# On Pooling in Queueing Networks

*Avishai Mandelbaum*
Faculty of Industrial Engineering and Management
Technion
Haifa, Israel

*Martin I. Reiman*
Bell Labs, Lucent Technologies
Murray Hill, New Jersey 07974

February 18, 1996
Revised: October 24, 1996; May 12, 1997; May 4, 1998

## Abstract

We view each station in a Jackson network as a queue of tasks, of a particular type, which are to be processed by the associated *specialized* server. A complete pooling of queues, into a single queue, and servers, into a single server, gives rise to an M/PH/1 queue, where the server is *flexible* in the sense that it processes all tasks. We assess the value of complete pooling by comparing the steady-state mean sojourn times of these two systems. The main insight from our analysis is that care must be used in pooling. Sometimes pooling helps, sometimes it hurts, and its effect (good or bad) can be unbounded. Also discussed briefly are alternative pooling scenarios, for example complete pooling of only queues which results in an M/PH/S system, or partial pooling which can be devastating enough to turn a stable Jackson network into an unstable Bramson network. We conclude with some possible future research directions.

## 1. Introduction

A fundamental problem in the design and management of stochastic service systems is that of pooling, namely the replacement of several ingredients by a functionally equivalent single

ingredient. We analyze the pooling phenomenon within the framework of queueing networks where in our case, as will be explained momentarily, it can take one of three forms: pooling queues (the demand), pooling tasks (the process) or pooling servers (the resources). Here we consider pooling queues and servers simultaneously, but keep the task structure intact, and we provide an efficiency index (5) to determine when such pooling is or is not advantageous.

Our models are described in terms of customers who seek *service* provided by *servers*. Service amounts to a collection of *tasks*, of which there are a finite number of *types*. Two main models are considered: in the first *specialized* model, each task type has a server and a queue dedicated to it. For example, Figure 1 exhibits a queueing network in which every customer requires a service that constitutes three tasks, and the tasks are carried out

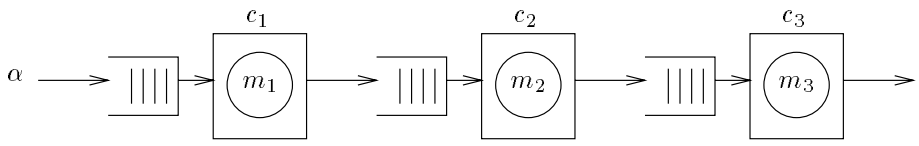


Figure 1: A specialized model with tasks attended by specialized servers.

successively, each by its own specialized server. Customers arrive at rate $\alpha$, average task durations are $m_k$ and servers' capacities are $c_k$. In the second *flexible* model, servers are capable of handling all tasks and they collectively attend to a single queue of services. For example, Figure 2 exhibits such a model, which arises through pooling the tandem network from Figure 1: customers arrive at rate $\alpha$, seeking the same three-task service as before; they all join a single queue, which is now attended by a single flexible server of capacity $\sum_k c_k$.

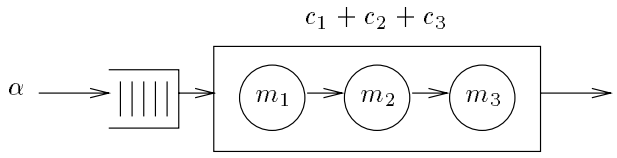


Figure 2: A flexible model with complete pooling into a single queue and a single flexible server.

Customer arrivals are assumed Poisson and task durations exponential. (We comment on these distributional assumptions in the Addendum.) As articulated in Section 2, we allow a service to consist of a random sequence of tasks in a way that the service duration has a phase-type distribution (a phase corresponds to a task). The specialized (unpooled) model turns out to be a Jackson network [19], as in Figure 3, and the flexible (pooled) architecture is modeled by an M/PH/1 system [26], as in Figure 4.

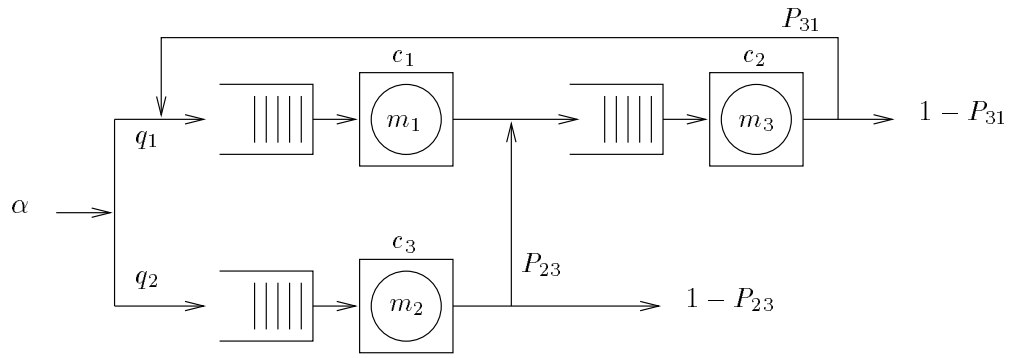


Figure 3: A specialized model with task repetition and feedback.

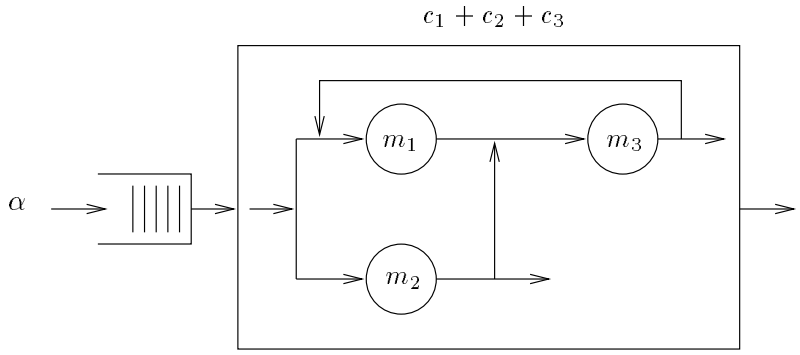


Figure 4: The flexible model, under complete pooling, that corresponds to Figure 3.

