OPTIMAL CONTROL OF A MULTICLASS, FLEXIBLE QUEUEING SYSTEM

NOAH GANS

The University of Pennsylvania, Philadelphia, Pennsylvania

GARRETT VAN RYZIN

Columbia University, New York, New York

(Received December 1995; revisions received September 1996; accepted November 1996)

We consider a general class of queueing systems with multiple job types and a flexible service facility. The arrival times and sizes of incoming jobs are random, and correlations among the sizes of arriving job types are allowed. By choosing among a finite set of configurations, the facility can dynamically control the rates at which it serves the various job types. We define system *work* at any given time as the minimum time required to process all jobs currently in the backlog. This quantity is determined by solving a linear program defined by the set of processing configurations.

The problem we study is how to dynamically choose configurations to minimize the time average system work. Using bounds and heuristics, we analyze a class of service policies that is provably asymptotically optimal as system utilization approaches one, as well as a policy that in numerical studies performs near-optimally in moderate traffic. Our analysis also yields a closed-form expression for the optimal, average work in heavy traffic.

This general problem has a number of applications in job shop and flexible manufacturing, in service organizations, and in the management of parallel processing and distributed database systems.

In many operating systems, jobs competing for the limited capacity of a service facility cause congestion. Classical queueing models have provided a great deal of insight into this phenomenon (see Wolff 1989 and Kleinrock 1976). In particular, they reveal the nonlinear, unbounded growth in queue lengths as utilization increases and the important role that variation plays in determining system performance. However, the traditional models of the service facility and of the processing requirements of jobs are quite simplified; the facility is modeled as a single server—or perhaps a collection of identical servers—and job requirements are defined only by scalar processing times.

In reality, many operating systems are considerably more complex. The service facility often has a complicated structure with a variety of resources such as pools of specialized labor, production equipment, data processing facilities, and supplies of material or energy. Jobs too are varied and complex, each one potentially requiring a different mix of the facility's resources. Moreover, the facility is often able to dynamically reallocate its resources in response to the size and mix of the backlog of jobs.

Yet most people who know queueing theory regularly extend the insights from basic queueing models to these more complicated settings. For example, students of queueing theory are not surprised if a plant manager reports a marked rise in backlogs as order volume increases, even though the plant is operating at less than full capacity. Likewise, they understand the reduction in customer service that a fast food outlet might suffer due to the increased variability caused by additions to its menu. Nevertheless, when pressed to extend these qualitative insights using quantitative queueing models, one is often confronted with a plethora of operational details that can confound even the most basic analysis.

One approach for extending queueing models to more complex systems is to consider networks of queues. In queueing network models, jobs of various classes follow either fixed or randomized routes through a network of processing nodes. Jobs wait for processing at each node and, when processed, move on to the next node on their routes. Over the past several decades, a vast literature and rich theory has been developed in this direction (for example, see Gelenbe and Pujolle 1987 and Walrand 1988).

While capturing considerably more processing complexity than classical queueing models, queueing networks have their own deficiencies as models of complex service facilities. The fixed or randomized routing ignores the significant control managers have over work flow in actual systems. In a flexible system, processing rates and work flows can often be adjusted by reallocating resources (labor, machinery, etc.) based on the current backlog of jobs.

Several researchers have addressed these limitations by considering control issues in queueing networks (for recent reviews of manufacturing applications see Bitran and Dasu 1992 and Buzacott and Shanthikumar 1992). However, doing so introduces significant analytical difficulties into an already intricate theory. Moreover, most queueing control formulations assume that the only management control is

Subject classifications: Queues: multiclass, flexible. Queues, limit theorems: heavy traffic, optimal expected system work. Production/scheduling, approximations/heuristics: job shops, flexible manufacturing/service systems, dynamic resource allocation.
Area of review: STOCHASTIC PROCESSES & THEIR APPLICATIONS.

0030-364X/97/4505-0677 $05.00

over which classes of jobs to process at the various nodes in the network, rather than how resources are allocated over production areas to achieve a mix of processing rates.

For example, in a classical job shop, jobs require certain machine tools and properly trained workers to operate the tools. Both resources—labor and equipment—are needed to complete a job, and there is a high degree of flexibility in pairing workers with equipment. A "processing node"—in the sense of a piece of equipment which is perpetually manned by an operator—does not really exist in a job shop setting. In addition, the processing rate of a job may depend on *which* set of resources (operators and equipment) is assigned to it. That is, the resource requirements of a job need not be unique, and hence an optimal assignment of resources to job types at any point in time will depend on the state of the backlog. Clearly, modeling the entire facility as a single node that services jobs with scalar processing requirements does not adequately capture the system's complexity.

Another example of this sort of complexity is found in parallel computing, where arriving jobs require simultaneous access to several processors in order to be served. A job type may be served equally well by a number of different subsets of a pool of processors, different types requiring different subsets from the pool. Again, the idea of a network of nodes breaks down. Are jobs-in-service at several nodes simultaneously? Does the network change each time a job enters or completes service? At the same time, a model that includes only a single node does not adequately capture the capacity to dynamically schedule the jobs being processed.

A flexible manufacturing system (FMS) provides a third example. An FMS consists of a set of numerically controlled machine tools connected by an automated conveyance system. The dynamics of the flow of individual parts through the system are quite complex. While there may be precedence constraints on the tasks required to complete a job, these tasks can often be performed by more than one machine. In addition, there may be limited parts storage at the input and output of each machine tool, and the conveyance system usually has limited capacity. As a result, blocking and other complex contentions can arise as parts move through the system. A traditional network model may fail to capture these complexities or prove intractable if all the details of the flow and control of part-movements are represented. Still, modeling the entire FMS as a single-node queue is clearly too simplistic.

Our model of a flexible service system attempts to bridge this gap between realism and analytical tractability. We use an aggregate representation of a flexible facility that serves randomly arriving jobs of m different classes. The arrival process is quite general as well, allowing for dependencies among various arriving job classes. The service facility is described by a collection of n processing *configurations*, each specifying a set of rates at which the m job classes are processed. The performance measure we consider is the total *work* in the system, defined as the minimal time needed to clear the current backlog of jobs using the collection of feasible configurations.

The problem we analyze is how to operate the facility—that is, choose processing configurations to use at any given point in time—so as to keep the long-run average work in the system as small as possible. We also seek insights into the performance of the system. Under what conditions is the system stable? How does the optimal average work depend on the model parameters, such as the collection of available processing configurations and the statistics of the job arrival process?

Our model is an aggregate representation of how a facility operates, and our definition of work is an aggregate measure of system performance. As such, our formulation suppresses much of the detail concerning how work is actually processed within the facility and how performance varies across job classes. This is clearly a limitation. The payoff, however, is that minimal structure is needed to represent the problem, and this provides significant modeling flexibility.

For example, in a job shop a configuration would specify the rates at which jobs are processed under a specific assignment of operators and equipment to each type of job. A job type (e.g., a machined engine part) might be processed at one rate on an automated machine tool requiring no operator, and at a different rate using a manually operated tool. In the case of an FMS, a configuration might correspond to one cyclic pattern of production that—based on actual past operating experience—we know can be achieved. Other such patterns might be identified as well. All of these would form the collection of processing configurations. Similarly, in the parallel processing example, configurations would represent feasible concurrent assignments of job types to processors; no particular structure on these feasible assignments is required. Thus, it does not matter how feasibility is determined in a given application; our model requires only a list of the feasible configurations themselves.

This modeling approach is directly analogous to—and indeed is strongly motivated by—set covering formulations in mathematical programming (for example, see Balinski and Quandt 1964 and Charnes 1956). The appeal of these formulations is their tremendous generality in handling a wide range of practical constraints, combined with their computational efficiency. (See Desrochers et al. 1992 and Ceria et al. 1995 for recent effective algorithms based on the set-covering formulation.) Our approach transfers many of these modeling and computational advantages to the domain of queueing control problems.

Our model has analytical advantages as well. Specifically, we are able to obtain bounds and provably good policies for operating the facility. These results, in turn, yield succinct, closed-form expressions for optimal system performance in heavy traffic. The simple equations provide insights into what drives system performance that generalize and reinforce the insights obtained from classical queueing models. Furthermore, the analysis also suggests additional heuristics that, in computational experiments, perform very well even in moderate traffic.

1. LITERATURE REVIEW AND OVERVIEW

There is previous research on more traditional queueing models that has sought to characterize the steady state backlog in systems with complex service requirements or constraints. Green (1980, 1984) and Brill and Green (1984) characterize three related versions of an $M/M/c$ queue in which arriving jobs require a random number of servers. In these papers the authors develop analytical characterizations of the distribution of waiting times, number in queue, and number of busy servers. Federgruen and Green (1984) develop approximations to the waiting time and number-in-queue distributions for an analogous $M/G/c$ queue. In all these analyses the queue discipline is first-in-first-out (FIFO); for systems with priority classes, the discipline is FIFO within class.

For three cases of the Markovian version of the problem, Green (1981) also compares FIFO to other service disciplines. For two of these, she shows that variations of a "smallest number of servers" priority policy outperforms FIFO in a number of performance measures.

Green (1984) extends the model she developed in 1980 to accommodate two classes of server, and in 1985 she studies the related "toll-booth problem," in which general-service and special-purpose servers serve two classes of jobs, one of which can use either type of server and another of which can only use general-purpose servers. For the former problem, she develops approximations to the mean delay and blocking probabilities for customers, and for the latter, approximations to the waiting time and number-in-queue distributions.

In the computer science and electrical engineering literature a number of papers address issues of *simultaneous resource possession*. Examples include time-sharing computer systems in which jobs require concurrent access to a partition of memory and the CPU, and telephone circuits in which calls require simultaneous possession of memory buffers and transmission links. Sauer (1981) and Jacobson and Lazowska (1982) develop approximations to first-order performance measures for examples of these computer systems. Whitt (1985) develops bounds and approximations for blocking probabilities in a generic loss system, which is intended to model the performance of packet-switched communication networks.

Kaufman and Wang (1989) analyze a simple variation of a two-class $M/M/1$ queue in which each job of one of the classes must possess one of a limited number of tokens to be processed by the server. For FIFO and processor-sharing service disciplines they develop approximations for the mean sojourn time in the system.

Courcoubetis et al. (1987) analyze stability conditions of the FIFO service discipline for a system with N servers in which there are blocking jobs (requiring all N servers) and simple jobs (requiring only one server). Courcoubetis and Reiman (1987) prove the optimality of a simple threshold policy for a variant of this system with an infinite supply of simple jobs, rewards for job completion, and holding costs. In comparison to our work, this model is more specialized; throughput is an important issue and the cost structure is more detailed. However, the dynamic element of control and the focus on optimal policies in this problem is quite similar in spirit to our work.

