attention on problem of modelling the performance of queueing networks that support locally state-dependent path selection at some of the nodes. We call such

DOI: 10.1002/dac

nodes *forking nodes*, and assume that packets entering such a node can be directed to one of two output links according to a some local PS policy. Moreover, we generalize the problem by allowing the policy to consider the relative delays from the forking node to the final destination along each output path in its scheduling decisions. The main impact of this work is an existence proof: *forking nodes* that use high-performance state-dependent local PS policies can be included in tractable analytical models of network performance by approximating them as service centres in a standard product-form network.

More specifically, we used simulation to compare the performance of different PS policies—including random, deterministic and dynamic (i.e. instantaneous queue-length dependent) distribution of traffic to the two output links—and found that the state-dependent JSQ + b policy consistently gave us the best performance. We then applied standard time-series analysis techniques to the outputs of an isolated forking node to show that the 'offline' behaviour of a JSQ + b node comes very close to satisfying Muntz' sufficient condition for service centres that satisfy the product-form solution.

Unfortunately, although the JSQ + b policy would be easy to implement in a network switch, it is not easy to analyse, even in isolation. Therefore, another important contribution of this paper is to develop and validate some simple approximation methods for modelling the performance of a JSQ + b forking node in isolation, and to show how such a model can be used as service centre within the framework of a standard product-form queueing network model. We also presented some numerical examples to show the effectiveness of our method on various networks.

Finally, we showed that biasing the paths in favour of the faster one can lead to a significant performance improvement. Surprisingly, however, the 'greedy' policy of using the bias to equalize the end-to-end delays across the two alternate paths is *not* optimal. Instead, we found that the optimal bias increases with the difference in delays between the two alternate paths, and decreases with the utilization of the forking node; it is not a simple function, but we have provided an approximate solution.

REFERENCES

1. Elhafsi EH, Molle M, Manjunath D. Can we use product-form solution techniques in networks with asymmetric paths? *Proceedings of the International symposium on Performance Evaluation of Computer and Telecommunication Systems* (SPECTS 2005), 2005; 348–359.
2. Moy J. Rfc 1583, Open shortest path first (OSPF).
3. Kleinrock L. *Queueing Systems*, vols. 1 and 2. Wiley: New York, 1975.
4. Chandy KM, Herzog U, Woo LS. Approximate analysis of general queuing networks. *IBM Journal of Research and Development* 1975; **19**(1):43–49.
5. Jackson JR. Jobshop-like queueing systems. *Management Science* 1963; **10**:131–142.
6. Gordon WJ, Newel GF. Closed queueing networks with exponential servers. *Operations Research* 1967; **15**(2):252–267.
7. Kleinrock L. *Communication Nets: Stochastic Message Flow and Delay*. McGraw-Hill: New York, 1964.
8. Muntz RR. Poisson departure processes and queueing networks. *Technical Report*, IBM Thomas J. Watson Research Center, 1972.
9. Mani Chandy KM, Howard J, Towsley D. Product form and local balance in queueing networks. *Journal of the ACM* 1977; **24**(2):250–263.
10. Baskett F, Mani Chandy K, Muntz R, Palacios F. Open, closed, and mixed networks of queues with different classes of customers. *Journal of the ACM* 1975; **22**(2):248–260.
11. Walrand J. *An Introduction to Queueing Networks*. Prentice-Hall: Englewood Cliffs, NJ, 1988.
12. Noetzel A. A generalized queueing discipline for product form network solution. *Journal of the ACM* 1979; **26**(4):779–793.

