IIE Transactions (1997) **29**, 213–219

Simulation studies of multiclass queueing networks

J. BANKS and J. G. DAI
School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332–0205, USA

Received July 1995 and accepted September 1996

Multiclass queueing networks have been used to model communication and complex manufacturing systems. Recent results have shown that these networks can be unstable in the sense that the total number of jobs in the network explodes as time tends to infinity even if the traffic intensity at each station is less than unity. Simulation is used to demonstrate these unusual phenomena. Conjectures based on simulation are presented.

1. Introduction

Multiclass queueing networks have been used to model communication and complex manufacturing systems. A distinguishing feature of these networks is that there may be more than one job class served at a station. When there are multiple job classes waiting at a station, the server has to choose a job class each time that a service is completed. A dispatching rule is used to make such a choice. Figure 1 is an example of what Kumar [1] calls a *reentrant line*, a special type of multiclass open queueing network distinguished by a single external arrival process and deterministic routing. We define a different *job class* for each stop on the route and we allow each class to have its own service time distribution. There are three job classes served at station 1 and two job classes at station 2. For a general description of multiclass queueing networks, readers are referred to Harrison and Nguyen [2].

It has been widely believed that if the nominal load per unit of time ρ_i (also called traffic intensity) at each station i is less than one, the network ought to have enough capacity to handle the load. For the example in Fig. 1, $\rho_1 = m_1 + m_3 + m_5$ and $\rho_2 = m_2 + m_4$, assuming that the mean arrival rate $\lambda = 1$, where m_k is the mean service time for class k. For future reference,

$$\rho_i < 1 \quad \text{for each station } i$$

is called the *usual traffic condition*. However, recent examples have shown that there are non-idling dispatching rules under which the total number of jobs in the network explodes even if the usual traffic condition is satisfied. (A dispatching rule is non-idling if each server works whenever there is a job at the station.) Some of these unstable dispatching rules are mentioned in the literature and are often used in practice. Lu and Kumar [3] and Rybko and Stolyar [4] show that some buffer priority rules can be unstable. (Envisage the situation in which each job class has a separate buffer to queue its jobs. Under a buffer priority rule, jobs are assigned priorities based on the their class designations. Jobs within the same class are served according to a first-in-first-out, or FIFO, rule.) Bramson [5] and Seidman [6] showed that a FIFO rule can be unstable. For an updated account of this fast-growing area, see Dai and VandeVate [7], in which sharp stability regions have been determined for two-station multiclass queueing networks.

In this article we use simulation to demonstrate, but not to prove, that many commonly mentioned dispatching rules, including FIFO, shortest-mean-processing-time-first, shortest-mean-remaining-time-first and buffer priority rules, can be unstable. We use simulation to demonstrate these surprising phenomena because its results are statistically convincing. By using simulation we can also relax some of the assumptions on networks that were imposed in [7] to achieve some clean theoretical derivations.

All the simulations in this article, except that in Section 6, are performed by using the program `rline` developed by the second author to assist in the research conducted in [7]. The program was instrumental in redirecting some of the early stages of a research effort, which eventually led to a complete, explicit characterization of the stability region for two-station multiclass networks. (See Appendix A for an example of an input file and command line for `rline`.) The program offers a wide variety of dispatching rules that can be easily specified, and it runs quickly. The program runs on virtually any computer operating under DOS or UNIX. Readers who wish to obtain this free software can contact the second author.

The next section discusses a two-station five-class network. In Section 3 we demonstrate that a network can be unstable under a FIFO dispatching rule. In Section 4 we discuss a three-station network with common services times at each station that may be unstable under certain priority service disciplines. In Sections 5 and 6 we demonstrate that two commonly referenced dispatching rules can be unstable. Some concluding remarks are given in Section 7.