In addition to the above two main models, we also consider briefly alternative designs of pooling. For example, Figure 5 depicts the network from Figure 1, with its queues pooled into a single queue and the servers made flexible while still maintaining their individual identities (see Section 5.3). Figure 6 depicts partial pooling of only queues and servers 1

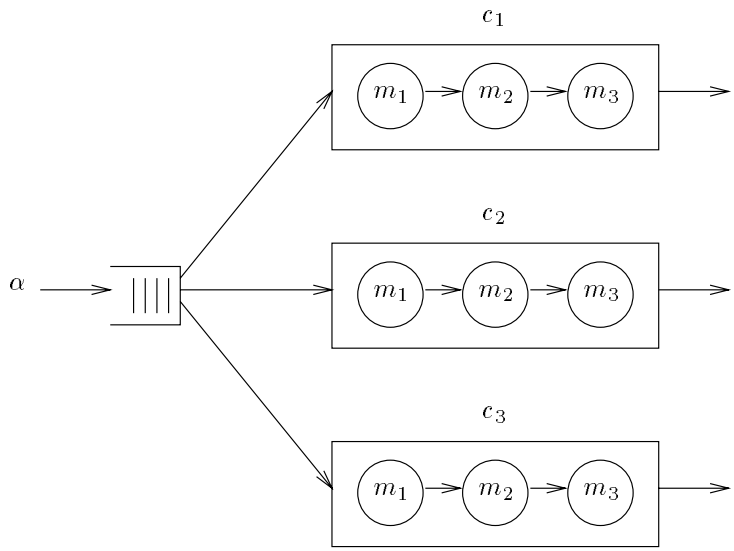


Figure 5: Complete pooling of queues only; servers are made flexible but maintain individual identities.

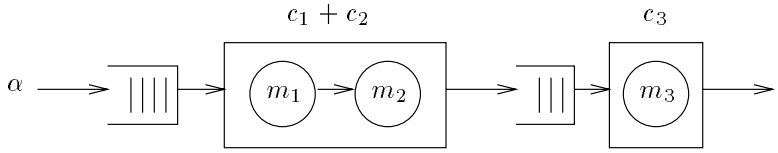


Figure 6: Partial pooling.

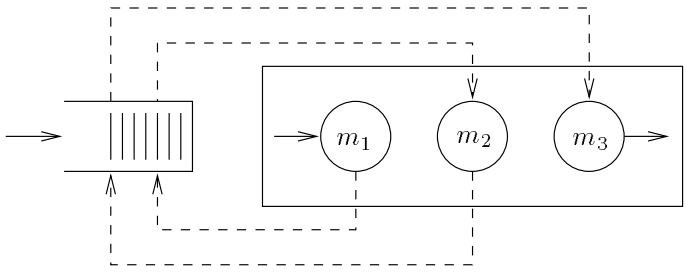


Figure 7: Splitting services. Each task returns to the end of the queue.

and 2 (see Section 5.4). Figure 7 depicts a split of the service so that a customer, upon completion of a task, rejoins the queue (see Section 5.5), and additional designs are possible

as well. A common feature of our models is that service is unaltered. For example, in Figures 1, 2, 5, 6 and 7, service always consists of tasks 1, 2 and 3 in succession.

## 1.1. Motivation

The present research arose from an analysis of a service network consisting of several specialized departments. The network was redesigned as a pooled single department, which was still responsible for the same services but whose servers were flexible enough to process all tasks. In trying to analyze this transition, we found that prevalent pooling models failed to cover our *network* scenario.

Our models provide a new simple framework that helps in assessing the effects on pooling of *utilization*, *variability* and service *design*. While this is not aimed as a review paper, our framework also relates, as it happens, rather disparate concepts and results, for example [6, 19, 22, 26, 35, 37]. We believe that the usefulness of the framework goes beyond the original motivating applications, pertaining to the design of telephone call centers [7], evaluation of communication networks [35], evolution (growth) of computer systems [21], group-technology in manufacturing [8], team-based product development [1], business reengineering [10, 13, 14, 24] (elaborated on below), and more.

Prior work on pooling seems to fall mainly into two categories: pool queues or pool servers. (In most of our analysis we do both.) As an example, pooling only queues would change several $M/M/1$ queues, say $K$, with arrival rate $\lambda$ and service rate $\mu$ into an $M/M/K$ queue with arrival rate $K\lambda$ and service rate $\mu$; results of this flavor are contained in [32, 35]. Pooling only servers would change an $M/M/K$ queue with arrival rate $\lambda$ and service rate $\mu$ into an $M/M/1$ queue with arrival rate $\lambda$ and service rate $K\mu$; pooling of this type is considered in [36]. For an illuminating depiction of these common pooling models, see [21], Fig. 5.5. Pooling also arises as an asymptotic phenomenon under appropriate rescaling of time and space [23, 29, 30]: for example, in heavy traffic, appropriate routing has the effect

of pooling servers (hence, in heavy traffic, the performance of the systems in Figures 2 and 5 coincides).

Our paper is in concert with current emphasis on business process reengineering [13, 14]. Indeed, referring to pooling as "integration of work," Loch [24] predicts that "the one idea from the reengineering era most likely to persist is that of integrated work." Similarly, in summarizing [13, 14], Buzacott [10] has "several tasks combined into one" as the first assertion of the superiority of a system that is designed using reengineering principles.

Both [10] and [24] use tasks in series and the transition from Figure 1 to Figure 5 as their paradigm for pooling in reengineering. It was shown in both [10] and [24] that the pooled system (with a single queue) is superior to the unpooled alternative, and higher task variability makes the advantage greater. The network-framework that we provide allows the results of [10] and [24] to be viewed in a more illuminating perspective. First we show, in Section 5.1, that pooling a tandem structure is always advantageous but for more general architectures this need not be the case. In particular, it has been known [35] that pooling a parallel structure can sometimes hurt unboundedly (see Section 5.2); we add the observation that partial pooling can turn a stable system unstable (Section 5.3, based on results of [6]). Second, the variability considered in [10, 24] is only *task* variability. In general, however, there are additional sources of variability, and their effects on pooling, as we now discuss, can be opposite to the variability effects in [10, 24]. Variability may be either predictable or stochastic: first-order sources for predictable variability are service design (e.g. scheduling tasks in tandem vs in parallel) or heterogeneity across task types (e.g. varying means); second-order sources for stochastic variability are, for example, fluctuations of task durations within a task type (e.g. due to human factors).

Our framework allows the consideration of both predictable and stochastic service variability. The sources that we explore here, however, are mainly first-order *structural*, since

the tasks that constitute a service are not altered. (Stochastic task variability is fixed by assuming an exponential duration for each task type.) In broad terms, with task variability fixed and workload approximately balanced, the design of the service determines its variability and, in turn, the effect of pooling: as servers' utilization increases and service variability *decreases*, pooling advantages are found to increase. This explains the apparent contradiction with the conclusion of [10, 24]. (Note that a balanced workload need not be optimal; see Section 4.)

## 1.2. Summary

The specialized and flexible models are introduced in Section 2. We start with a crude stability analysis in Section 2.3, showing that flexibility increases the workload that a specialized system can handle; see also Section 2.1 in Buzacott [10].