Ross and Tsang (1989) and Ross and Yao (1990) investigate the stochastic knapsack problem, a dynamic and stochastic version of the classic knapsack problem in combinatorial optimization. In the stochastic version, k classes of objects arrive at random intervals to a knapsack of fixed capacity. Arrivals for all object types are Poisson, the rate varying by class as well as with the state of the system. Every object within a class is of the same size, earns revenue at the same fixed rate, and has the same probability distribution for the length of time it stays in the knapsack, should it be packed. The knapsack is a loss system: if an arriving object cannot fit in the unused capacity of the knapsack, it is immediately lost. An optimal packing policy seeks to accept arriving objects to maximize the average rate of revenue accrual.

The stochastic knapsack differs from our work, however, both in its performance objective and in the complexity of the systems it models. While the knapsack is a revenue-maximizing loss system, we minimize system backlogs. While the knapsack represents systems whose performance is largely driven by a single shared resource, we seek to model systems which may have resource-sharing of arbitrary complexity.

There exists an extensive body of literature, beginning with Harrison (1988), which uses "Brownian network" approximations to queueing networks. The Brownian models explicitly allow for dynamic scheduling of different job classes at network nodes, and their analysis typically yields scheduling policies which are optimal for the approximations. Optimal policies for these Brownian networks, in turn, are interpreted in the context of the original queueing networks to provide effective policies for the original systems (for example, see Wein 1992).

These Brownian network models include a number of features, such as activity matrices and workload formulations, that are similar to the features of our model. As we noted in the introduction, however, the queueing networks which the Brownian models approximate are themselves restrictive in form. While we do not represent internal work flows, our model describes a more general class of service systems.

In the FMS literature, Kimemia and Gershwin (1983) use an aggregate representation of a flexible production facility that is similar to our approach. The FMS can process a mix of parts at deterministic rates, which are described by a polyhedral constraint set. In their work, variability is caused by the repair state of machines (working or not-working), and the polyhedron changes depending on the repair state. The objective is to control production rates to meet a set of planned output rates for the various parts, while minimizing a combination of inventory and shortage costs.

Exact results using this approach are analytically difficult to achieve and have been obtained only for the single part-type case (see Akella and Kumar 1986). Approximate methods, however, can handle quite complex systems. Sethi and Zhang (1994) and Chapter 9 in Gershwin (1994) provide surveys of results in this area.

While the polyhedral representation in these models is similar to ours, its source of uncertainty and the performance objectives are different. Our model corresponds more closely to a make-to-order or service system in which there is considerable demand uncertainty, and our objectives are to minimize order backlogs and to characterize optimal backlog performance.

Stability issues in multiclass systems similar to ours have been investigated in the past. Courcoubetis and Rothblum (1991) model a bin-packing system in which the server can decide the types of bins into which randomly arriving objects should be placed. These bins are analogous to our service configurations. Courcoubetis and Weber (1994) analyze a discrete-time production system in which both the quantities of arriving jobs and the service rates are stochastic. In both articles, the authors use polyhedral cones to define conditions for system stability. This linear algebraic approach is quite similar to some of the concepts used in our analysis. Just as their analyses extend the queueing concept of system stability to more complicated settings, our analysis extends the classical queueing characterization of expected system work. Bambos and Walrand (1993) analyze an elaboration of the parallel-processing system described in our introduction. Their work also concentrates on finding system stability conditions, and the stability results we present in this paper are a special case of theirs. However, they do not analyze other measures of system performance.

Finally, Krichagina et al. (1992) consider a cutting stock problem in which demands for different sizes of paper sheets arrive stochastically. The cutting facility can dynamically start and stop production of sheets, subject to a shutdown cost, and it can dynamically alter the cutting pattern. Using a linear program, the authors identify a fixed subset of the cutting patterns with which they construct a policy for operating the system. Due to the complexity of the problem, the authors rely on simulation to evaluate the effectiveness of the proposed policy. In our somewhat simpler setting, we are able to employ linear programming in a similar fashion to *analytically* characterize system performance and find *provably* good policies.

The remainder of this paper is organized as follows. Section 2 defines the model. Sections 3 and 4 then present the detailed analyses: lower and upper bounds on expected system work and a proof of asymptotic optimality for a class of heuristic policies. Section 5 then presents two additional heuristics, which we analyze numerically in Section 6. Finally, Section 7 presents four extensions that broaden the scope of the original model.

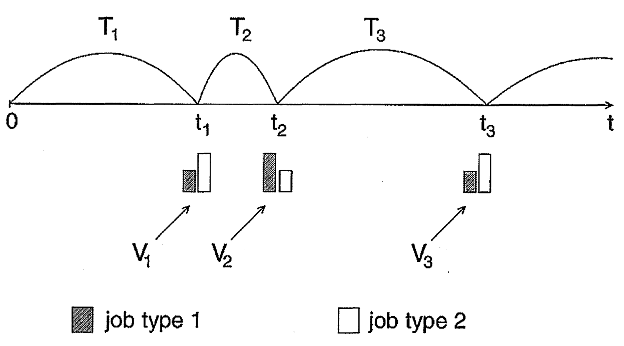

Figure 1. Example arrival process.

2. PROBLEM DEFINITION

In this section we define the queueing model and the measure of performance. After a brief summary of notational conventions, we present the elements of the arrival process, a model of the service facility, and a definition of admissible service policies. We then define *work* as a measure of system performance and the optimality of policies in terms the steady state expectation of system work.

2.1. Notation

When describing vectors, we use the following conventions. $\mathbf{R}^m$ is m-dimensional Euclidean space, and $\mathbf{R}^m_+$ is its nonnegative orthant. Boldface $\mathbf{0} \in \mathbf{R}^m$ represents a vector of zeroes, and boldface $\mathbf{1} \in \mathbf{R}^m$ a vector of ones. The vector $e^j \in \mathbf{R}^m_+$ has a one in the jth element and zeroes elsewhere. For $x \in \mathbf{R}^m$ and real $\epsilon > 0$, $\mathcal{N}(x, \epsilon_1) \overset{\text{def}}{=} \{y \in \mathbf{R}^m\colon y_i \in (x_i - \epsilon_1, x_i + \epsilon_1);\ i = 1, \ldots, m\}$ is the L_∞-neighborhood of x.

For vectors and matrices that are members of a sequence we use the following notation: a subscript denotes the vector's or matrix's place within the sequence, while a superscript indexes individual elements within the vector or matrix. For example, for the sequence of matrices $\{B_l \in \mathbf{R}^{m\times m}\colon l \geq 1\}$, B_l^{ij} denotes the ijth element of the lth matrix.

We follow these conventions when describing probabilistic events: $\{\cdot\}$ represents an event; $\mathbf{1}_{\{\cdot\}}$ denotes the indicator function of an event; and $P\{\cdot\}$ designates the probability of an event. The abbreviations *i.i.d.* and *a.s.*, respectively, stand for "independent and identically distributed" and "almost surely."

2.2. Job Arrival Process

Consider an arrival stream of jobs to be processed. There are m types of jobs, and at arrival epochs, $\{t_k\colon k = 1, 2, \ldots\}$, real vectors $V_k = (V_k^1, \ldots, V_k^m)$ of job *quantities* arrive to the system. The quantity associated with an arriving job is the amount of service effort required to process the arrival. That is, for a fixed rate of service, quantities are directly proportional to processing times. (Figure 1 shows a sample path of an arrival process for $m = 2$.)

Arrival epochs form a renewal process; if we set $t_0 \equiv 0$ and let $T_k \overset{\text{def}}{=} t_k - t_{k-1}$ denote the interarrival times, then

$T_1, T_2, \ldots$ are i.i.d. We assume $E[T] = 1/\lambda > 0$ and that the variance of T, denoted σ_T^2, is finite.

The sequence $\{V_k: k = 1, 2, \ldots\}$ is also assumed to be i.i.d. with expectation $E[V] = \gamma \in \mathbf{R}_+^m$ and variance-covariance matrix $\Gamma \in \mathbf{R}^{m \times m}$. We require that $\gamma > \mathbf{0} \in \mathbf{R}^m$ and that the distribution of V has bounded support, i.e., there exists a real number C_1 such that $V \leqslant C_1\mathbf{1}$ (a.s.); this of course implies that γ and Γ are finite. There are no other restrictions on γ and Γ. In particular, we note that there may be dependencies among the components of V. For example, more than one component of V_k may be positive, which would correspond to the simultaneous arrival of several types of jobs. Finally, we assume that the sequences $\{V_k\}$ and $\{T_k\}$ are independent of each other.

2.3. The Service Facility

The service facility has n service configurations available to process jobs. Each configuration, j, simultaneously processes the m types of jobs at constant rates given by a real vector, $a_j \in \mathbf{R}_+^m$. That is, each element, a_{ij}, represents the quantity of job type i that is processed in one unit of time under configuration j. The matrix $A \in \mathbf{R}_+^{m \times n}$ defines all n possible service configurations. We require only that A be nonnegative and have rank m.

At any time $t \geqslant 0$ the service facility can use a mixture (convex combination) of service configurations or be idle. In practice, this simply means the service facility is able to switch among its n configurations arbitrarily quickly. Formally, we represent the control action at time t by a vector $U_t \in \mathbf{R}_+^n$, which is restricted to the set $\{u: \mathbf{1}^\top u \leqslant 1, u \geqslant 0\}$. For example, if the service facility is idle, then $U_t \overset{\text{def}}{=} \mathbf{0}$; if the facility allocates its total effort to configuration j, then $U_t \overset{\text{def}}{=} e^j$.

The set of arrivals up to time t, $\{(t_k, V_k): t_k < t\}$, along with the sample path of processing configurations used by the service facility up to t, $\{U_r: r < t\}$, is called the *history* of the process up to t, and is denoted $\mathcal{H}_t$.

The history of the process determines the system backlog as follows. Let $Q_t = (Q_t^1, \ldots, Q_t^m)$ be the quantities of jobs in the system at t. Note that at arrival epochs, t_k, there is a discontinuity in Q_{t_k} due to the arrival of the kth vector of jobs, V_k. As a convention, we include V_k in Q_{t_k}, and we denote by $Q_{t_k^-}$ the quantities of jobs that the kth arriving vector *finds* in the system. $\{Q_t: t \geqslant 0\}$ obeys the recursion

$$Q_0 \equiv 0,$$

$$Q_t = \begin{cases} \left(Q_{t_k} - A \displaystyle\int_{t_k}^{t} U_r \, dr\right)^+, & t_k < t < t_{k+1}, \\ (Q_{t_k} - A \displaystyle\int_{t_k}^{t} U_r \, dr)^+ + V_{t_{k+1}}, & t = t_{k+1}. \end{cases} \tag{1}$$

Note that in (1), $A \int_{t_k}^t U_r \, dr$ may exceed Q_{t_k} in one or more of its components. One should think of the term $A \int_{t_k}^t U_r \, dr$ as the capacity—or service potential—of the system over $(t_k, t]$, rather than as the actual quantity of jobs processed.