2. The five-class network

In this section we study the two-station five-class network pictured in Fig. 1. This network was first presented in [7]. Recall that the mean arrival rate is assumed to be $\lambda = 1$. We fix the mean service times to be $m_1 = m_3 = m_4 = 0.1$ and $m_2 = m_5 = 0.6$. The traffic intensities are given by

$$\rho_1 = 0.8 \quad \text{and} \quad \rho_2 = 0.7.$$

Therefore the usual traffic condition is satisfied for the parameter set. Consider the following buffer priority dispatching rule: last-buffer-first-served (LBFS) at station 1 and first-buffer-first-served (FBFS) at station 2. (In our network, LBFS at station 1 is equivalent to the priority list $\{5,3,1\}$, where class 5 has the highest priority and class 1 has the lowest priority. With the list notation, the buffer priority rule is denoted as $(\{5,3,1\},\{2,4\})$.) We use `rline` to simulate the network starting empty. Although the stability region seems not to depend on distributions, we used independent exponential distributions for interarrival and service times in our simulation. As shown in Table 1, the average time in system grows as the number of jobs departed increases. This is consistent with the result in [7] that, with probability one, the average time in system explodes as the number of jobs departed tends to infinity. Notice that the fraction of time that server 1 is busy (utilization at station 1) is less than 0.65, which is 0.15 less than the traffic intensity at station 1. This 15% forced idle time for server 1, caused by mutual blocking between station 1 and station 2, effectively reduces the capacity of station 1. Figure 2 shows that in many time intervals, one server has no job to process whereas the other station has numerous jobs waiting.

It is proved in [7] that as long as class 2 has higher priority than class 4, and class 5 has higher priority than class 1, and high priority jobs preempt low priority jobs, then the total number of jobs in the network explodes as time $t \to \infty$ when

$$m_2 + m_5 > 1.$$

That is, when the buffer priority dispatching rules are as follows: $(\{5,3,1\},\{2,4\})$, $(\{5,1,3\},\{2,4\})$ and $(\{3,5,1\},\{2,4\})$, then the network is unstable (because $m_2 + m_5 = 1.2 > 1$ in our example). The preemption assumption in [7] is needed to argue that when the network starts empty, class 2 jobs and class 5 jobs can never be served simultaneously under the specified buffer priority rule. In our simulation, no preemption assumption is used. Yet our simulation still shows that the network is

Table 1. A simulation study of a two-station five-class reentrant line that is unstable under a buffer priority dispatching rule

Number of jobs departed	*100*	*1000*	*10 000*	*100 000*
Average time in system	15.8	183.7	1740.2	17 043.7
Utilization at station 1	0.65	0.60	0.61	0.65
Utilization at station 2	0.59	0.68	0.67	0.61

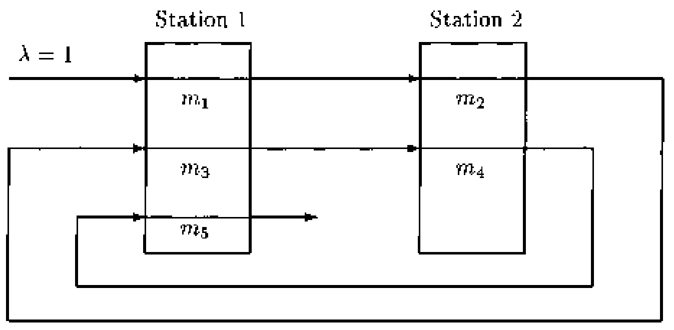

Fig. 1. An example of reentrant lines.