In Section 3 we quantify the effects of pooling in terms of an efficiency index (5), which is the product of a utilization factor $\mathcal{E}_u$ and a variability factor $\mathcal{E}_v$. We show that pooling always helps in light traffic, since a customer at the pooled system typically enjoys a service rate that is the total capacity of the specialized system. In heavy traffic, pooling effects can go either way.

For given arrivals and services, resource utilization is determined by how capacity is allocated among the servers. In Section 4 we use the square-root allocation of Kleinrock [21] to show that optimal capacity allocation mitigates the advantage of pooling. This advantage also decreases as variability increases. Indeed, crude analysis of the efficiency index (5) reveals the insight that with low enough variability pooling is always advantageous.

In Section 5 we explore both network and pooling designs. Sections 5.1 and 5.2 treat tandem and parallel systems respectively. With tandem tasks the structural variability is small enough so that pooling always helps. For parallel tasks, as already discovered by Smith and Whitt [35], the effect can go either way. In Section 5.3 we consider pooling queues only,

as in Figure 5. Performance is worse than with pooled servers, with the difference being maximal in light traffic and diminishing in heavy traffic. In Section 5.4 we investigate the effect of pooling design by considering partial pooling, as in Figure 6. It turns out that one can interpret the recent results in Bramson [6] to show that partial pooling can turn a stable system unstable. In Section 5.5 we require service splitting, as in Figure 7, rather than pursuing service until all of its tasks are completed. Through an example we show that the relative performance of these two systems depends on the structural variability of the total service time. There are numerous additional pooling issues that can be pursued, within the framework opened up here. Some are briefly discussed or mentioned in the concluding Section 6.

# 2. The Models

In our two models, customers arrive for service according to a Poisson process, at a rate of $\alpha$ per unit of time. A service constitutes a random sequence of tasks. There are $K$ types of tasks, indexed by $k = 1, \ldots, K$, and we refer to a task of type $k$ as simply task $k$. The work content in task $k$ is exponentially distributed with mean $m_k$. Let $q_k$ be the probability that task $k$ is first in a given service, and let $P_{jk}$ be the probability that task $k$ is a direct successor of task $j$; $1 - \sum_{k=1}^{K} P_{jk}$ is therefore the probability that service ends after task $j$.

Assume that arrivals of customers are independent of services and that, within each service, sequencing of tasks and task durations are all mutually independent. Also assume that each service constitutes a finite number of tasks with probability one; this is equivalent to the existence of the matrix

$$R = [I - P]^{-1},$$

where $P$ is the $K$-dimensional matrix $P = [P_{jk}]$. (The element $R_{jk}$ is the expected number of times that a task $k$ is performed during a single service, given that $j$ is the task to start

that service.)

To sum up, customer arrivals are characterized by a scalar $\alpha$, and services by a triplet $(q, P, m)$: $q^T = (q_1, \ldots, q_K)$, $P = [P_{jk}]$ and $m^T = (m_1, \ldots, m_K)$. (It is naturally assumed that $K \geq 2$, $\alpha > 0$, $m > 0$ and $q^T R > 0$.) Servers will be characterized momentarily, as they are model-dependent.

## 2.1. The specialized model

In the specialized model, every task $k$ has a server $k$ *dedicated* to it, whose service capacity is $c_k > 0$ units of work per unit of time. It follows that the processing times of task $k$ by server $k$ are i.i.d., each distributed exponentially with mean $m_k/c_k$. Furthermore, envisioning tasks of every type queueing up for processing at their respective dedicated servers, our specialized model reduces to an open Jackson network [19] with $K$ single-server stations, arrival rates $\alpha q^T$, service rates $c_k/m_k$ and a routing matrix $P$. (See Figure 3.)

We assume that the specialized system is stable (ergodic). This entails that each server $k$ has traffic intensity less than unity:

$$\rho_k^s = \frac{\lambda_k m_k}{c_k} < 1;$$

here $\lambda = \alpha q^T R$ is the vector whose $k$-th coordinate $\lambda_k$ stands for the effective arrival rate (in units of task $k$) to server $k$. Equivalently, stability prevails if and only if

$$\alpha < \alpha^s = \bigwedge_k \frac{c_k}{(q^T R M)_k},$$

where $M$ is the $k$-dimensional diagonal matrix, with $M_{kk} = m_k$, $k = 1, \ldots, K$. This is a consequence of the representation

$$\rho_k^s = \alpha \frac{(q^T R M)_k}{c_k}.$$

## 2.2. The flexible model

In the flexible model, customers arrive for service as before, but now they obtain service from a single *flexible* server, whose service capacity is $c^T e$ ($e$ is the $k$-dimensional vector of one's.) Services are as above, hence the work content in services are i.i.d., each with a phase-type distribution [26] that is characterized by the triplet $(q, P, m/c^T e)$: there are $K$ phases each corresponding to a task, the duration of phase $k$ is exponential with mean $m_k$, the initial phase is chosen according to $q$ and successive phases according to the routing matrix $P$. In other words, the flexible model reduces to an $M/PH/1$ queue, in which the average work content in a service is $q^T RMe$ and the server's capacity is $c^T e$; the average service time is therefore $q^T RMe/c^T e$. (See Figure 4.) We assume that this queue is stable (ergodic), which entails that its traffic intensity satisfies

$$\rho^f = \alpha \frac{q^T RMe}{c^T e} < 1.$$

Equivalently, stability prevails if and only if

$$\alpha < \alpha^f = \frac{c^T e}{q^T RMe}.$$

## 2.3. Stability analysis

The flexible system can handle any load that the specialized one can. This is formalized in terms of each of the following two inequalities:

$$\alpha^s \leq \alpha^f, \qquad \text{or} \tag{1}$$

$$\rho^f \leq \bigvee_k \rho_k^s. \tag{2}$$

To verify (1), note that for any positive vectors $a$ and $b$,

$$\frac{a^T e}{b^T e} \in \left[ \bigwedge_k \frac{a_k}{b_k}, \bigvee_k \frac{a_k}{b_k} \right], \tag{3}$$

since the left hand-side is a convex combination of $a_k/b_k$, $k = 1, \ldots, K$, namely,

$$\frac{\sum\limits_k a_k}{\sum\limits_k b_k} = \sum_k \frac{b_k}{\sum\limits_j b_j} \frac{a_k}{b_k}.$$

Letting $a_k = c_k$ and $b_k = (q^T RM)_k$ establishes (1). Similarly, letting $a_k = (q^T RM)_k$ and $b_k = c_k$ yields

$$\rho^f = \sum_k d_k \rho_k^s, \qquad d_k = c_k / c^T e, \tag{4}$$

which implies (2).

If $\alpha > \alpha^f$ then the flexible and specialized models are both unstable (have no steady state), hence a steady-state comparison between them is vacuous. If $\alpha \in [\alpha^s, \alpha^f)$, then the specialized model is unstable while the flexible one is stable, in which case pooling is advantageous trivially. One is left with $\alpha < \alpha^s \leq \alpha^f$, which will be assumed from now on.