A service *policy*, π, is a rule which, given $\mathcal{H}_t$, allows the service facility to determine which processing configurations to use at t. We consider only those service policies π for which U_t^π is adapted to $\mathcal{H}_t$ (i.e. U_t^π is nonanticipating and $\mathcal{H}_t$-measurable), and let Π denote the class of all such $\mathcal{H}_t$-adapted policies.

2.4. System Work and Optimization

Our basic measure of performance is the total *work* in the system, denoted W_t, where

$$\begin{aligned} W_t &= \min \sum_{j=1}^{n} x_j, \\ &\text{s.t.} \\ Ax &\geqslant Q_t, \\ x &\geqslant \mathbf{0}. \end{aligned} \tag{2}$$

W_t is the minimum time required by the service facility to clear the system starting at time t, assuming that no additional orders arrive after t. This definition of work is the same as that of *completion time* in Bambos and Walrand (1993) and is a direct generalization of the definition of work in system for simple single-server queues in Wolff (1989).

There are several reasons why we focus on system work (2) as a measure of performance. First, it is a natural and tangible measure of aggregate system congestion, describing, for example, how much time would be needed (under an optimal processing schedule) to clear the current backlog of jobs. Second, it is amenable to analysis, and hence helps provide insights into how system parameters affect congestion. However, in contrast to classical queueing models, finding policies that minimize work is not trivial in this model.

Finally, it appears that a characterization of system work is important in the analysis of other cost and service level measures. Indeed, we show in Section 7 that, for a wide class of cost measures, work forms the basis of a lower bound on system cost. As utilization grows in these cases, the service facility cannot control these costs without also effectively controlling system work.

Note that the sample paths of $\{Q_t: t \geqslant 0\}$ and $\{W_t: t \geqslant 0\}$ depend on the service policy, π, as well as on the sample sequences of interarrival times $\{T_k: k = 1, 2, \ldots\}$ and arrival quantities $\{V_k: k = 1, 2, \ldots\}$. When we wish to emphasize the dependence of Q_t and W_t on π, we will write Q_t^π and W_t^π.

The linear program (2) which defines W_t leads directly to a useful lower bound on the amount of work in the system at any time, t:

Lemma 1. *Let $y^* \in \mathbf{R}^m$ be the optimal solution to*

$$\begin{aligned} &\max \gamma^\top y, \\ &\text{s.t.} \\ y^\top A &\leqslant \mathbf{1}, \\ y &\geqslant \mathbf{0}. \end{aligned} \tag{3}$$

Then for any policy $\pi \in \Pi$

$$y^{*\top} Q_t^\pi \leqslant W_t^\pi.$$

Proof. Consider the dual of (2) for an arbitrary policy π: $\max\{Q_t^{\pi\top}y: y^\top A \leqslant \mathbf{1}, y \geqslant \mathbf{0}\}$. Clearly, y^* is a feasible solution for this problem. Therefore, by weak duality for linear programs, $y^{*\top}Q_t^\pi \leqslant W_t^\pi$. □

The linear program (3) is the dual of

$$\min \sum_{j=1}^{n} x_j,$$

s.t.

$$Ax \geqslant \gamma,$$

$$x \geqslant \mathbf{0}, \tag{4}$$

which minimizes the total time required to process an "expected" vector (γ) of arriving work. One can interpret y^*, the dual solution of (4), to be an allocation of work-content, or processing time, to the different classes of jobs. This allocation is important in characterizing system performance. In particular, we will show that $y^{*\top}\gamma$ represents the expected amount of work that enters the system at each arrival and that $\rho \overset{\text{def}}{=} \lambda y^{*\top}\gamma$ is an appropriate measure of utilization for the service facility. Note from (3) and (4) that $\gamma > \mathbf{0}$ implies that $y^{*\top}\gamma > 0$ as well.

We say that a policy π, is *stable* if

$$E_\pi[W] \overset{\text{def}}{=} \lim_{t\to\infty} \frac{1}{t}\int_0^t W_s^\pi\, ds,$$

exists and is finite. We define

$$W^* \overset{\text{def}}{=} \inf_{\pi\in\Pi} E_\pi[W], \tag{5}$$

and ideally we would like to find a stable policy, π^*, (if one exists) that achieves the infinum above. A weaker but more tractable notion of optimality than (5) is asymptotic optimality. Specifically, we call a policy, π°, *asymptotically optimal* in heavy traffic if

$$\lim_{\rho\uparrow 1} \frac{E_{\pi^\circ}[W]}{W^*} = 1,$$

where $\rho \uparrow 1$ indicates we approach the limit from below. In general, the limit $\rho \uparrow 1$ is shorthand for describing a sequence of stable queueing systems (indexed by n) for which V_n and T_n both converge in distribution and $\rho_n \uparrow 1$. (See Lemma 6 in the appendix.) To simplify the exposition, however, we will henceforth consider the distribution of the arrival sequence $\{V_k; k \geqslant 1\}$ (i.e., the mix of job quantities) to be *fixed*, and interpret $\rho \uparrow 1$ to mean an increase in the *rate* at which these job quantities arrive.

2.5. An Example

We next provide a brief example to illustrate the model. In Section 7 we return to this example for our numerical experiments. In our example (see Figure 2), a make-to-order flexible production system processes two job types. Type 1 jobs are processed at station $S1$ and then station $S2$; type 2 jobs are processed at station $S1$ and then station $S3$. Station $S1$ can process seven units of either job type 1 or 2 in a unit of time, station $S2$ can process four units of job type 1 in a unit of time, and station $S3$ can process five units of job type 2 in a unit of time. We assume there is a continuous flow of parts throughout the system and that no work in process is allowed between stations. In our model, this entire system of three stations is considered a *single* service facility, with each processing configuration defining a different pair of rates at which the facility can process job types 1 and 2.

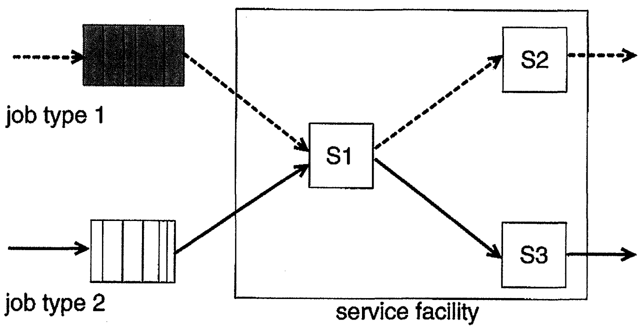

Figure 2. Example Service Facility.

In this example, we can construct the columns of A in a two-step process. First, we define the polyhedron of *feasible* processing rates as shown in Figure 3. Points in the polyhedron represent feasible sets of rates at which the service facility may simultaneously process the two types of jobs. Second, we use the extreme points as the columns of A, since any point in the polyhedron can be described as a convex combination of its extreme points (note that we exclude the column $(0, 0)^\top$ from A, since we consider an idle facility not to be actively using any processing configuration). The resulting A matrix is shown in Figure 3.

Remark. In this production system example, processing configurations are defined by the extreme points induced by the intersection of several linear resource constraints. Therefore, the number of extreme points can grow exponentially in the number of job types and resources. An alternative formulation can reduce the size of the LP in this case. Let $M = [m_{ij}]$, where m_{ij} is the amount of station

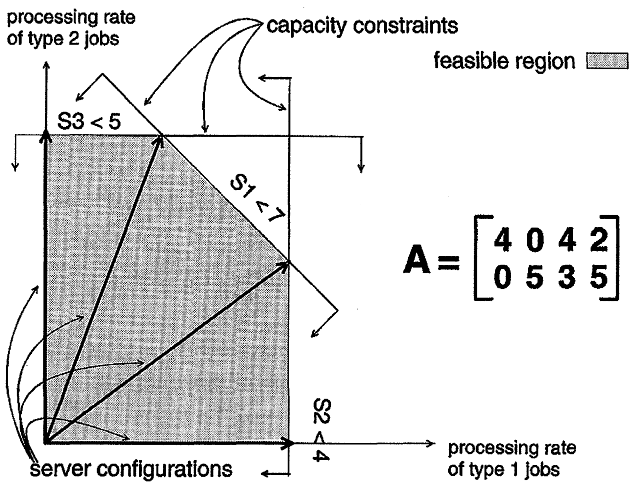

Figure 3. Model of example service facility.

i time required to process a unit of job type j, so M has one column for every job type and one row for every resource. For the example of Figure 2,

$$M = \begin{bmatrix} 1/7 & 1/7 \\ 1/4 & 0 \\ 0 & 1/5 \end{bmatrix}.$$

The work in the system is then given by

$$W_t = \min\{s\colon Mx \leqslant \mathbf{1}s;\ x \geqslant Q_t;\ s \geqslant 0\},$$

where s is the total time to process the backlog Q_t and x_j, $j = 1, 2$ represents the quantity of job type j processed. Though more concise, the formulation above is less general than the set covering representation (2), which allows for an essentially arbitrary structure among the columns of A.

3. LOWER BOUND PROCESS

We next use Lemma 1 to develop a stochastic process that is a sample-path lower bound on the work process under any policy, π. Define $\{\underline{W}_t\colon t \geqslant 0\}$ according to the recursion:

$$\underline{W}_0 \equiv 0,$$

$$\underline{W}_t \doteq \begin{cases} (\underline{W}_{t_k} - (t - t_k))^+, & t_k < t < t_{k+1}, \\ (\underline{W}_{t_k} - (t_{k+1} - t_k))^+ + y^{*\top} V_{k+1}, & t = t_{k+1}. \end{cases} \tag{6}$$

Lemma 2. *For all policies $\pi \in \Pi$, for all $t \geqslant 0$, $\underline{W}_t \leqslant W_t^\pi$.*

Proof. We will prove by induction that for any policy, π, $y^{*\top}Q_t^\pi \geqslant \underline{W}_t$ at all times. Together with Lemma 1 we then have $W_t^\pi \geqslant y^{*\top}Q_t^\pi \geqslant \underline{W}_t$ for all $\pi \in \Pi$ and $t \geqslant 0$.