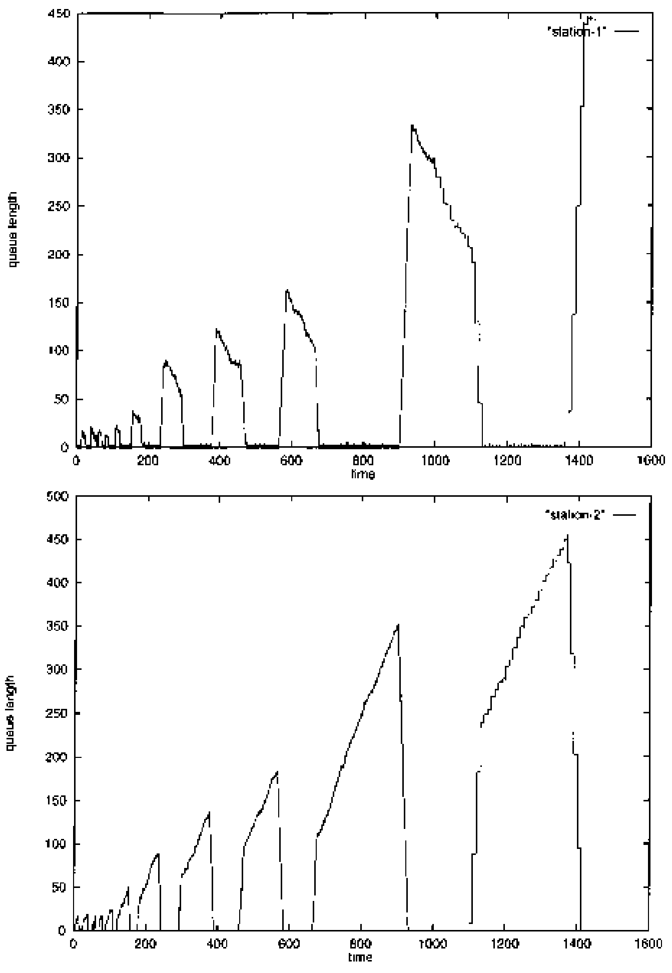

Fig. 2. Job size plots at stations 1 and 2.

unstable. Furthermore, our simulation indicates that classes 2 and 5 can never be served together even if no preemption is allowed. We conjecture that when the network starts empty, as long as class 2 has higher priority than class 4 and class 5 has higher priority than class 1, classes 2 and 5 can never be served simultaneously regardless of whether preemption is used or not.

It is proved in [7] that if

$$m_2 + m_5 < 1,$$

in addition to the usual traffic condition, the network is stable under any non-idling dispatching rule. Therefore, for the buffer priority rules in cases 8, 10 and 11 of Table 2, the stability region is given by

$$m_1 + m_3 + m_5 < 1, \quad m_2 + m_4 < 1, \quad m_2 + m_5 < 1.$$

It is also proved in [7] that if class 1 receives the highest priority and the usual traffic condition is satisfied, then the network is stable. Dai and Weiss [8] proved that if all stations use the LBFS buffer priority rule, any reentrant line is stable under the usual traffic condition.

When the usual traffic condition holds and class 5 receives the lowest priority, the subnetwork containing the first four classes is stable, as proved in [7]. Hence the five-class network is stable as a generalization of an argument in Proposition 7.2 of Dai *et al.* [9]. Therefore the stability region for cases 5 and 6 is as given in Table 2.

This leaves the stability region undetermined for the following two buffer priority rules: $(\{3,5,1\}, \{4,2\})$ and $(\{5,1,3\}, \{4,2\})$. We conducted an extensive simulation study for these dispatching rules. On the basis of the simulation, we conjectured that under any one of these rules the network is stable if the usual traffic condition is satisfied. (During the review process, Chen and Zhang [10] indeed proved these conjectures.) A summary of stability regions for the five-class network under all buffer priority rules is given in Table 2, where $\{u,v\}$ represents any permutation of $\{1,5\}$ and $\{x,y\}$ represents any permutation of $\{2,4\}$.

Table 2. Summary of stability regions under all buffer priority rules

	Rule			
Case	*Station 1*	*Station 2*	*Stability region*	*Source of reference*
1–4	$\{1,u,v\}$	$\{x,y\}$	$\rho_1 < 1,\ \rho_2 < 1$	[7]
5–6	$\{3,1,5\}$	$\{x,y\}$	$\rho_1 < 1,\ \rho_2 < 1$	This paper
7	$\{3,5,1\}$	$\{4,2\}$	$\rho_1 < 1,\ \rho_2 < 1$	[10]
8	$\{3,5,1\}$	$\{2,4\}$	$\rho_1 < 1, \rho_2 < 1,\ m_2 + m_5 < 1$	[7]
9	$\{5,1,3\}$	$\{4,2\}$	$\rho_1 < 1,\ \rho_2 < 1$	[10]
10	$\{5,3,1\}$	$\{2,4\}$	$\rho_1 < 1,\ \rho_2 < 1,\ m_2 + m_5 < 1$	[7]
11	$\{5,1,3\}$	$\{2,4\}$	$\rho_1 < 1,\ \rho_2 < 1,\ m_2 + m_5 < 1$	[7]
12	$\{5,3,1\}$	$\{4,2\}$	$\rho_1 < 1, \rho_2 < 1$	[8]