# 3. Performance analysis

Let $W^s$ and $W^f$ denote the steady-state average sojourn-times in the specialized and flexible models respectively. Then

$$W^s = \frac{1}{\alpha} \sum_k \frac{\rho_k^s}{(1 - \rho_k^s)}$$

by Little's law and Jackson's characterization of individual stations as $M/M/1$ queues in steady state. For the flexible model, the Pollaczeck-Khintchine formula yields

$$W^f = E(S) \left[ 1 + \frac{\rho^f}{(1 - \rho^f)} \frac{1 + C^2(S)}{2} \right].$$

Here $S$ is a phase-type random variable characterized by $(q, P, m/c^T e)$, whose moments are given by [26]

$$E(S^n) = \frac{n!}{(c^T e)^n} q^T (RM)^n e, \qquad n \geq 1,$$

and whose squared coefficient of variation is $C^2(S) = Var(S)/E(S)^2$. Define the *efficiency index* of pooling to be $\mathcal{E} = W^s/W^f$. Then pooling is advantageous, as far as average sojourn

time is concerned, when $\mathcal{E} > 1$. Simple algebra leads to the representation

$$\mathcal{E} = \frac{\frac{1}{K}\sum_k \frac{\rho_k^s}{1-\rho_k^s}}{\frac{\rho^f}{1-\rho^f}} \cdot \frac{K}{(1-\rho^f) + \rho^f \frac{1+C^2(S)}{2}} , \tag{5}$$

in which

$$\frac{1+C^2(S)}{2} = \frac{E(S^2)}{2E(S)^2} = \frac{q^T(RM)^2 e}{(q^T RMe)^2} .$$

We write $\mathcal{E} = \mathcal{E}_u \mathcal{E}_v$, where

$$\mathcal{E}_u = \frac{\frac{1}{K}\sum_k \frac{\rho_k^s}{1-\rho_k^s}}{\frac{\rho^f}{1-\rho^f}} \qquad \text{and} \qquad \mathcal{E}_v = \frac{K}{(1-\rho^f) + \rho^f \frac{1+C^2(S)}{2}}$$

are the *utilization* index and *variability* index respectively. They represent the effects of utilization and variability on pooling efficiency, the analysis of which constitutes the rest of the paper.

*Ranges of the indices:* The ranges are given by

$$\mathcal{E}_u \in (\frac{1}{K}, \infty), \qquad \mathcal{E}_v \in (0, 2K),$$

and

$$\mathcal{E} \in (0, \infty);$$

the indices can take on all values within the specified intervals, and the end-points of the intervals provide tight asymptotic bounds. To elaborate, let

$$f(x) = \frac{x}{(1-x)}, \quad 0 \le x < 1,$$

a strictly convex, strictly increasing function with $f(0) = 0$ and $f(1-) = \infty$. Then, using (4), these properties imply

$$\mathcal{E}_u = \frac{\frac{1}{K}\sum_k f(\rho_k^s)}{f(\sum_k d_k \rho_k^s)} > \frac{\frac{1}{K}\sum_k f(\rho_k^s)}{\sum_k d_k f(\rho_k^s)} \ge \frac{1}{K}\frac{c^T e}{\bigvee_k c_k} \ge \frac{1}{K}.$$

One way for $\mathcal{E}_u \downarrow \frac{1}{K}$ is to let $\rho_k^s \downarrow 0$, for all $k \geq 2$, while maintaining $\rho^f \approx \rho_1^s$, both being bounded away from 0. This requires that $c_1/c^T e \uparrow 1$. On the other hand, $\mathcal{E}_u \uparrow \infty$, for example, as $\alpha \downarrow 0$, $\alpha/c_k \downarrow 0$ for all $k$, and $c_j/c_k \uparrow \infty$ for some pair $j \neq k$.

Turning to $\mathcal{E}_v$, the upper-bound $2K$ is an immediate consequence of $C^2(S) \geq 0$ and $\rho^f \leq 1$. The lower bound 0 is approached as $C^2(S) \uparrow \infty$ while maintaining $\rho^f$ bounded away from 0. Finally, the ranges of $\mathcal{E}$ will emerge during later analysis.

*Observations on variability and utilization:* If variability is low enough, formally if $C^2(S) < 1$, then $\mathcal{E} > 1$ since $\mathcal{E}_v > K$; in other words pooling is advantageous. If utilization is balanced, formally if $\rho_k^s = \rho_j^s$, $\forall j, k$, (hence also $\rho^f = \rho_k^s$, $\forall k$,) then $\mathcal{E}_u = 1$ and pooling efficiency is determined by the variability index. In particular, increasing utilization ($\rho^f \uparrow 1$) and reducing variability ($C^2(S) \downarrow 0$) attains the maximum pooling efficiency achievable under balanced utilization ($2K$).

*Light traffic:* In light traffic, the pooled system is always better because its customers are served at the pooled capacity of all specialized servers. Formally, light traffic prevails as $\alpha \downarrow 0$, while keeping the other parameters unaltered. Let $\mathcal{E} = \mathcal{E}(\alpha)$ in (5). Then

$$\lim_{\alpha \downarrow 0} \mathcal{E}(\alpha) = \sum_k \frac{(q^T RM)_k}{c_k} / \frac{q^T RM e}{c^T e} > 1 \ , \tag{6}$$

since the limiting efficiency belongs to the interval $\left[\frac{c^T e}{\vee_k c_k}, \frac{c^T e}{\wedge_k c_k}\right]$, in view of (3). Pooling, therefore, is always better in light traffic, and it is $K$ times better when the $c_k$'s are all equal.

Equation (6) can be explained with light traffic theory [31]. The light traffic limit of the mean sojourn time is the mean sojourn time of a single customer that moves alone through the system. For the pooled system, this time is $q^T RM e/c^T e$. For the specialized system, the mean sojourn time is $\sum_k (q^T RM)_k/c_k$, since the $k$-th summand is the total time at station $k$.