For $t = 0$, $\underline{W}_t = W_t^\pi \equiv 0$. As an induction assumption, suppose that for some arbitrary arrival epoch t_k, $\{y^{*\top}Q_s^\pi \geqslant \underline{W}_s\colon 0 \leqslant s \leqslant t_k\}$, and consider the times up through the next arrival epoch, $t \in (t_k, t_{k+1}]$. First we consider $t_k < t < t_{k+1}$. Equation (1) gives us

$$y^{*\top}Q_t^\pi = y^{*\top}\left(Q_{t_k}^\pi - A\int_{t_k}^t U_r\,dr\right)^+$$
$$\geqslant y^{*\top}\left(Q_{t_k}^\pi - A\int_{t_k}^t U_r\,dr\right).$$

From the induction assumption, $y^{*\top}Q_t^\pi \geqslant \underline{W}_{t_k} - y^{*\top}A\int_{t_k}^t U_r\,dr$, and from (3) we know that $y^{*\top}A \leqslant \mathbf{1}$ so that $y^{*\top}Q_t^\pi \geqslant \underline{W}_{t_k} - \mathbf{1}^\top \int_{t_k}^t U_r\,dr$. Finally, $\mathbf{1}^\top U_t \leqslant 1$ implies that $\mathbf{1}^\top \int_s^t U_r\,dr \leqslant t - s$, so that $y^{*\top}Q_t^\pi \geqslant \underline{W}_{t_k} - (t - t_k)$. Since both y^* and Q_t^π are greater than or equal to zero, we conclude that for $t_k < t < t_{k+1}$

$$y^{*\top}Q_t^\pi \geqslant (\underline{W}_{t_k} - (t - t_k))^+ = \underline{W}_t. \tag{7}$$

Similarly, consider time t_{k+1}:

$$y^{*\top}Q_{t_{k+1}}^\pi = y^{*\top}(Q_{t_{k+1}^-}^\pi + V_{k+1}).$$

From (7) we then have $y^{*\top}Q_{t_{k+1}}^\pi \geqslant (\underline{W}_{t_k} - (t_{k+1} - t_k))^+ + y^{*\top}V_{t_{k+1}} = \underline{W}_{t_{k+1}}$. Since the proof relies only on sample paths, it clearly holds for any policy at all times. $\square$

Note that $\{\underline{W}_t\colon t \geqslant 0\}$ is the work in system process (Wolff 1989, p. 291) for a classical $GI/GI/1$ queue with i.i.d. interarrival times $\{T_k\colon k = 1, 2, \ldots\}$ and i.i.d. service times $\{Z_k\colon k = 1, 2, \ldots\}$, where

$$Z \overset{\text{def}}{=} y^{*\top}V,$$
$$E[Z] = E[y^{*\top}V] = y^{*\top}\gamma, \quad \text{and}$$
$$\sigma_Z^2 = \operatorname{var}(y^{*\top}V) = y^{*\top}\Gamma y^*. \tag{8}$$

We define $E[\underline{W}] \overset{\text{def}}{=} E[\lim_{t\to\infty}\underline{W}_t]$ to be the expected work found in the lower bound system in equilibrium. Again, if $\{\underline{W}_t\colon t \geqslant 0\}$ is a regenerative process, then

$$E[\underline{W}] = \lim_{t\to\infty}\frac{1}{t}\int_0^t \underline{W}_s\,ds, \tag{9}$$

as well.

For the original queueing system, $\{\underline{W}_{t_k^-}\colon k = 1, 2, \ldots\}$ represents a sample path lower bound on the work found upon arrival by the sequence of arriving vectors of job quantities. We define

$$E[\underline{D}] \overset{\text{def}}{=} \lim_{k\to\infty}\frac{1}{k}\sum_{j=1}^k \underline{W}_{t_j^-}, \tag{10}$$

so that $E[\underline{D}]$ is a lower bound on the expected work found by a new arrival to the original system, in equilibrium.

Since the lower bound is a $GI/GI/1$ queue, it immediately follows that when $\rho > 1$, $E[\underline{D}]$ is unbounded. This leads to the following result, which is a special case of Theorem 1 in Bambos and Walrand (1993) (we omit the proof):

Theorem 1. *If $\rho \overset{\text{def}}{=} \lambda y^{*\top}\gamma > 1$ then $E_\pi[W] = \infty$ for any policy $\pi \in \Pi$.*

Note that y^* in the above theorem is a function of both the processing capabilities of the facility, as given by A, and the product mix, as given by γ. Indeed, this stability result is equivalent to saying that if the system

$$\lambda y^\top \gamma \leqslant 1,$$
$$y^\top A \leqslant \mathbf{1},$$
$$y \geqslant \mathbf{0},$$

is not solvable in y, then no stable policy exists.

The $GI/GI/1$ structure of the lower bound also leads directly to the next theorem, which follows from Lemmas 6 and 7 in the appendix (we omit the proof):

Theorem 2. *If we scale T as in Lemma 6 so that $\rho \uparrow 1$ as $\lambda \uparrow 1/y^{*\top}\gamma$, then*

$$\lim_{\rho\uparrow 1}(1 - \rho)E[\underline{W}] = \frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^*)}{2},$$

which implies that for any policy π

$$\liminf_{\rho\uparrow 1}(1 - \rho)E[W^\pi] \geqslant \frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^*)}{2}.$$

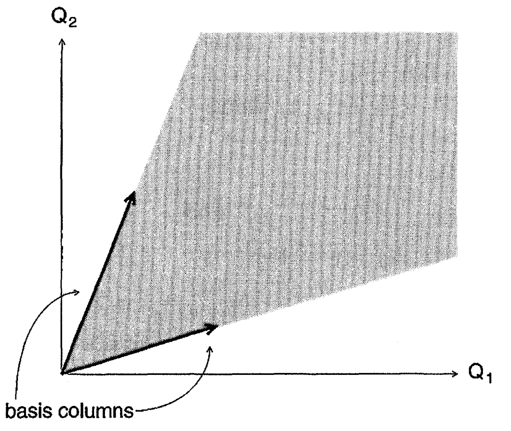

Figure 4. The cone of B.

4. A CLASS OF POLICIES AND AN UPPER BOUND

Our strategy in designing an effective heuristic policy is based on the following observations. First, the optimal basis, B, of the LP (4) identifies a set of efficient processing configurations. These efficient columns have reduced costs of zero, meaning in a unit of time they process one unit of work (as defined by the allocation y^*). Columns with positive reduced costs process *less* than a unit of work in one unit of time and hence are inefficient in this sense. Indeed, one can show that if, during a busy period, Q_t remains in the cone of the basis B *and* the service facility only uses columns of B for processing, then W_t will equal the lower bound $\underline{W}_t$ throughout the busy period.

Second, as shown in Figure 4, if Q_t lies in the cone of B, then $x = B^{-1}Q_t$ is a feasible solution to (2), and the associated processing time is $\mathbf{1}^\top x = y^{*\top}Q_t$, which *equals* the lower bound of Lemma 1. More precisely, if (3) has a unique solution, then (4) is nondegenerate, so there exists an $\epsilon > 0$ such that y^* remains the vector of optimal dual prices of (4) for all right-hand-side vectors in $\mathcal{N}(\gamma, \epsilon)$ (see Bazaraa et al. 1990, p. 260). Because (4) is homogeneous of degree one in its right-hand side, this means whenever there exists an $\alpha > 0$ such that for all $Q_t \in \mathcal{N}(\alpha\gamma, \alpha\epsilon)$, y^* is an optimal dual solution of (2), and hence the system work W_t equals the lower bound $y^{*\top}Q_t$.

These observations suggest that a policy should try to satisfy two objectives: (1) use the efficient configuration in B as much as possible, and (2) maintain the backlog Q_t close to the ray defined by γ. To accomplish these objectives, we propose a class of policies, $\{\pi_{B_N}\}$, in which the service facility processes arrivals in batches. First we describe the mechanics of how the policy operates. Then we demonstrate that the class of policies is asymptotically optimal in heavy traffic.

For the class $\{\pi_{B_N}\}$, the service facility acts as a *bulk service queue* in which the individual arrival vectors, $\{V_k: k = 1, 2, \dots\}$, are served in batches of N. We may think of bulk service as operating in two stages (see Figure 5). In the first stage, an *accumulator* collects batches of N arrivals. In the second, the *batch server* processes these batches of N.

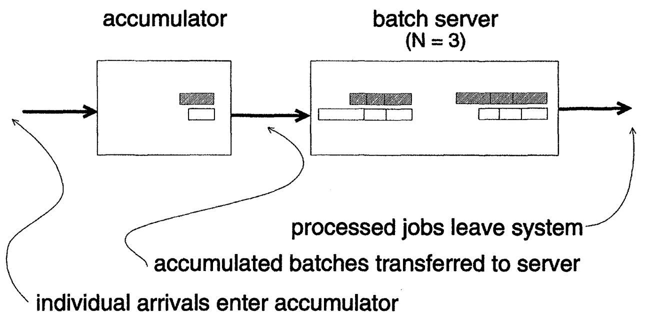
accumulator
batch server
(N = 3)
processed jobs leave system
accumulated batches transferred to server
individual arrivals enter accumulator

Figure 5. The bulk service queue under policy τ_{B_N}.

For the policy π_{B_N} in particular, a batch is formed every Nth arrival, where N may depend on ρ and will be determined later. We call every Nth arrival epoch a *batching epoch*, since these are times at which the accumulator passes batches to the batch server. To facilitate the analysis of π_{B_N}, we define two sequences of random variables, $\{\hat{V}_l: l = 1, 2, \dots\}$ and $\{\hat{T}_l: l = 1, 2, \dots\}$, where

$$\hat{V}_l \overset{\text{def}}{=} \sum_{k=1}^{N} V_{N(l-1)+k} \quad \text{and} \quad \hat{T}_l \overset{\text{def}}{=} \sum_{k=1}^{N} T_{N(l-1)+k}. \tag{11}$$

Thus, $\hat{V}_l$ is the vector of job quantities making up the lth batch and $\hat{T}_l$ is the interarrival time between the $(l-1)^{\text{st}}$ and lth batches. Like $\{V_k\}$ and $\{T_k\}$, $\{\hat{V}_l\}$ and $\{\hat{T}_l\}$ are sequences of i.i.d. random variables and are independent of each other.

Under policy π_{B_N} the batch server processes incoming batches on a FCFS basis. It determines which configuration it will use to process a given batch, l, by substituting $\hat{V}_l$ for Q_t in (2) and solving the linear program. Accordingly, we define the sequence of random variables $\{\hat{S}_l: l = 1, 2, \dots\}$ to be the processing times derived from $\{\hat{V}_l: l = 1, 2, \dots\}$. Thus, the batch server behaves as a $GI/GI/1$ queue with interarrival times $\{\hat{T}_l: l = 1, 2, \dots\}$ and service times $\{\hat{S}_l: l = 1, 2, \dots\}$.

The strategy behind using the class $\{\pi_{B_N}\}$ is to pick a large N so that every $\hat{V}_l$ is likely to fall inside the cone of B. Then the processing times of all batches are likely to achieve the lower bound, $y^{*\top}\hat{V}_l$. If the solution to (4) is not degenerate, then γ lies in the interior of the cone of the basis, and as N grows large, the probability that $\hat{V}_l$ falls outside of the cone decreases rapidly.

To develop an upper bound on $E_{\pi_B N}[W]$ we will separately bound the expected quantities of work found, in equilibrium, in the accumulator and in the batch server. First, we define $E[\hat{D}]$ to be the expected amount of work an arriving batch finds waiting at the batch server. Because the batch server acts a $GI/GI/1$ queue the following lemma, Kingman's upper bound, provides an upper bound on $E[\hat{D}]$.