3. First-in-first-out dispatching rule

In this section we demonstrate that a network whose stations satisfy the usual traffic condition can be unstable under the FIFO dispatching rule. This surprising phenomenon is presented by Bramson [5] for a stochastic network and by Seidman [6] for a deterministic network. Consider the reentrant line in Fig. 3. Notice that each job makes three consecutive stops at station 2. The mean service times are given by

$$m_1 = 0.01, \quad m_2 = 0.88,$$
$$m_3 = 0.01, \quad m_4 = 0.01, \quad m_5 = 0.89.$$

Therefore the last visit to station 1 and the first visit to station 2 have long service times, whereas all other service times are short. The traffic intensities at stations 1 and 2 are 0.90. The performance measures in Table 3 are obtained from one replication because the interest is in the growth in the average time in system as the number of jobs departed increases. The utilization rate at each station is significantly less than the traffic intensity. As shown in Table 3, the average time in system grows as the number of jobs departed increases. We conjecture that the average time in system explodes as the number of jobs departed tends to infinity. In Bramson's original work, to carry out the necessary mathematical analysis, thousands of short visits to station 2 are allowed after the first long service at station 2. Kelly [11] showed that under the FIFO rule, if the mean service times within a station are the same, although not necessarily the same between all stations, then the network has a product-form stationary distribution, and therefore is stable. This suggests that in order for the instability phenomenon to occur under the FIFO rule, it is essential that the mean service times for different classes at a station be of a different order of magnitude.

The network in Fig. 3 contains immediate feedback at station 2. Immediate feedback may not be realistic in practice. We slightly alter the two-station network into the three-station network in Fig. 4. In this three-station network there is no immediate feedback. We can still demonstrate Bramson's instability result. Notice that if

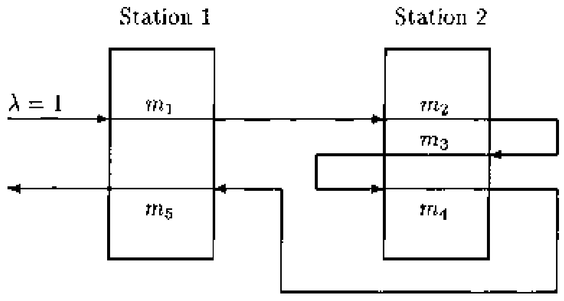

Fig. 3. A Bramson type network that can be unstable under FIFO.

Table 3. A simulation study of the Bramson-type network

Number of jobs departed	*100*	*1000*	*10 000*	*100 000*
Average time in system	23.7	150.8	1155	11 210
Utilization at station 1	0.56	0.69	0.76	0.76
Utilization at station 2	0.68	0.85	0.81	0.86

we set the service times at station 3 to be zero, the network is reduced to the two-station network shown in Fig. 3.

Now we assume that there is only one repeat stop at station 2 in Fig. 3. That is, the routing of each job is to stations 1, 2, 2, 1 and then exit. This is, in fact, the network considered in Lu and Kumar [3]. They proved that the network can be unstable if class 2 has higher priority than class 3 and class 4 has higher priority than class 1. However, it is an open question whether the network is stable under the FIFO rule. On the basis of our extensive simulations, we conjecture that the network is stable under FIFO as long as the usual traffic condition is satisfied.