*Heavy traffic:* There are two cases of heavy traffic: $\alpha^s = \alpha^f$ and $\alpha^s < \alpha^f$. We consider the case $\alpha^s = \alpha^f$ here, and treat $\alpha^s < \alpha^f$ at the end of Section 4. The equality $\alpha^s = \alpha^f$ occurs if and only if $\alpha^s = c_k/(q^T RM)_k$ for all $k$, in which case $\rho_k^s = \rho^f$ for all $k$. Let $\mathcal{E} = \mathcal{E}(\rho)$ in (5), where $\rho$ denotes the common utilization. Then $\mathcal{E}_u = 1$, and

$$\lim_{\rho \uparrow 1} \mathcal{E}(\rho) = \frac{2K}{1 + C^2(S)} \ .$$

This finite limit prevails even though, as $\rho \uparrow 1$, both $W^s(\rho)$ and $W^f(\rho)$ grow unboundedly. Indeed, as $\rho \uparrow 1$,

$$(1-\rho)W^s(\rho) \to KE(S), \quad (1-\rho)W^f(\rho) \to E(S)\frac{1 + C^2(S)}{2} \ .$$

# 4. Division of Work, or Capacity Allocation

Fix $\alpha$, $q$, $P$ and $m$. Introduce an additional scalar $\gamma > 0$, to be interpreted as total available service capacity, and consider positive vectors $c$ such that $c^T e = \gamma$. As before, $\rho^f = \alpha q^T RMe/c^T e = \sum\limits_k \lambda_k m_k / c^T e < 1$, hence $\rho^f$ is fixed, $\gamma > \sum\limits_k \lambda_k m_k$, and $\mathcal{E}_v$ is also fixed. It follows that, as a function of $c$, the index $\mathcal{E} = \mathcal{E}(c)$ is minimized by the solution to

$$\min_c \sum_k \frac{\lambda_k m_k}{c_k - \lambda_k m_k} \ ,$$

$$\text{s.t. } \sum_k c_k = \gamma, \quad c \geq 0 \ .$$

This is Kleinrock's well-known capacity allocation problem [21, Section 5.7], solved by the "square-root" allocation

$$c_k = \lambda_k m_k + \left(\gamma - \sum_j \lambda_j m_j\right) \frac{\sqrt{\lambda_k m_k}}{\sum\limits_j \sqrt{\lambda_j m_j}} \ .$$

The corresponding value of $\mathcal{E}$ is given by the product of $\mathcal{E}_v$, in which $\rho^f = m^T\lambda/\gamma$, with

$$\mathcal{E}_u = \frac{\frac{1}{K}\left(\sum\limits_k \sqrt{\lambda_k m_k}\right)^2}{\sum\limits_k \lambda_k m_k} = \frac{\frac{1}{K}\left(\sum\limits_k \sqrt{(q^T RM)_k}\right)^2}{\sum\limits_k (q^T RM)_k} \leq 1. \tag{7}$$

The last inequality is a simple consequence of the Cauchy-Schwartz inequality, that also guarantees equality to unity if and only if $(q^TRM)_k = (q^TRM)_j$, $\forall k, j$, in which case also $c_k = c_j$, $\forall k, j$. The quantity $(q^TRM)_k$ represents the amount of work of task $k$ that is embodied in an arrival. Hence, under the optimal capacity allocation, $\mathcal{E}_u = 1$ if and only if workload and capacity are both balanced.

Optimal capacity allocation typically results in uneven utilization of the servers (see also [11] and [17]), which in turn is associated with a smaller benefit from pooling. That is indicated by (7), from which it follows that $\mathcal{E} \leq 2K$; in words, pooling benefits do not exceed $2K$. This upper bound can be approached only in a balanced system that is both heavily utilized ($\rho^f \uparrow 1$) and almost deterministic ($C^2(S) \downarrow 0$).

*Heavy traffic, continued:* We can now treat the other case of heavy traffic, $\alpha^s < \alpha^f$. If $\alpha^s < \alpha^f$, and $\alpha \in [\alpha^s, \alpha^f)$, as observed in Section 1, the specialized model explodes while the flexible one is stable, so pooling is trivially advantageous. To allow for a meaningful comparison, fix $q, P, M$ and total capacity $\gamma$, then assume that for each $\alpha$, the specialized system employs the corresponding optimal capacity allocation. This makes $\alpha^s$ a function of $\alpha$, enforcing $\alpha^s(\alpha) \uparrow \alpha^f$, as $\alpha \uparrow \alpha^f$. Thus, both the specialized and flexible system approach heavy traffic, in a way that

$$\lim_{\rho^f \uparrow 1} \mathcal{E}(\rho^f) = \frac{\frac{1}{K}\left(\sum_k \sqrt{(q^TRM)_k}\right)^2}{\sum_k (q^TRM)_k} \frac{2K}{1 + C^2(S)} ,$$

by (5) and (7). The discussion that follows (7) applies here as well.

# 5. Design

This section is devoted to some effects of network design on pooling efficiency. In Subsections 5.1 and 5.2 respectively, we consider tasks that are processed in tandem and parallel. The pooling of queues only, as depicted in Figure 5, is briefly discussed in Subsection 5.3. We

then highlight possible negative effects of poor pooling design (with partial pooling) in Subsection 5.4. In Subsection 5.5 we consider the effect of having customers rejoin the queue for each task.

## 5.1. Tandem tasks

Here $C^2(S)$ is small enough to render pooling always advantageous. Indeed, for $K$ tasks in tandem, $q_1 = 1$ (so $q_k = 0$, $k \geq 2$), $P_{k,k+1} = 1$ for $k < K$, $\rho_k^s = \alpha m_k / c_k$, $\rho^f = \alpha m^T e / c^T e$, and $C^2(S) = m^T m / (m^T e)^2 \leq 1$. It follows that $\mathcal{E}_v \geq K$, hence $\mathcal{E} \geq 1$ since $\mathcal{E}_u \geq \frac{1}{K}$ always.

## 5.2. Parallel tasks

Here the effect of pooling can be good or bad. For $K$ tasks in parallel (each service consists of exactly one task, which is task $k$ with probability $q_k$), $P = 0$, $\rho_k^s = \alpha q_k m_k / c_k$, and $\rho^f = \alpha q^T m / c^T m$. The service time of the pooled system is hyper-exponential, hence $C^2(S) \geq 1$. This also follows immediately from

$$\frac{1 + C^2(S)}{2} = \frac{\sum_k q_k m_k^2}{(\sum_k q_k m_k)^2} \geq 1 \ , \tag{8}$$

where the last inequality is a consequence of viewing $\sum_k^K q_k m_k$ and $\sum_k^K q_k m_k^2$ as the first and second moments of a discrete random variable.

From $C^2(S) \geq 1$ follows that $\mathcal{E}_v \leq K$. Under optimal capacity allocation, in fact $\mathcal{E} \leq K$ by $\mathcal{E}_u \leq 1$. The upper bound $K$ is attained, for example, as follows: $\mathcal{E}_v = K$ when $m_k = m, \forall k$, since then $C^2(S) = 1$; $\mathcal{E}_u = 1$ by letting also $q_k = 1/K, \forall k$.

We now show that it is possible for a pooled system to be arbitrarily worse than the specialized one. To this end, we achieve $\mathcal{E} \downarrow 0$ by constructing families of parallel systems that adhere to optimal capacity allocation, implying $\mathcal{E}_u \leq 1$, while having $\rho^f$ fixed and $C^2(S) \uparrow \infty$, implying $\mathcal{E}_v \downarrow 0$.