Lemma 3. (*Kingman, from Wolff* 1989, *p.* 476.) *For a GI/GI/1 queue with interarrival times $\hat{T}$ and service times $\hat{S}$, such that* $E[\hat{S}]/E[\hat{T}] < 1$

$$E[\hat{D}] \leq \frac{\operatorname{var}(\hat{S} - \hat{T})}{2E[\hat{T}](1 - E[\hat{S}]/E[\hat{T}])}.$$

To use this bound we, in turn, must find bounds on the first two moments of $\hat{T}$ and $\hat{S}$. The moments of $\hat{T}$ follow directly from the definitions of $\{T_k\}$ and $\{\hat{T}_l\}$:

$$E[\hat{T}] = NE[T] = \frac{N}{\lambda}, \quad \text{and}$$

$$\operatorname{var}(\hat{T}) = N \operatorname{var}(T) = N\sigma_T^2.$$

Since $\hat{S}$ is the solution to the linear program (2), however, its moments are less straightforward to derive.

For some batches, $\hat{S}_l = y^{*\top}\hat{V}_l$ and the processing time achieves the lower bound. Recall that this happens when $\hat{V}_l$ lies within the cone of the optimal basis, B, of (4). More formally, if the solution to (3) is unique, then $\hat{S}_l = y^{*\top}\hat{V}_l$ for any batch, l in which $\hat{V}_l \in \mathcal{N}(N\gamma, N\epsilon)$.

For other batches, $\hat{V}_l \notin \mathcal{N}(N\gamma, N\epsilon)$ and the dual prices, y^*, may not apply. Still, because the elements of V are uniformly bounded by C_1, we can find a finite upper bound on $\hat{S}_l$. In this case, let $C_2 \overset{\text{def}}{=} mC_1 \max\{a_{ij}^{-1}: a_{ij} > 0, 1 \leq i \leq m, 1 \leq j \leq n\}$. Then since A is nonnegative and has rank m by assumption, for each type, i, there exists a configuration, $j(i)$, with $a_{ij(i)} > 0$. Hence,

$$\begin{aligned}\hat{S}_l &\leq \sum_{i=1}^{m} a_{ij(i)}^{-1}\hat{V}_l^i \\ &= \sum_{i=1}^{m} a_{ij(i)}^{-1} \sum_{k=1}^{N} V_{N(l-1)+k}^i \\ &= \sum_{k=1}^{N} \sum_{i=1}^{m} a_{ij(i)}^{-1} V_{N(l-1)+k}^i \\ &\leq \sum_{k=1}^{N} C_2 \\ &= NC_2.\end{aligned}$$

Let $\{E_l\} \overset{\text{def}}{=} \{\hat{V}_l \notin \mathcal{N}(N\gamma, N\epsilon)\}$. The following lemma shows that the probability that $\{E_l\}$ occurs is exponentially decreasing in N.

Lemma 4. *There exists a $\theta > 0$ such that $P\{E_l\} = O(e^{-\theta N})$.*

Proof.

$$\begin{aligned}P\{E_l\} &= P\{\hat{V}_l \notin \mathcal{N}(N\gamma, N\epsilon)\} \\ &= P\{\exists\, i \in \{1, \ldots, m\} \text{ s.t. } \hat{V}_l^i \\ &\qquad \notin [N(\gamma_i - \epsilon), N(\gamma_i + \epsilon)]\} \\ &\leq \sum_{i=1}^{m} P\{\hat{V}_l^i \notin [N(\gamma_i - \epsilon), N(\gamma_i + \epsilon)]\} \\ &\leq \sum_{i=1}^{m} P\{\hat{V}_l^i \leq N(\gamma_i - \epsilon)\} + \sum_{i=1}^{m} P\{\hat{V}_l^i \geq N(\gamma_i + \epsilon)\}.\end{aligned}$$

We recall that V is nonnegative and uniformly bounded above by C_1, so that $E[e^{\alpha V^i}] < \infty$ for all $|\alpha| < \infty$. For each type, i, we let $X \overset{\text{def}}{=} V^i$, $S_N \overset{\text{def}}{=} \hat{V}_l^i$, $a_i \overset{\text{def}}{=} \gamma_i - \epsilon$ and $b_i \overset{\text{def}}{=} \gamma_i + \epsilon$, and we apply the Chernoff bounds in Corollary 2 of Lemma 9 (see the appendix) to find $\alpha_i > 0$ and $\beta_i > 0$ such that

$$P\{E_l\} \leq \sum_{i=1}^{m} e^{-\alpha_i N} + \sum_{i=1}^{m} e^{-\beta_i N}.$$

Letting $\theta \overset{\text{def}}{=} \min\{\min_i\{\alpha_i\}, \min_i\{\beta_i\}\}$, we have found a $\theta > 0$ such that

$$\begin{aligned}P\{E_l\} &\leq \sum_{i=1}^{m} e^{-\theta N} + \sum_{i=1}^{m} e^{-\theta N} \\ &= 2me^{-\theta N},\end{aligned}$$

which completes the proof. $\square$

Rather than directly analyze the first two moments of $\hat{S}$, we define a sequence of random variables $\{\bar{S}_l: l = 1, 2, \ldots\}$ in which

$$\bar{S}_l \overset{\text{def}}{=} y^{*\top}\hat{V}_l + NC_2\mathbf{1}_{\{E_l\}}. \tag{12}$$

We note that for any sample path, $\hat{S}_l \leq \bar{S}_l$, for all l. Therefore, any upper bound on the work found in a $GI/GI/1$ queue which uses the sequences $\{\bar{S}_l: l = 1, 2, \ldots\}$ and $\{\hat{T}_l: l = 1, 2, \ldots\}$ for service and interarrival times will also be an upper bound for the batch server under policy π_{B_N}. We can readily provide an upper bound on the first two moments of $\bar{S}$ using Lemma 4:

$$\begin{aligned}E[\bar{S}] &= E[y^{*\top}\hat{V}_l + NC_2\mathbf{1}_{\{E_l\}}] \\ &= y^{*\top}E[\hat{V}_l] + NC_2E[\mathbf{1}_{\{E_l\}}] \\ &= Ny^{*\top}\gamma + NC_2P\{E_l\} \\ &= Ny^{*\top}\gamma + O(N)P\{E_l\},\end{aligned}$$

and

$$\begin{aligned}\operatorname{var}(\bar{S}) &= \operatorname{var}(y^{*\top}\hat{V}_l + NC_2\mathbf{1}_{\{E_l\}}) \\ &= \operatorname{var}(y^{*\top}\hat{V}_l) + \operatorname{var}(NC_2\mathbf{1}_{\{E_l\}}) \\ &\quad + 2\operatorname{cov}(y^{*\top}\hat{V}_l, NC_2\mathbf{1}_{\{E_l\}}).\end{aligned}$$

For the first term we have $\operatorname{var}(y^{*\top}\hat{V}_l) = Ny^{*\top}\Gamma y^*$, and for the second we have

$$\begin{aligned}\operatorname{var}(NC_2\mathbf{1}_{\{E_l\}}) &= N^2C_2^2 \operatorname{var}(\mathbf{1}_{\{E_l\}}) \\ &= N^2C_2^2(E[\mathbf{1}_{\{E_l\}}^2] - P\{E_l\}^2) \\ &\leq N^2C_2^2E[\mathbf{1}_{\{E_l\}}^2] \\ &= N^2C_2^2P\{E_l\}.\end{aligned}$$

Finally, we find that

$$\begin{aligned}\operatorname{cov}(y^{*\top}\hat{V}_l, NC_2\mathbf{1}_{\{E_l\}}) &= NC_2 \operatorname{cov}(y^{*\top}\hat{V}_l, \mathbf{1}_{\{E_l\}}) \\ &= NC_2(E[y^{*\top}\hat{V}_l\mathbf{1}_{\{E_l\}}] \\ &\qquad - Ny^{*\top}\gamma P\{E_l\}) \\ &= NC_2E[y^{*\top}\hat{V}_l\mathbf{1}_{\{E_l\}}] \\ &\qquad - N^2C_2y^{*\top}\gamma P\{E_l\}.\end{aligned}$$

Then

$$E[y^{*\top}\hat{V}_l\mathbf{1}_{\{E_l\}}] = E[y^{*\top}\hat{V}_l\mathbf{1}_{\{E_l\}}|\{E_l\}]P\{E_l\} + E[y^{*\top}\hat{V}_l\mathbf{1}_{\{E_l\}}|\{\bar{E}_l\}]P\{\bar{E}_l\} \leq Ny^{*\top}\mathbf{1}C_1P\{E_l\},$$

where $\{\bar{E}_l\}$ is the complement of $\{E_l\}$. This implies that

$$\operatorname{cov}(y^{*\top}\hat{V}_l, NC_2\mathbf{1}_{\{E_l\}}) \leq NC_2Ny^{*\top}\mathbf{1}C_1P\{E_l\}$$

Therefore, we have

$$\begin{aligned}\operatorname{var}(\bar{S}) &\leq Ny^{*\top}\Gamma y^* + N^2C_2^2P\{E_l\} \\ &\quad + 2N^2C_1C_2y^{*\top}\mathbf{1}P\{E_l\} \\ &= Ny^{*\top}\Gamma y^* + O(N^2)P\{E_l\}.\end{aligned}$$

With the first two moments of $\hat{T}$ and $\bar{S}$ at our disposal, we are ready to prove an upper bound. We present the first part of the proof as a lemma.

Lemma 5. *Suppose y^* is unique. If $\rho < 1$ then there exists an integer N_0 such that for all $N \geq N_0$,*

$$E[\bar{S}] < \frac{1}{\lambda},$$

and

$$E[\hat{D}] \leq \frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^* + O(Ne^{-\theta N}))}{2(1-\rho-\lambda O(e^{-\theta N}))}.$$

Proof. The fact that $E[\bar{S}] < 1/\lambda$ for a sufficiently large N_0 follows from (12) and Lemma 4. Then from Lemma 3, we know that for all $N \geq N_0$

$$E[\hat{D}] \leq \frac{\operatorname{var}(\hat{S}-\hat{T})}{2E[\hat{T}](1-E[\hat{S}]/E[\hat{T}])}.$$

Furthermore, for any sample path $\hat{S}_l \leq \bar{S}_l$, $\{l = 1, 2, \ldots\}$. Substituting $\{\bar{S}_l: l = 1, 2, \ldots\}$ for $\{\hat{S}_l: l = 1, 2, \ldots\}$ we see that

$$\begin{aligned}E[\hat{D}] &\leq \frac{\operatorname{var}(\bar{S}-\hat{T})}{2E[\hat{T}](1-E[\bar{S}]/E[\hat{T}])} \\ &= \frac{\operatorname{var}(\bar{S})+\operatorname{var}(\hat{T})}{2E[\hat{T}](1-E[\bar{S}]/E[\hat{T}])} \\ &\leq \frac{Ny^{*\top}\Gamma y^* + O(N^2)P\{E\} + N\sigma_T^2}{2\left(\frac{N}{\lambda}\right)\left(1-\frac{Ny^{*\top}\gamma + O(N)P\{E\}}{N/\lambda}\right)} \\ &= \frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^* + O(N)P\{E\})}{2(1-\rho-\lambda P\{E\})} \\ &= \frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^* + O(Ne^{-\theta N}))}{2(1-\rho-\lambda O(e^{-\theta N}))}. \quad \square\end{aligned}$$

While the lemma offers a bound on the expected work that an arriving batch finds at the batch server, we seek a bound on the total amount of work found in both the accumulator and the batch server at an arbitrary time. To this end, we extend the previous lemma:

Theorem 3. *Suppose y^* is unique. If $\rho < 1$ then there exists a policy π_{B_N} that is stable and for which*

$$E_{\pi_{B_N}}[W] \leq \frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^* + O(Ne^{-\theta N}))}{2(1-\rho-\lambda O(e^{-\theta N}))} + O(N).$$

Proof. The proof proceeds in three steps. First, we bound the work found in the accumulator at an arbitrary point in time. Then we find a bound on the work found waiting at the batch server at an arbitrary point in time in terms of $E[\hat{D}]$. To finish, we put the two bounds together.