4. Kelly-type networks

For our purposes we call a network *Kelly-type* if jobs at a station have a common service time distribution. In other words, in a Kelly-type network the service time distributions are not class dependent. When all distributions are exponential, Kelly [11] proved that the network has a product form stationary distribution if the dispatching rule is FIFO and the usual traffic conditions are satisfied. From Sections 2 and 3, readers might conclude that the instability phenomenon was mainly attributed to the unevenness of mean service times at a station. In this section, we show that Kelly-type networks can be unstable under some non-idling dispatching rules. For the network shown in Fig. 5, jobs in classes 1, 7 and 9 visit station 1; jobs in classes 2, 4 and 6 visit station 2; and jobs in classes 3, 5 and 8 visit station 3. Assume that the mean arrival rate to class 1 is 1, and the mean service time for each visit to stations 1 and 2 is 0.3 and for each visit to station 3 is 0.1. Therefore

$$\rho_1 = 0.90 < 1, \quad \rho_2 = 0.90 < 1 \quad \text{and} \quad \rho_3 = 0.30 < 1.$$

Assuming that the dispatching rules are LBFS at station 1 and FBFS at stations 2 and 3, we simulated this network under two distributional assumptions. In the first case (case (M)), all distributions are assumed to be exponential. In the second case (case (D)), all interarrival and service times are deterministic. For (M), the network is initially empty. For the deterministic case (D), two subcases, (D1) and (D2), are considered. For (D1) there are two jobs initially in front of buffer 1. For (D2) the network is initially empty. It seems from Table 4 that the average times in system in the simulations related to cases (M) and (D1) are growing without bound, whereas in the simulation related to case (D2) the total customer population seems bounded. The reason for the vast difference in the performance between measures in cases (D1) and (D2) is yet to be explained. Obviously, the network is extremely sensitive to the initial starting conditions. The network shown in Fig. 5 was simulated in Dai and Meyn [12] under a slightly different dispatching rule.

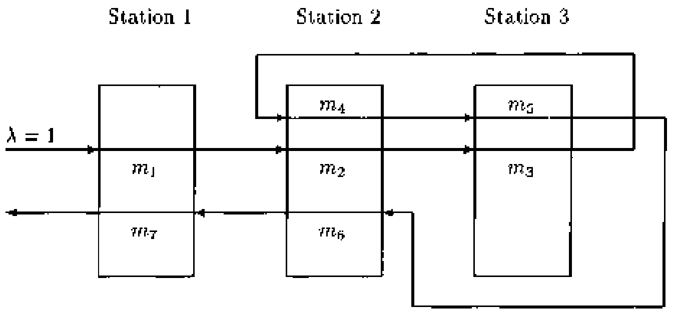

Fig. 4. A three-station network without immediate feedback.

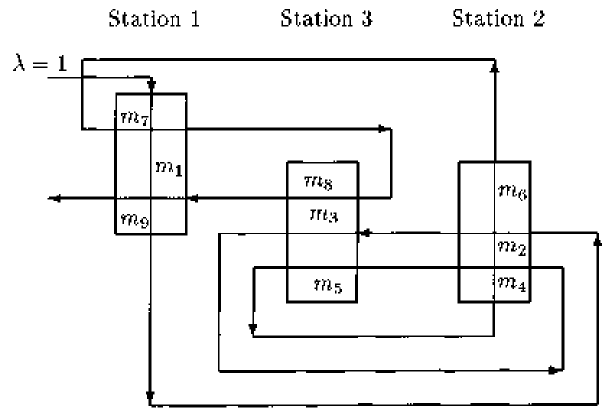

Fig. 5. A three-station Kelly-type network that can be unstable under certain priority service disciplines.

Table 4. Simulation study of the Kelly-type network

Case	*Number of jobs departed*	*Utilization rate at each station*			*Average time in system*
		1	*2*	*3*	
(M)	1 000	0.70	0.80	0.27	181.5
	10 000	0.73	0.81	0.26	1 771.9
	100 000	0.71	0.81	0.25	17 199.6
(D1)	1 000	0.70	0.80	0.27	161.2
	10 000	0.72	0.81	0.26	1 698.5
	100 000	0.72	0.81	0.26	17 486
(D2)	1 000	0.90	0.90	0.30	2.85
	10 000	0.90	0.90	0.30	2.85
	100 000	0.90	0.90	0.30	2.85

They found the same phenomenon as that displayed in Table 4. Readers are referred to Dai and Weiss ([8], Remark 3 in Section 6) for insight into the instability revealed in these simulations; see also Chen ([13], Theorem 4.5) and Gu [14].