One way is to follow [35], where there exist tasks which are both rare and "challenging".

Such tasks rarely challenge the specialized system but, when pooled, they delay all other tasks sufficiently to render pooling inefficient, unboundedly. The driver is variability, made high enough for $\mathcal{E} \downarrow 0$. To be specific, vary $q_k$ and $m_k$ in a way that does not change the $K$ products $q_k m_k$; simultaneously let, say, $m_1 \uparrow \infty$, while maintaining $m_k/m_1$ bounded for all $k \geq 2$; $\alpha$ is fixed to guarantee stability. ($q_1 \downarrow 0$ by $\rho_1^s = \alpha q_1 m_1/c_1 < 1$ and $m_1 \uparrow \infty$, thus tasks of type 1 are both rare and challenging.) It follows that $\rho^f$, the optimal capacity allocation, and the denominator in (8) are all constant, but $C^2(S) \uparrow \infty$ with the numerator of (8).

A second way is to have slow servers in addition to challenging customers. Specifically, in an optimal allocation, take $c_1 \uparrow \gamma$ and $q_1 \downarrow 0$ (hence $\sum_{k\geq 2} c_k \downarrow 0$ and $\sum_{k\geq 2} q_k \uparrow 1$: the servers $2, \ldots, K$ are the slow ones), while maintaining $\rho_1^s$ bounded away from 1. One can then show that $q_k m_k/q_1 m_1 \to 0$, $\forall k \geq 2$. By (8)

$$\frac{1 + C^2(S)}{2} \geq \frac{1/q_1}{\left(1 + \sum_{k\geq 2} q_k m_k/q_1 m_1\right)^2},$$

verifying that, again, $C^2(S) \uparrow \infty$.

## 5.3. Heterogeneous servers

There are situations in which servers cannot be pooled into a single server and, while still flexible, they must retain their individual identities. The flexible model would thus become a multi-server single station (M/G/S), with phase-type service and possibly heterogeneous servers, as depicted in Figure 3. Both systems enjoy the same stability region [4], nevertheless performance is now worse than with a single server because service is not always rendered at the maximal capacity $c^T e$. (This can be verified through coupling.)

A comparison between our specialized system and a flexible system with heterogeneous servers would require formulae for the M/PH/S queue with heterogeneous servers. Such formulae do not exist so we restrict our attention below to light and heavy traffic. In

specific cases, there exists approximations which enable certain (approximate) comparisons. For example, Buzacott [10] uses second-moment approximations to compare series systems (as in Figure 1 and 3) while varying stochastic variability of tasks.

In light traffic, the performance of the single server could be made better than the heterogeneous system by a factor of $c^T e / \bigvee_k c_k$. Indeed, by the light-traffic rationale [31], the mean sojourn time of the single-server system is $E(S)/c^T e$. For the heterogeneous system, assume that services are always performed by the fastest available server. The mean sojourn time, in light traffic, is then $E(S)/\bigvee_k c_k$, which yields the above factor.

The heavy traffic limit of the single server and the heterogeneous system coincide [18]. One expects, therefore, that the difference in performance between the systems is maximal in light traffic, and it diminishes as utilization increases to heavy traffic. A precise justification would require a comparison via stochastic ordering.

## 5.4. Partial pooling

In partial pooling, $K$ specialized servers are pooled into $K' < K$ servers, typically more flexible, thus resulting in a queueing network with $K'$ stations. In this section we show, by way of examples, that it is possible for partial pooling to make a stable system unstable. Our examples are based upon networks introduced by Bramson [6], which have opened up a yet uncharted research territory.

We start with a specialized system that is a tandem network, as in Subsection 5.1, with $K$ taken odd for notational convenience. Let

$$\begin{aligned} \alpha = 1; \qquad & m_2 = m_K = d, \quad m_k = \delta,\ k \neq 2, K\,; \\ & c_1 = 2\delta,\ c_2 = 1 - (K-3)\delta\,, \\ & c_3, c_5, \ldots, c_{K-2} = 2/(K-3)\,, \\ & c_4, c_6, \ldots, c_{K-1} = 2\delta\,, \\ & c_K = 1 - 2\delta\,. \end{aligned}$$

Bramson [6] chose first $\frac{399}{400} \leq d < 1$, then $K$ large enough for $d^{K-2} < 1/50$, and finally $\delta$ small enough so that $0 \leq \delta < (1-d)/50(K-2)^2$. The specialized network is, therefore, stable ($\rho_k^s < 1,\ 1 \leq k \leq K$) and its (complete) pooling, as in Subsection 5.1, is advantageous.

We consider now two (related) poolings. In the first, depicted in Figure 8, the $K$ servers are pooled into 3 servers as follows: server 1 attends to tasks 1 and $K$; server 2 serves

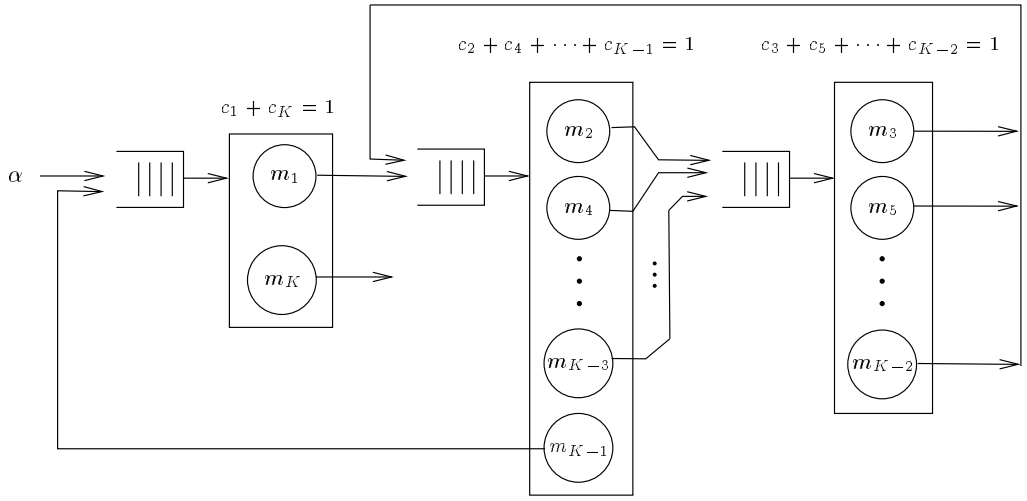


Figure 8: Bramson's unstable network obtained by partial pooling.

tasks $2, 4, \ldots, K-1$; server 3 cares for tasks $3, 5, \ldots, K-2$. Thus, a customer starts with server 1, moves on to 2, then 3, back to 2, and so on, until service $K-1$ at server 2, then the last service back at 1 and finally out. Each server uses the FIFO discipline, under which Bramson [6] proved that the network is unstable. (See his comment, immediately following the statement of Theorem 1.) In particular, with probability 1, the sojourn time of customers increases to infinity, as $t \uparrow \infty$. Instability arises because the system roughly alternates between busy periods of server 2, attending mainly to incoming tasks 2 while starving server 1, and busy periods of server 1, attending to tasks $K$ while starving server 2. The starvation of both servers is a consequence of FIFO, under which ample $\delta$-tasks are forced into queueing behind few $d$-tasks. (A more refined and quantitative intuition is provided in [6].)