Step 1. Under π_{B_N} the quantity of jobs in the accumulator starts at zero at the beginning of each batching cycle, increases throughout the cycle, and reaches its peak as the Nth job of the batch arrives and the batch is dispatched to the batch server. Similarly, the amount of work in the accumulator peaks at the end of any cycle, so for a given cycle, l, $\bar{S}_l$ provides a uniform upper bound on the work found in the accumulator during the cycle. Since the sequences $\{T_k: k = 1, 2, \ldots\}$ and $\{V_k: k = 1, 2, \ldots\}$ are independent and N is constant, it follows that $E[\bar{S}]$ is an upper bound on the expected amount of work in the accumulator at an arbitrary time.

Step 2. For a stable, work-conserving $GI/GI/1$ queue with service times $\{S\}$ and interarrival times $\{T\}$ it is well known (see Wolff 1989, p. 279) that the following relationship holds between the time average of system work, $E[W]$, and expected delay upon arrival $E[D]$:

$$E[W] = \rho E[D] + \frac{E[S^2]}{2E[T]}.$$

For our problem this immediately leads to a simple upper bound on work found waiting at the batch server at an arbitrary time:

$$E[\hat{D}] + \frac{\lambda E[\bar{S}^2]}{2N}.$$

Step 3. Putting together the results from first two steps we have

$$E_{\pi_{B_N}}[W] \leq E[\hat{D}] + E[\bar{S}] + \frac{\lambda E[\bar{S}^2]}{2N}. \tag{13}$$

Lemma 5 implies that for any $\rho < 1$ this quantity is bounded for large integers N and the queue is stable. Furthermore, NC_2 is an upper bound on $\bar{S}$, which gives us

$$E_{\pi_{B_N}}[W] \leq \frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^* + O(Ne^{-\theta N}))}{2(1-\rho-\lambda O(e^{-\theta N}))} + O(N). \quad \square$$

Theorem 3 establishes the sufficiency of $\rho < 1$ for stability. As before, the condition $\rho < 1$ can be expressed in an equivalent vector form that better illustrates its dependency on the product mix, γ, and the system capabilities, A; namely, a stable policy exists if the system

$$\begin{aligned}\lambda y^\top\gamma &< 1, \\ y^\top A &\leq 1, \\ y &\geq 0,\end{aligned}$$

is solvable in y.

We next relate the expected work of the heuristic to the lower bound of Theorem 2 under heavy traffic conditions. First, note that as the batch size N increases without bound, the expected work found by a batch arriving to the batch server approaches the lower bound. The expected work found in the accumulator, however, increases linearly with N. From Theorem 2 we recall that in heavy traffic the lower bound on W^* is $O(1/(1-\rho))$. Therefore, if we let $N \to \infty$ *sublinearly* with $1/(1-\rho)$, the expected work in the system under the policies $\{\pi_{B_N}\}$ will converge to optimal levels. Theorem 4 formalizes this idea.

Theorem 4. *Suppose y^* is unique. Let $N \overset{\text{def}}{=} \lceil (1-\rho)^{-b} \rceil$ for some arbitrary, fixed $b \in (0, 1)$. If we scale T as in Lemma 6 so that $\rho \uparrow 1$ by letting $\lambda \uparrow 1/y^{*\top}\gamma$, we find that*

$$\lim_{\rho \uparrow 1} \frac{E_{\pi_{B_N}}[W]}{W^*} = 1. \tag{14}$$

Proof. From Theorem 3,

$$\begin{aligned}(1-\rho)E_{\pi_{B_N}}[W] &\le (1-\rho) \\ &\quad \cdot \left[\frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^* + O(Ne^{-\theta N}))}{2(1-\rho-\lambda O(e^{-\theta N}))} + O(N) \right] \\ &= \frac{1-\rho}{1-\rho-\lambda O(e^{-\theta N})} \\ &\quad \times \frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^* + O(Ne^{-\theta N}))}{2} \\ &\quad + (1-\rho)O(N).\end{aligned}$$

For $N \overset{\text{def}}{=} \lceil (1-\rho)^{-b} \rceil$ for any fixed $b \in (0, 1)$, $\lim_{\rho \uparrow 1}(1-\rho)O(N) = 0$, and

$$\lim_{\rho \uparrow 1} \frac{1-\rho}{1-\rho-\lambda O(e^{-\theta N})} = \lim_{\rho \uparrow 1} \frac{1}{1 - \dfrac{\lambda O(e^{-\theta N})}{1-\rho}} = 1.$$

Also note that whenever $N \to \infty$ as $\rho \uparrow 1$, $\lim_{\rho \uparrow 1} O(Ne^{-\theta N}) = 0$. Therefore,

$$\lim_{\rho \uparrow 1}(1-\rho)E_{\pi_{B_N}}[W] \le \frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^*)}{2}.$$

Dividing this limit by that of Theorem 2 completes the proof. □

5. OTHER HEURISTICS

The batching policies in the previous section are designed primarily to provide analytically tractable upper bounds. As our simulation results below show, in moderate traffic the batching policies do not perform well. In this section we examine two alternative policies motivated by our analysis, which, although not provably good, are more appealing on a practical level. In simulation experiments they also perform well relative to the lower bound of Theorem 2.

Our definition of work, the minimum time required by the service facility to clear the system backlog, suggests a greedy policy (GREEDY) as a natural choice. A greedy policy would always seek to serve the current system backlog as quickly as possible, without regard for future arrivals. At arrival epochs, t_k, GREEDY substitutes Q_{t_k} in the right-hand side of (2) and solves the linear program. GREEDY then uses the result of (2) to determine a direct path back to the origin that requires a minimum of processing time (see left side of Figure 6). We note that GREEDY does not make use of the optimality of processing using B, nor of any information concerning the distributions of V and T.

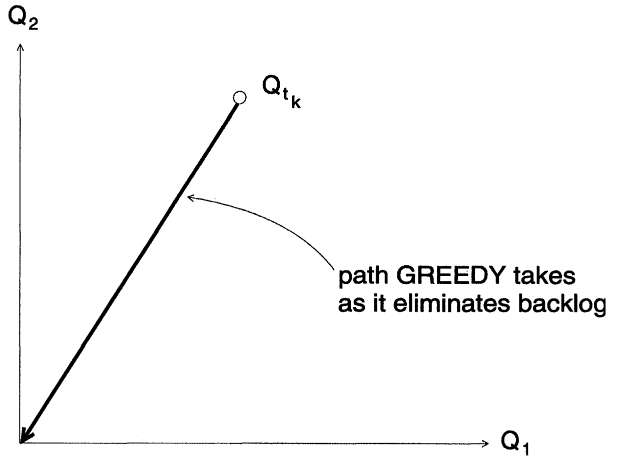

Figure 6. Example sample path of GREEDY policy.

The second policy, CENTER, uses the insights from the batching policies but applies them more dynamically. Recall, there were two objectives of a good policy suggested by our earlier analysis: (1) the policy should process using the columns of B as much as possible; (2) it should try to maintain the backlog "centered" within the cone of B at all times. Like GREEDY, CENTER reevaluates which processing configurations the service facility should be using at each arrival epoch, t_k. When the system backlog at an arrival epoch, Q_{t_k}, is in the cone of B, CENTER processes the backlog by using the columns of B to first move to a center line, C, and then move down C to the origin (see right side of Figure 7). When Q_{t_k} is not in the cone of B, CENTER processes the backlog by first processing as much work as possible using columns of B, and then using whatever remaining columns of A will return the backlog to the origin as quickly as possible (see Figure 7).

Since the CENTER policy aims to keep the backlog within the cone of B as much as possible, we construct the centering ray, C, to reduce the probability that, given the backlog at some t_k is near to C, the arrival that occurs at t_k forces the backlog out of the cone. In particular, we note that if the ray of γ is near a boundary of the cone of B, then by placing C near the *opposite* boundary, we can reduce the probability that this event occurs (see Figure 8).

For a detailed description of the GREEDY and CENTER policies, we refer the reader to the appendix.

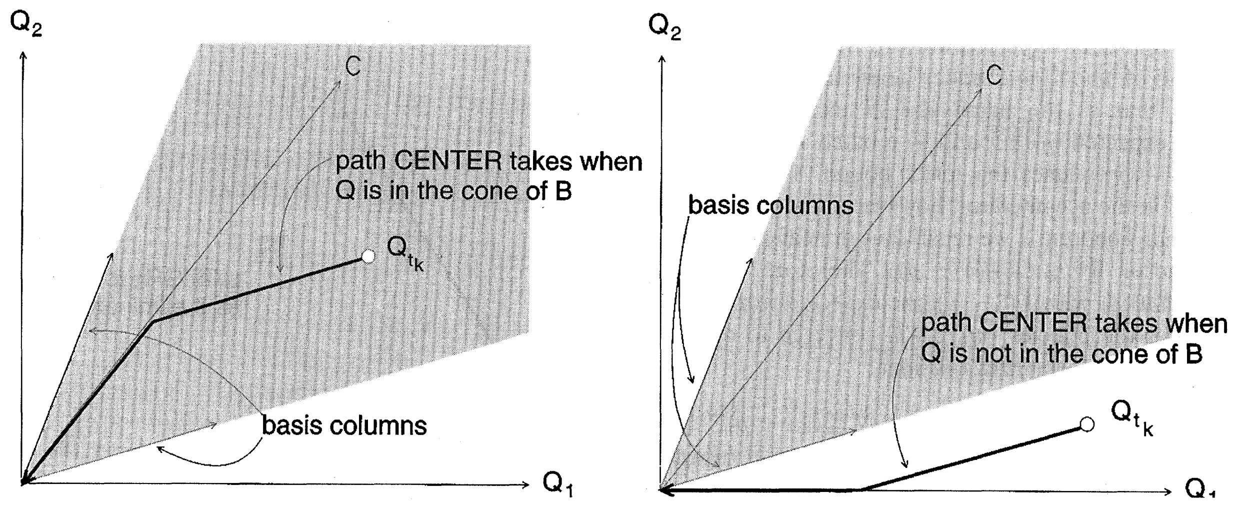

Figure 7. Example sample paths of CENTER policy.