5. Shortest-mean-processing-time-first

In this section we demonstrate that the shortest-mean-processing-time-first dispatching rule can be unstable. Let us perturb the mean service times for the network considered in Section 4. We take

$$m_1 = 0.35,\ m_2 = 0.25,\ m_3 = 0.10,\ m_4 = 0.30,\ m_5 = 0.20,$$
$$m_6 = 0.35,\ m_7 = 0.30,\ m_8 = 0.30,\ m_9 = 0.25.$$

Under the same buffer priority rule used in Section 4, we simulated the network by using exponential distributions. Because the mean service times at station 1 are $m_1 = 0.35$, $m_7 = 0.3$ and $m_9 = 0.25$, the LBFS rule at station 1 is equivalent to the shortest-mean-processing-time-first dispatching rule. At station 2 the mean processing times are $m_2 = 0.25$, $m_4 = 0.30$ and $m_6 = 0.35$. The FBFS rule at station 2 is equivalent to the shortest-mean-processing-time-first dispatching rule. Similarly, at station 3 the mean processing times are $m_3 = 0.1$, $m_5 = 0.2$, $m_8 = 0.3$, and the FBFS rule is equivalent to the shortest-mean-processing-time-first dispatching rule. Table 5 shows that the average time in system is growing as the number of jobs departed increases. Hence it is conjectured that the shortest-mean-processing-time-first dispatching rule is unstable.

6. Shortest-mean-remaining-processing-time-first

In this section we demonstrate that the shortest-mean-remaining-processing-time-first dispatching rule can be unstable. The network in Fig. 6 was first considered by Kumar and Seidman [15] and later by Rybko and Stolyar [4]. Rybko and Stolyar showed that if classes 2 and 4 have high priorities, the network can be unstable. Botvich and Zamyatin [16] further proved that if

$$m_2 + m_4 < 1,$$

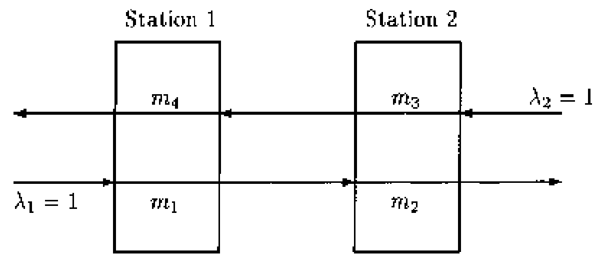

Fig. 6. A network with two types of job.

in addition to the usual traffic conditions, the network is stable under any non-idling dispatching rules. If

$$m_2 + m_4 > 1,$$

the network is unstable under the priority rule considered in Rybko and Stolyar [4].

Consider the choice $m_2 = m_4 = 0.6$ and $m_1 = m_3 = 0.1$. For class 1 the remaining mean processing time is $m_1 + m_2 = 0.7$. For class 2, the remaining mean processing time is $m_2 = 0.6$. Similarly, for class 3, the remaining mean processing time is 0.7, and for class 4 the remaining mean processing time is 0.6. Therefore the shortest-remaining-mean-processing-time-first rule is the same as the Rybko–Stolyar buffer priority rule. Since

$$m_2 + m_4 = 1.2 > 1,$$

the network is unstable under the shortest-remaining-mean-processing-time-first rule.

Table 6 shows the simulation results for this network. Because the network is not a reentrant line, `rline` cannot be used for simulation. Instead, the simulation is performed with a commercial package. In all runs, distributions are assumed to be exponential and the network starts empty. It can be seen that the average queue lengths are growing explosively, and further that the utilization rates at both stations are less than their nominal values of 0.70.