The second pooling is into 2 servers as follows: server 1 serves tasks 1 and $K$ and server 2 attends to the rest. The service discipline is again FIFO, where immediate feedbacks at server 2 (of tasks $2, \ldots, K-2$) join the end of the queue, upon service completion. (There were no immediate feedbacks in our first example.) Thus, a service starts at server 1, moves on to 2 where it cycles for $K-2$ times, then back to 1 and out. Again, such a network was proved unstable in [6], Theorem 1, following the same rationale as above.

In the second pooling, server 2 could have served tasks $2, \ldots, K-2$ of a given service in succession, rather than separating the service so that a task joins the end of the queue upon service completion. Then the system would have been stable [4], which gives rise to the general issue of splitting services. We address this next.

## 5.5. Splitting services

Suppose that, after a task completion, each customer returns to the end of the (single) queue; see Figure 7. Thus, the queue consists of services that are at different stages of their processing. Although such a protocol seems naive, there are circumstances under which it is superior (in terms of mean wait) to having services carried out in an uninterrupted manner. For its performance analysis, one must retain task-identities in queue. An exact analysis is then possible [34], in terms of a set of linear equations whose solution yields mean waiting times. More explicit results can be obtained in heavy traffic [12, 28]. We just examine a special case, with the aim of showing that the advantage can go either way.

In our special case, all tasks have exponential service requirements with the same mean, $m$. This gives rise to a product-form system [5, 20], under which the distribution of total queue length is that of an $M/M/1$ queue with traffic intensity $\rho^f$. The sojourn time per 'pass' is thus $m/[c^T e(1-\rho^f)]$, and the mean number of passes through the queue is $q^T Re$.

If we let $W^n$ denote the mean sojourn time in the naive system, we obtain

$$W^n = \frac{m}{c^T e} q^T R e \left[ 1 + \frac{\rho^f}{1 - \rho^f} \right] .$$

For this case, $M = mI$, implying that

$$W^f = \frac{m}{c^T e} q^T R e \left[ 1 + \frac{\rho^f}{1 - \rho^f} \frac{1 + C^2(S)}{2} \right] .$$

We thus see that $W^n$ is less than (resp. equal to, greater than) $W^f$ if $C^2(S) > 1$ (resp. $C^2(S) = 1$, $C^2(S) < 1$): the naive protocol is superior under high variability. Note that, in this special case, $W^n \leq W^s$. (The comparison amounts to the inequality $\mathcal{E}_u \geq 1/K$, which was established in Section 3.)

# 6. Addendum

We conclude the paper with a discussion of our distributional assumptions and possible further research directions.

*Distributional Assumptions:* Only the exponential tasks require an elaboration since the role of the Poisson process as a model for exogenous random arrivals is well established. Empirical experience [3, 25] with human services supports the phase-type service structure, as evidence suggests that homogeneous human tasks are surprisingly often exponential. Admittedly, however, exponentially distributed task times will not be a good assumption for all applications. Then the simplicity of the resultant analysis becomes a driving motivation: explicit results make it easier to obtain insights from the analysis. (One could in fact analyze generally distributed tasks, in the spirit of [10, 38, 39] and in analytical support of [24]; this would require approximations of non-parametric Jackson networks and it is left as a possible avenue for future research.) Although there is a basis for questioning the universality of exponentially distributed task times, it should be pointed out that the distribution of the

total service time as a phase-type distribution is not a practical restriction because phase-type distributions are dense in the set of all distributions (see, for example [2]).

*Stochastic Ordering:* Most of our results invite finer comparisons, via various stochastic ordering schemes. For example, under what conditions would complete pooling of a tandem network lead to stochastically smaller sojourn times? (For an example of this type of result, see [9], Example 1.7.1.) Beyond the basic assessment of flexible vs. specialized models, other possibilities include parametric analysis, from light to heavy traffic (see the discussion at the end of Subsection 5.3) or an investigation of the effects of task-variability, for example refining [10].

*Control:* It is possible to maintain identities of tasks, or customer-types. One reason is to identify the types that benefit and those that suffer from pooling. More generally, this enables the incorporation of control (admission, sequencing, routing), with the goal of improving performance. Recall the devastating effects of FIFO, within the partial pooling of Subsection 5.3. Also note that pooling all servers into a single server while maintaining task identities raises the question of task sequencing, as analyzed in [15], [16], [22], [37]. With appropriate sequencing control (allowing preemption) the pooled system can always be made at least as good as the unpooled system. This is achieved by reproducing in the pooled system (using preemption) the performance of the original unpooled system. Preemption plays an important role here because with it certain customer types can be made effectively invisible to some other types, thus preventing the phenomenon of "challenging" tasks from Section 5.2. (The well-known formula for the waiting times in the M/G/1 queue with non-preemptive priorities [21] allow the reproduction of the arguments from Section 5.2.)

*Economies of Scale:* Consider a parallel specialized network, with statistically identical tasks and servers (equal $m_k$'s and $c_k$'s). Then $\mathcal{E} = K$ since $\mathcal{E}_u = 1$ and $C^2(S) = 1$. In words, pooling advantage equals the number of servers pooled. This is a manifestation of economies of

scale because the pooled larger-scale system can achieve, with higher utilization, the service level provided by the specialized system. Such higher utilization could be the outcome of reduced capacity, hence reduced cost. In the spirit of reengineering [24], one often seeks to take advantage of economies of scale (increasing $K$ by pooling), in a way that outweighs the variability overhead that ensue ($C^2(S)$ increasing). The desirable outcome is an operation that is as efficient as mass production ($\rho^f$ near unity) and as flexible as customized services (large $C^2(S)$), yet provides a very high operational service level (fast response, due to short and predictable sojourn times.)

The notion of flexible specialization [27] or mass customization is a current key concept in manufacturing strategy. This is also a main goal in the design of distributed telephone call centers [7] and packet switches for integrated broadband telecommunication networks [33]. The main obstacle to achieving this goal is the significant transactional overhead that arises due to pooling. Consider, for example, the time-overhead required for matching queueing customers to servers that become idle, in a face-to-face service operation with, say, 20 servers in parallel that attend to a single queue. Another interesting example involving overhead is to trade off transportation times in the specialized model (adding ample-server stations) against set-up times in the flexible model, due to switching among task-types.