DOI: 10.1002/dac

13. Towsley D. Queueing network models with state dependent routing. *Journal of the ACM* 1980; **27**(2):323–337.
14. Nelson RD. The mathematics of product form queueing networks. *ACM Computing Surveys* 1993; **25**(3): 339–369.
15. Bolch G, Greiner S, Meer H, Trivedi K. *Queueing Networks and Markov Chains—Modeling and Performance Evaluation with Computer Science Applications*. Wiley: New York, 1998.
16. Visschers J, Adan I, Wessels J. Product form solutions to production systems with simultaneous resource possession. *Technical Report 99-15*, 1999.
17. Adan I, Wessels J, Zijm W. Analysis of the symmetric shortest queue problem. *Stochastic Models* 1990; **6**:691–713.
18. Adan I, Wessels J, Zijm W. Analysis of the asymmetric shortest queue problem. *Queuing Systems* 1991; **8**: 1–58.
19. Turner SRE. A join the shorter queue model in heavy traffic. *Journal of Applied Probability* 2000; **37**(1).
20. Koole G. A simple proof of the optimality of a threshold policy in a two-server queueing system. *Systems and Control Letters* 1995; **26**(5):301–303.
21. Lin W, Kumar PR. Optimal control of a queueing system with two heterogeneous servers. *IEEE Transactions on Automatic Control* 1984; **29**.
22. Viniotis I, Ephremides A. Extension of the optimality of the threshold policy in heterogeneous multiserver queueing systems. *IEEE Transactions on Automatic Control* 1988; **33**:104–109.
23. Karagiannis T, Molle M, Faloutsos M, Broido A. A nonstationary poisson view of internet traffic. *Proceedings of IEEE INFOCOM*, *23rd Annual Joint Conference of the IEEE Computer and Communication Societies*, 2004; **3**:1558–1569.
24. Kumar A, Manjunath D, Kuri J. *Communication Networking: An Analytical Approach*. Morgan Kaufman: San Francisco, CA, 2004.
25. Walrand J. *Queueing Networks*. Prentice-Hall: Englewood Cliffs, NJ, 1988.
26. Winston W. Optimality of the shortest line discipline. *Journal of Applied Probability* 1977; **14**:181–189.
27. Elhafsi EH, Molle M. Optimal routing between alternate paths with different network transit delays. *Proceedings of IEEE Global Telecommunications Conference*, GLOBECOM, San Fransisco, CA, November 2006; CAM04-3.
28. Mesquite Software Inc. *Csim 19 Documentation: User Guide*. http://www.mesquite.com/documentation/index.htm, 2004.
29. Gertsbakh I. The shorter queue problem: a numerical study using the matrix geometric solution. *European Journal of Operational Research* 1984; **15**:374–381.
30. Elhafsi EH, Molle M. On a genaralization of the join-the-shortest-queue scheduling with bias. *Technical Report UCR-CS-2005-09001*, September 2005.
31. van Houtoum GJ, Adan IJBF, Wessesl J, Zijm WHM. Performance analysis of parallel identical machines with a generalized shortest queue arrival mechanism. *OR Specktrum* 2000; **23**:411–428.
32. Tandra R, Hemachandra N, Manjunath D. Join minimum cost queue for multiclass customers: stability and performance bounds. *Probability in the Engineering and Informational Sciences*, 2002.
33. Dube P, Borkar VS, Manjunath D. Differential join prices for parallel queues: social optimality, dynamic pricing algorithms and application to internet pricing. *Proceedings of IEEE INFOCOM*, 2002.

AUTHORS' BIOGRAPHIES

Essia H. ElHafsi received the BSc and the MS degree in Industrial and Systems Engineering from the Ohio State University and the University of Florida respectively. She received a PhD in Computer Science from the University of California Riverside. She is currently a Postdoctoral Researcher at the Fundamental Computer Science Laboratory at the Univ. Lille 1, CNRS/INRIA, France. Her research interests include energy efficient based geographic routing in sensor and *ad hoc* networks and performance evaluation of computer networks.

DOI: 10.1002/dac

Mart Molle received the BSc (Hons) degree in Mathematics/Computing Science from Queen's University at Kingston, Canada, and the MS and PhD degrees in Computer Science from the University of California, Los Angeles. He is now a Professor in the department of Computer Science and Engineering at the University of California, Riverside. His research interests include the performance evaluation of protocols for computer networks and of distributed systems.

D. Manjunath received his BE from Mysore University, MS from Indian Institute of Technology, Madras and PhD from Rensselaer Polytechnic Institute, Troy NY in 1986, 1989 and 1993 respectively. He has worked in the Corporate R&D Center of General Electric in Scehenectady NY (1990), Computer and Information Sciences department of the University of Delaware (1992–1993) and the Computer Science department, University of Toronto (1993–1994). He was with the department of Electrical Engineering of Indian Institute of Technology, Kanpur during 1994–1998. He has been with the Electrical Engineering department of IIT-Bombay since July 1998 where he is now an Associate Professor. His research interests are in the general areas of communication networks and performance analysis. His recent research has concentrated on network traffic and performance measurement, analysis of random wireless data and sensor networks, network pricing and queue control. He is a coauthor of 'Communication Networking: An Analytical Approach' published by Morgan-Kaufman in May 2004.

DOI: 10.1002/dac