7. Concluding remarks

Researchers have been rather surprised to learn that there are open queueing networks that are unstable even though their traffic intensity at each station is less than unity. This article described some of the networks that

Table 5. Simulation study of the nine-class reentrant line that is unstable under the shortest-mean-processing-time-first dispatching rule

Number of jobs departed	*100*	*1000*	*10 000*	*100 000*
Average time in system	23.2	110.4	882.7	8183
Utilization at station 1	0.80	0.79	0.80	0.81
Utilization at station 2	0.77	0.80	0.84	0.84
Utilization at station 3	0.58	0.54	0.54	0.55

Table 6. Simulation study of the Rybko–Stolyar network that is unstable under the shortest-mean-remaining-processing-time-first dispatching rule

Simulation ending time	*100*	*1000*	*10 000*	*50 000*
Average queue size at station 1	2.6	80	966	4905
Average queue size at station 2	1.9	50	628	3346
Utilization at station 1	0.61	0.62	0.65	0.65
Utilization at station 2	0.51	0.54	0.54	0.54

have exhibited this surprising behavior and further illustrated how simulation has been used to understand the instability or to conjecture about both stability and instability. In particular, we demonstrated that some buffer priority dispatching rules, as well as FIFO, shortest-mean-processing-time-first, and shortest-remaining-processing-time-first dispatching rules, can be unstable. Simulation has also been used in the calculation of the feasible region for stability. Researchers are continuing their investigations in this area, which has both theoretical and applied interests.

Acknowledgements

We thank Dave Goldsman, one of the editors for this special issue, and an associate editor for helping improve the clarity of this article. This research was supported in part by NSF Grant DMI-9457336 and matching funds from CACI Products Co. and AutoSimulations, Inc.

References

[1] Kumar, P. R. (1993) Re-entrant lines. *Queueing Systems: Theory and Applications*, **13**, 87–110.
[2] Harrison, J. M. and Nguyen, V. (1993) Brownian models of multiclass queueing networks: current status and open problems. *Queueing Systems: Theory and Applications*, **13**, 5–40.
[3] Lu, S. H. and Kumar, P. R. (1991) Distributed scheduling based on due dates and buffer priorities. *IEEE Transactions on Automatic Control*, **36**, 1406–1416.
[4] Rybko, A. N. and Stolyar, A. L. (1992) Ergodicity of stochastic processes describing the operation of open queueing networks. *Problems of Information Transmission*, **28**, 199–220.
[5] Bramson, M. (1994) Instability of FIFO queueing networks. *Annals of Applied Probability*, **4**, 414–431.
[6] Seidman, T. I. (1994) 'First come, first served' can be unstable! *IEEE Transactions on Automatic Control*, **39**, 2166–2171.
[7] Dai, J. G. and VandeVate, J. (1996) Virtual stations and the capacity of two-station queueing networks. Preprint.
[8] Dai, J. G. and Weiss, G. (1996) Stability and instability of fluid models for re-entrant lines. *Mathematics of Operations Research*, **21**, 115–134.
[9] Dai, J. G., Hasenbein, J. and Vande Vate, J. H. (1996) On the monotone global stability regions of multiclass queueing networks. Preprint.
[10] Chen, H. and Zhang, H. (1996) Stability of multiclass queueing networks under priority service disciplines. Preprint.
[11] Kelly, F. P. (1975) Networks of queues with customers of different types. *Journal of Applied Probability*, **12**, 542–554.
[12] Dai, J. G. and Meyn, S. P. (1995) Stability and convergence of moments for multiclass queueing networks via fluid limit models. *IEEE Transactions on Automatic Control*, **40**, 1889–1904.
[13] Chen, H. (1995) Fluid approximations and stability of multiclass queueing networks. I. Work-conserving disciplines. *Annals of Applied Probability*, **5**, 637–665.
[14] Gu, J. M. (1995) Convergence and performance for some Kelly-like queueing networks. Ph.D. thesis, University of Wisconsin, Madison.
[15] Kumar, P. R. and Seidman, T. I. (1990) Dynamic instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems. *IEEE Transactions on Automatic Control*, **35**, 289–298.
[16] Botvich, D. D. and Zamyatin, A. A. (1992) Ergodicity of conservative communication networks. Rapport de recherche 1772, INRIA.