6. NUMERICAL ANALYSIS

This section reports the results of four sets of simulation experiments that test the performance of the GREEDY and CENTER policies, as well as that of the batching policies, which we will call BATCH. The simulations sample the system *work* found upon arrival, $\{W^{\pi}_{t_k^-}: k = 1, 2, \ldots\}$, under the three heuristics, as well as the analogous quantities for the lower bound process, $\{\underline{W}_{t_k^-}: k = 1, 2, \ldots\}$, which in this section we will call LOWER.

In each simulation run, we use the method of "batch means" (see Law and Kelton 1982, p. 295–297) with batches of size

$$M \approx 10\,\frac{\lambda^2\sigma_T^2 + y^{*\,\top}\Gamma y^*/(y^{*\,\top}\gamma)^2}{(1-\rho)^2}. \tag{15}$$

(See Whitt 1989, p. 1355–1357.) For the lower bound and each of the three policies, the sample points of the work process that fall within each batch are averaged. The simulation run terminates when the 95% confidence intervals for the estimates of all four population means (the average of the averages) are less than or equal to ±10% of the estimate of the population means themselves—or after the simulation has run for twenty-four hours (on an Intel 486 DX2/66-based microcomputer), whichever comes first.

In all simulation runs, we use i.i.d. exponential interarrival times. Then by PASTA (see Wolff 1989, p. 293–297) we may interpret the arrival averages calculated by the simulation to be unbiased estimates for the analogous time averages, $E_\pi[W]$.

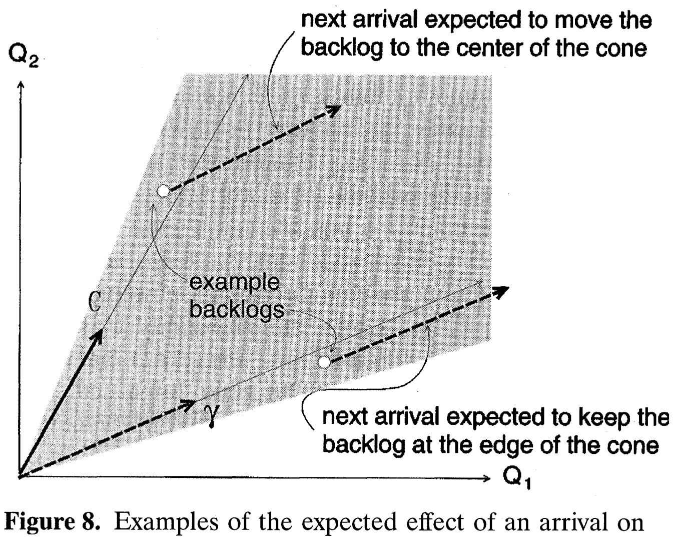
next arrival expected to move the backlog to the center of the cone
Q2
example backlogs
C
γ
next arrival expected to keep the backlog at the edge of the cone
Q1

Figure 8. Examples of the expected effect of an arrival on the backlog.

6.1. Simulation Input Data

All four sets of simulations use the same model of a service facility described in Section 2. We recall that the A matrix from this example is

$$\begin{bmatrix} 4 & 4 & 0 & 2 \\ 0 & 3 & 5 & 5 \end{bmatrix}.$$

Within each of the four sets, we vary only the mean of the interarrival time distribution, T, to achieve ρs of 0.8, 0.9, 0.95, and 0.99.

In the first two sets of simulations, called Examples 1 and 2, we let $\gamma^\top = (10, 10)$. In these cases, the solution to (3) is $y^* = (1/7, 1/7)$, station $S1$ is the bottleneck, and both job types contribute equally to the congestion at $S1$. In addition, the basis for the optimal solution to (4) is

$$\begin{bmatrix} 4 & 2 \\ 3 & 5 \end{bmatrix}.$$

In the remaining two sets of experiments, called Examples 3 and 4, we let $\gamma^\top = (16, 6)$. In these cases, station $S2$ is the bottleneck, and only job type 1 contributes to the congestion. In these two cases, the solution to (3) is $y^* = (1/4, 0)$ and the optimal basis for (4) is

$$\begin{bmatrix} 4 & 4 \\ 0 & 3 \end{bmatrix}.$$

Examples 1 and 3 are distinguished from the Examples 2 and 4 by the correlation between the arrival quantities of

Table I
Data for V in Examples 1 through 4

Example	(γ_1, γ_2)	(CV_1, CV_2)	CC
1	(10, 10)	(0.86, 0.86)	0.34
2	(10, 10)	(1.22, 1.22)	−0.17
3	(16, 6)	(0.94, 0.85)	0.26
4	(16, 6)	(1.26, 2.27)	−0.27

the two job types. In the former, the correlation is positive, and in the latter it is negative. Table I describes the first two moments—means, coefficients of variation (CV), and correlation coefficient (CC)—of V in each of the four cases:

As ρ increases from 0.8 to 0.99, we scale the batch size, N, in the BATCH policy so that $N \approx 2.5(1 - \rho)^{-0.75}$, where the parameters 2.5 and 0.75 were chosen based on a series of shorter simulation runs. Specifically, at each utilization we performed several runs in which we varied N in order to minimize the expected backlogs of BATCH. After finding "good" Ns for utilizations of 0.8, 0.9, 0.95, and 0.99, we then chose an a and b so that they would loosely fit $N = a(1 - \rho)^{-b}$ for these "good" Ns.

6.2. Simulation Results

The simulation results, presented in Table II, reveal a number of interesting phenomena. First of all, the CENTER policy performs quite well. For example, in simulations with $\rho = 0.99$, the expected system work under CENTER is between 0% and 1% above the LOWER bound. While in one case the GREEDY policy performed nearly optimally, its results were generally inferior to those of CENTER. Furthermore, whenever CENTER's results rose above the lower bound, the performance of GREEDY deteriorated even more sharply. As we noted previously, the BATCH policy does not fare well; in all cases its backlog remains at least 60% above that of the lower bound.

Note that the heuristics fared *worse* with respect to the lower bound when the correlations were negative (Examples 2 and 4) than when the correlations were positive (Examples 1 and 3). Indeed, it appears from our observations of some sample paths that the positive correlation actually *helps* the backlog stay inside the cone of the basis, B, resulting in better performance relative to the lower bound (see Figure 8). Further evidence of this phenomenon is that the relative performance of GREEDY is closer to that of CENTER when the correlation is positive than when it is negative. Again, when the correlation is positive, the backlog also stays in the cone of B with GREEDY, and there is less of a difference between the performance of the two heuristics. When the correlation is negative, however, the CENTER policy keeps the backlog inside the cone of B more effectively than GREEDY, and CENTER's relative performance deteriorates more slowly than GREEDY's. However, the examples with negative correlations also have higher coefficients of variation for the individual jobs, and this factor may be contributing to the effect as well.

We point out that these observations concern the heuristics' performance *relative* to the lower bound. It is also important to note that since $y^* > 0$, negative correlation among the elements of the arrival vector tends to reduce the expected backlogs of all the heuristics through the term $y^{*\top}\Gamma y^*$. Thus, negative correlation tends to improve the *absolute* performance of the heuristics, even though their performance relative to the lower bound may suffer.

Table II
Simulation Results for the Lower Bound and Three Policies

				CENTER		GREEDY		BATCH	
Example	ρ	$M/G/1$[1]	LOWER	interval[2]	premium[3]	interval[2]	premium[3]	interval[2]	premium[3]
1	0.80	8.57	8.54 (±0.85)	8.96 (±0.85)	4.9%	9.53 (±0.90)	11.6%	26.64 (±1.24)	211.9%
	0.90	19.29	19.74 (±1.97)	20.12 (±1.96)	1.9%	21.36 (±2.05)	8.2%	52.89 (±2.44)	167.9%
	0.95	40.72	41.75 (±4.12)	42.06 (±4.14)	0.7%	44.19 (±4.24)	5.8%	102.30 (±4.76)	145.0%
	0.99[4]	212.16	203.62 (±24.27)	203.83 (±24.26)	0.1%	207.01 (±24.28)	1.7%	412.32 (±25.03)	102.5%
2	0.80	9.29	9.43 (±0.94)	12.13 (±1.04)	28.6%	14.01 (±1.26)	48.6%	31.98 (±2.22)	239.1%
	0.90	20.89	22.86 (±2.28)	26.04 (±2.30)	13.9%	31.36 (±2.77)	37.2%	70.18 (±4.87)	207.0%
	0.95	44.11	43.44 (±4.28)	46.39 (±4.25)	6.8%	56.71 (±4.70)	30.5%	132.81 (±9.98)	205.7%
	0.99[4]	229.84	227.77 (±29.74)	230.13 (±29.77)	1.0%	254.57 (±29.66)	11.8%	560.10 (±57.42)	145.9%
3	0.80	15.06	14.47 (±1.44)	14.52 (±1.44)	0.3%	14.58 (±1.44)	0.8%	36.38 (±1.70)	151.4%
	0.90	33.89	32.25 (±3.20)	32.29 (±3.20)	0.1%	32.41 (±3.20)	0.5%	76.10 (±3.49)	136.0%
	0.95	71.56	69.81 (±6.85)	69.82 (±6.85)	0.0%	69.94 (±6.85)	0.2%	149.83 (±7.24)	114.6%
	0.99	372.83	346.99 (±34.68)	346.99 (±34.68)	0.0%	347.15 (±34.67)	0.0%	634.46 (±34.83)	82.8%
4	0.80	20.69	19.94 (±1.98)	20.89 (±1.96)	4.8%	21.48 (±2.00)	7.7%	44.59 (±2.61)	123.6%
	0.90	46.55	49.86 (±4.94)	50.67 (±4.91)	1.6%	52.20 (±4.94)	4.7%	99.86 (±6.09)	100.3%
	0.95	98.28	90.71 (±8.99)	91.32 (±8.95)	0.7%	94.34 (±8.98)	4.0%	182.62 (±11.86)	101.3%
	0.99	512.07	502.57 (±49.94)	502.73 (±49.93)	0.0%	507.91 (±49.99)	1.1%	802.16 (±53.43)	59.6%

[1]Analytical expectation of steady state system work for an $M/G/1$ queue with the same arrival and service statistics as LOWER's.
[2]95% confidence interval for expected system work.
[3]Expected system work's percentage premium over LOWER.
[4]Simulation run terminated after twenty-four hours, before the confidence intervals reached ±10% of the estimated mean.