The general issue here is cost/benefit analysis of economies of scale (increasing $K$ or $\rho^f$) in the presence of various pooling-dependent constraints and overheads. Special attention can be given to specific topologies, for example hub-networks.

*More Networks:* Analyze pooling within queueing networks that are richer in features and capabilities, for example fork-join networks, where one must also trade off the effects of coordination and synchronization; or finite-buffer networks, with various blocking protocols, giving rise to the option of pooling buffers.

# Acknowledgement

The comments of the area editor, associate editor and two anonymous referees helped turn a rather different first version of the paper into its more readable, so it is hoped, present form.

Part of the research of A.M. was carried out while visiting Bell Labs. The hospitality of the "Mathematics of Networks and Systems Research Department" is greatly appreciated.

# References


[1] Adler, P.S., Mandelbaum, A., Nguyen, V. and Schwerer, E., From project to process management: An empirically based framework for analyzing product development time, *Management Science* **41**, 458–484, March 1995.

[2] Asmussen, S., *Applied Probability and Queues*, Wiley, 1987.

[3] Asmussen, S., Nerman, O. and Olsson, M., Fitting phase type distributions via the EM algorithm, Preprint ISSN 0347-2809, Chalmers Univ., Goteborg, 1994.

[4] Baccelli, F. and Foss, S., private communication, 1996.

[5] Baskett, F., Chandy, K.M., Muntz, R.R. and Palacois, F.G., Open, closed and mixed networks of queues with different classes of customers, *J. Assoc. Comput. Mach.* **22**, 248–260, 1975.

[6] Bramson, M., Instability of FIFO queueing network, *Ann. Appl. Prob.* **4**, 414–431 (correction on p. 952), 1994.

[7] Brigandi, A.J., Dargon, D.R., Sheehan, M.J. and Spencer, T. III, AT&T's call processing simulator (CAPS): Operational design for inbound call centers, *Interfaces* **24**, 6–28, 1994.

[8] Burbidge, J.L., Production flow analysis for planning group technology, *J. of Operations Management* **10**, 5–27, 1991.

[9] Buzacott, J.A., Shanthilicmar, J.G. and Yao, D.D., Jackson network models of manufacturing systems, in *Stochastic Modeling and Analysis of Manufacturing Systems*, D.D. Yao, Ed., pp. 1–45, Springer, 1994.

[10] Buzacott, J.A., Commonalities in reengineered business processes: models and issues, *Management Science* **42**, 768–782, 1996.

[11] Calabrese, J.M., Optimal workload allocation in open networks of multiserver queues, *Management Science* **38**, 1792–1802, 1992.

[12] Dai, J.G. and Kurtz, T.G., A multiclass station with Markovian feedback in heavy traffic, *Math. of Operations Research* **20**, 721–742, 1995.

[13] Hammer, M., Reengineering work: Don't automate, obliterate, *Harvard Business Review*, July-August, 104–112, 1990.

[14] Hammer, M. and Champy, J., *Reengineering the corporation: A manifesto for business revolution*, Harper Business, 1993.

[15] Harrison, J.M., A priority queue with discounted linear costs, *Operations Research* **23**, 260–269, 1975.

[16] Harrison, J.M., Dynamic scheduling of a multiclass queue: discounted optimality, *Operations Research* **23**, 270–282, 1975.

[17] Hillier, F.S. and So, K.L., On the simultaneous optimization of server and work allocations in production line systems with variable processing times, ORSA/TIMS Meeting, Anaheim, CA 1991.

[18] Iglehart, D.L. and Whitt, W., Multiple channel queues in heavy traffic, I and II, *Adv. in Applied Prob.* **2**, 150–177 and 355–364, 1970.

[19] Jackson, J.R., Networks of waiting lines, *Operations Research* **5**, 518–521, 1957.

[20] Kelly, F.P., *Reversibility and Stochastic Networks*, Wiley, 1979.

[21] Kleinrock, L., *Queueing Systems, Vol. II: Computer Applications*, Wiley, 1976.

[22] Klimov, G.F., Time sharing service systems I, *Theory of Probability and its Applications* **19**, 532–551, 1974.

[23] Laws, C.N., Resource pooling in queueing networks with dynamic routing, *Adv. Appl. Prob.* **24**, 699–726, 1992.

[24] Loch, C.H., Operations management and reengineering, *European Management Journal*, **16**, 1998, to appear.

[25] Mandelbaum, A. Service Networks: Modelling, Analysis and Inference, Workshop on Stochastic Networks, IMA Minnesota, 1994.

[26] Neuts, M.F., *Matrix-Geometric Solutions in Stochastic Models*, Johns Hopkins Univ. Press, 1981.

[27] Priore, M.J. and Sabel, C.F., *The Second Industrial Divide: Possibilities for Prosperities*, Basic Books, 1984.

[28] Reiman, M.I., A multi-class feedback queue in heavy traffic, *Adv. Appl. Prob.* **20**, 179–207, 1988.

[29] Reiman, M.I., Some diffusion approximations with state space collapse, in *Modelling and Performance Evaluation Methodology*, F. Baccelli and G. Fayolle, Eds., pp. 209–240, Springer, 1984.

[30] Reiman, M.I. and Simon, B., A network of priority queues in heavy traffic: One bottleneck station, *Queueing Systems* **6**, 33–58, 1990.

[31] Reiman, M.I. and Simon, B., Open queueing systems in light traffic, *Math. of Operations Research* **14**, 26–59, 1989.

[32] Rothkopf, M.H. and Rech, P., Perspectives on queues: Combining queues is not always beneficial, *Operations Research* **35**, 906–909, 1987.

[33] Schwartz, M., *Broadband Integrated Networks*, Prentice Hall, 1996.

[34] Simon, B., Priority Queues with Feedback, *J. Assoc. Comput. Mach.* **31**, 134–149, 1984.

[35] Smith, D.R. and Whitt, W., Resource sharing for efficiency in traffic systems, *Bell System Tech. J.* **60**, 39–55, 1981.

[36] Stidham, S. Jr., On the optimality of single-server queueing systems, *Operations Research* **18**, 708–732, 1970.

[37] Tcha, D.-W. and Pliska, S.R., Optimal control of single-server queueing networks and multiclass M/G/1 queues with feedback, *Operations Research* **25**, 248–258, 1977.

[38] Wein, L.M., Capacity allocation in generalized Jackson networks, *O.R. Letters* **8**, 143–146, 1989.

[39] Whitt, W., The queueing network analyzer, *Bell System Tech. J.* **62**, 2779–2815, 1983.