Biographies

Jerry Banks is a Professor in the School of Industrial and Systems Engineering at the Georgia Institute of Technology, Atlanta, USA. He received the Ph.D. degree from the Oklahoma State University. He teaches discrete-event simulation, quality control, and creativity/innovation. He is the author, co-author, or co-editor of ten books, several chapters in texts, and numerous technical papers. He is the co-author, with John Carson and Barry Nelson, of *Discrete-Event Systems Simulation*, Second Edition, published by Prentice-Hall, Upper Saddle River, NJ, (1996). He is the co-author of *Introduction to SIMAN V and CINEMA V*, published by John Wiley, New York, in 1995. He is also the co-author of *Getting Started with GPSS/H*, Second Edition, published by Wolverine Software Corporation, Annandale, VA, in 1995. Other recent titles include the co-authored text *Forecasting and Management of Technology* published in 1991 and the single-authored text *Principles of Quality Control* published in 1989, both by John Wiley, New York. He is the editor of the *Handbook of Simulation*, scheduled to be published in 1997 by John Wiley. His current area of research is simulation modeling for surge production. He is the Associate Director for Research of the Center for International Standards and Quality at the Georgia Institute of Technology. He was a founding partner in the simulation consulting firm Carson/Banks and Associates, Inc., located in Atlanta, GA. The firm was purchased by AutoSimulations, Inc., Bountiful, UT, in May 1994. He is a full member of many technical societies including the Institute of Industrial Engineers for which he served eight years as that organization's representative to the Board of the Winter Simulation Conference, including two years as Board Chair.

Jim Dia is an Associate Professor holding a joint appointment in the School of Industrial and Systems Engineering and the School of Mathematics at Georgia Institute of Technology, Atlanta, GA. He received the B.A. and M.S. degrees from Nanjing University, China, in 1982 and 1985, and the Ph.D. degree from Stanford University in 1990. His research interests include performance analysis and scheduling of multiclass queueing networks arising from telecommunication and manufacturing systems. He is a member of the Institute for Operations Research and the Management Sciences and the Institute of Mathematical Statistics. He currently serves as an associate editor for *Operations Research, Mathematics of Operations Research* and *Queueing Systems*. He was the recipient of an NSF Young Investigator Award in 1994 from the National Science Foundation, a Bergmann Memorial Grant in 1995 from the US Israel Binational Foundation, and the Best Paper Award in 1996 from the Georgia Tech Chapter of Sigma Xi.

Appendix A

In this appendix we present an input data file for `rline`. `rline` was used for most of the simulations in this article. The data file is used to produce all of the results in Table 1. Notice that the run length is not specified in the data file. It is controlled with the options provided by `rline`.

```
# "5class.rln": input data file for the net-
work in Fig. 1.
```

```
# The pound sign # is a comment character.
#
5       # number of classes
1       # mean interarrival times
1       # squared coefficient of
        # variation (scv) for
        # interarrival times
1       # class 1 visits station 1
0.1     # mean service time for class 1
1       # scv for service time
2       # class 2 visits station 2
0.6
1
1       # class 3 visits station 1
0.1
1
2       # class 4 visits station 2
0.1
1
1       # class 5 visits station 1
0.6
1
Prio    # station 1 uses a priority
        # dispatching rule
5 3 1   # class 5 has the highest priority
        # and class 1 has the lowest priority
prio    # station 2 uses a priority
        # dispatching rule
2 4     # class 2 has the highest
        # priority and class 4 has
        # the lowest priority
```

There are many runtime options built into {\tt rline}. For example, the command line

```
rline .q .r 10 .t 1000 5class.rln
```

will accomplish the following:

`rline`	causes the execution of the program;
`-q`	generates average queue length and utilization at each station;
`-r` 10	runs ten replications;
`-t` 1000	discards the first 1000 jobs departed before doing statistical calculations;
`5class.rln`	specifies the data filename.