Finally, the performance of all of the policies improves relative to the lower bound as system utilization increases. This improvement is also likely due to an increase in the probability that the backlog is in the cone of B as the number of arrivals waiting to be processed grows and is consistent with our theoretical analysis.

In addition to the results of the simulation experiments, Table II presents the analytically derived expectation for the steady state work in an $M/G/1$ queue with interarrival times T and service times $y^{*\top}V$ (labeled "$M/G/1$"). In this case we have

$$E[\underline{W}] = \frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^*)}{2(1-\rho)},$$

not only in heavy traffic, but for all $\rho < 1$ (see Wolff 1989, p. 280).

With exponential interarrival times, the lower bound process becomes just such an $M/G/1$ queue. Therefore, as a check on the validity of the simulation results for the lower bound, we compare LOWER to the corresponding analytical expectation in $M/G/1$. In fact, in all cases the $M/G/1$ expectation falls within the 95% confidence interval of the simulated mean for LOWER.

7. EXTENSIONS

We next present four simple extensions.

7.1. Processing Costs

Consider the case in which the operation of the service facility under configuration j incurs costs at rate c_j. Suppose there is a fixed *operating budget*, which we model as a limit, C, on the rate the facility can spend per unit of time. We can modify (2) to include this budget constraint as follows:

$$\begin{aligned} W_t &= \min \sum_{j=1}^{n} x_j, \qquad (16)\\ &\text{s.t.}\\ Ax &\geq Q_t,\\ c^{\top}x &\leq C\sum_{j=1}^{n} x_j,\\ x &\geq \mathbf{0}. \end{aligned}$$

Similarly, (3) becomes

$$\begin{aligned} &\max \gamma^{\top}y, \qquad (17)\\ &\text{s.t.}\\ &[y^{\top} u]\begin{bmatrix} A \\ c - \mathbf{1}C \end{bmatrix} \leq \mathbf{1},\\ &y \geq \mathbf{0},\\ &u \leq 0. \end{aligned}$$

For any fixed C we can use y^* and the optimal basis to the dual of (17), B, as before. The resulting bounds are then valid for policies which generate costs at fixed rate C. As we vary C, we trace an *efficient frontier* for which we can view the tradeoff between the rate at which the service facility incurs expense and long run expected system work when operating optimally under the expense constraint.

7.2. Backlog Costs

Suppose the backlog at time t incurs cost at the rate $C_t \overset{\text{def}}{=} f(Q_t)$, where $f(\cdot)$ is nondecreasing. Then it is not difficult to show that, given a lower bound on the system work at time t, $\underline{W}_t$,

$$\begin{aligned} \underline{C}_t &= \min f(Q), \qquad (18)\\ &\text{s.t.}\\ y^{*\top}Q &= \underline{W}_t,\\ Q &\geq \mathbf{0}, \end{aligned}$$

is a lower bound on C_t, and, in turn, the *process* $\{\underline{C}_t: t \geq 0\}$ is a lower bound on $\{C_t: t \geq 0\}$. Similarly, the accumulated discounted cost ($\int_0^t \underline{C}_s e^{-\alpha s}\,ds$) and the average cost ($t^{-1}\int_0^t \underline{C}_s\,ds$) generated by $\{\underline{W}_t: t \geq 0\}$ and (18) are lower bounds on the corresponding costs for the original system $\{Q_t: t \geq 0\}$. We are currently using this lower bound to investigate the performance of cost-minimizing heuristics.

We note that (18) and its associated average cost measure are closely related to the *workload formulation* and solution presented in Wein (1992), Equations (24)–(27) and (31)–(32). (See also Harrison 1988 for the development of this approach.)

7.3. Changes to the Service Facility

The intimate connection of the heavy traffic limit

$$\frac{\lambda(\sigma_T^2 + y^{*\top}\Gamma y^*)}{2(1-\rho)},$$

to the linear program (4) allows one to apply standard linear programming sensitivity analysis to evaluate changes in the system. For example, the addition of a new processing configuration would have *no* effect on the expected backlog if the new column of A were not a part of the optimal basis in the solution to (3). New configurations can therefore be "priced-out" to decide how they would effect the optimal backlog.

7.4. Adaptive Policies

Adaptive versions of the policies can be implemented in cases where the demand statistics are not known. Indeed, very little data are needed to execute the three heuristics presented. GREEDY requires only the A matrix and the sample path of arrivals. In addition to these, CENTER requires only γ to determine the basis, B, and the center line, C; and BATCH requires only γ and λ to determine y^*, ρ, and, in turn, the batch size, N.

To apply these policies adaptively, we might begin with estimates of the two parameters, γ_0 and λ_0, and update our estimates as the sample path evolves. If the arrival process is truly stationary, then after the kth arrival we could update our estimates to be

$$\lambda_k \overset{\text{def}}{=} \frac{k+1}{\dfrac{1}{\lambda_0} + \sum_{j=1}^{k} T_j},$$

$$\gamma_k^i \overset{\text{def}}{=} \frac{\gamma_0^i + \sum_{j=1}^{k} V_k^i}{k+1},$$

and know that, as $k \to \infty$, the behavior of the adaptive versions of the policies converge to the behavior of the policies when γ and λ are known.

To implement CENTER in this situation, we can recalculate C after each arrival and use the new C for next arrival. Similarly, we can re-solve (3) and, if the basis of the optimal solution changes, use the new basis, B. For BATCH, we can recalculate N after each arrival. Then, if the number of arrivals in the accumulator equals or exceeds N, a new batch can be formed and passed to the batch processor.

If the arrival process is not stationary, we can use moving averages, exponential smoothing, or regression techniques to update the estimates of γ and λ. Furthermore, if the arrival process exhibits seasonal fluctuations, a "moving" estimate of γ would allow us to recompute a new B and C that should move appropriately as demand characteristics vary.

8. SUMMARY AND CONCLUSIONS

Our queueing model has a number of appealing features. The definition of the arrival process is quite general and allows for correlation among the arriving quantities of the different job types. The representation of the service facility as a matrix of processing configurations provides significant flexibility in how one models service facilities. The definition of system work is a natural one, and the use of a linear program in determining the work gives a great deal of information on the sources of system congestion. More specifically, the linear program identifies an optimal basis for the service facility to use, as well as a set of dual prices that provides information on the work content of the various job types.

We make use of these insights to construct the batching policy, which performs asymptotically optimally in heavy traffic, and the CENTER policy, which performs nearly optimally in moderate traffic. Moreover, we are able to obtain a closed form characterization of the optimal work in heavy traffic.

As we noted in the introduction, however, our model of the service facility suppresses much of the detail concerning how work is actually processed within the facility. Indeed, there may be restrictions on service which the "processor sharing" assumption does not address. In many systems jobs may require non-preemptive service: service which, once it commences, is not interrupted until it completes. In others, such as logistics systems (in which the configurations correspond to various delivery routes), the server may only use integral multiples of configurations to feasible serve the system backlog. Systems for which these restrictions apply are the subject of two forthcoming papers (Gans and van Ryzin 1996a and 1996b).

APPENDIX

Lemmas

Lemma 6. (*Asmussen, cited in Wolff* 1989, *p.* 518.) *Consider a sequence of stable GI/GI/1 queues in which the nth queue in the sequence has all random variables indexed by n. In particular, the nth queue has interarrival times T_n, service times Z_n and waiting times (customer delay in the system) D_n. Suppose as $n \to \infty$, $T_n \xrightarrow{\mathcal{D}} T$ and $Z_n \xrightarrow{\mathcal{D}} Z$. If in addition, as $n \to \infty$,*

(i) $\rho_n \to \rho = 1$;
(ii) $E[Z_n^2] \to E[Z^2]$ *and* $E[T_n^2] \to E[T^2]$;
(iii) $\text{var}(Z - T) > 0$;

then as $n \to \infty$,

$$\frac{2(1-\rho_n)D_n}{\lambda \ \text{var}(Z_n - T_n)} \xrightarrow{\mathcal{D}} \exp(1),$$

an exponential random variable with mean 1.

Lemma 7. (*Wolff* 1989, *p.* 291.) *Given a G/G/1 queue with interarrival times T, service times Z, let $E[\underline{W}]$ and $E[\underline{D}]$ be defined as in* (9) *and* (10). *If $\lambda = 1/E[T]$ and $E[\underline{D}]$ exist and are finite, then $E[\underline{W}]$ exists, is finite, and*

$$E[\underline{W}] = E[Z\underline{D}] + \frac{\lambda E[Z^2]}{2}.$$

Lemma 8. (*Chernoff, cited in Coffman and Lueker* 1991, *p.* 16.) *Suppose X is a random variable and that $\mu = E[X]$ exists and is finite. Let S_n be the sum of n independent samples of X. Then, defining*

$$u(a) \overset{\text{def}}{=} \inf_{\alpha} E[e^{\alpha(X-a)}],$$

we have

(i) $a \geq \mu \Rightarrow P\{S_n \geq na\} \leq u(a)^n$;
(ii) $a \leq \mu \Rightarrow P\{S_n \leq na\} \leq u(a)^n$.

Lemma 9. (*Durrett* 1991, *Lemma* 9.4, *p.* 59.) *Suppose X is a random variable and there exists some $\alpha > 0$ such that $E[e^{\alpha X}]$ is finite. Let $\mu = E[X]$. Then for $a > \mu$ and small α*

$$E[e^{\alpha(X-a)}] < 1.$$

Corollary 1. *If X is a random variable, $\mu = E[X]$, and there exists some $\alpha < 0$ such that $E[e^{\alpha X}]$ is finite, then for $a < \mu$ and small $\alpha < 0$*

$$E[e^{\alpha(X-a)}] < 1.$$

Proof. Let $Y \overset{\text{def}}{=} -X$, $b \overset{\text{def}}{=} -a$, $\beta \overset{\text{def}}{=} -\alpha$, and apply Lemma 9. □

Corollary 2. *Let X be a random variable with $0 \leq X \leq C$, a.s., for some $C < \infty$. Suppose S_n is the sum of n i.i.d. samples of X. Then*

(i) *for each $a > E[X]$ there exists a $\theta > 0$ such that* $P\{S_n \geq na\} = O(e^{-\theta n})$;

(ii) *for each* $a < E[X]$ *there exists a* $\theta > 0$ *such that* $P\{S_n \leqslant na\} = O(e^{-\theta n})$.

Proof. Since $0 \leqslant X < \infty$, $|\mu \overset{\text{def}}{=} E[X]| < \infty$ and $E[e^{\alpha X}] < \infty$ for all $|\alpha| < \infty$. From Lemma 8, we then know that $a > \mu$ implies that $P\{S_n \geqslant na\} \leqslant u(a)^n$ and $a < \mu$ implies that $P\{S_n \leqslant na\} \leqslant u(a)^n$. In turn, from Lemma 9 and its corollary, we know that in both cases $u(a) < 1$. But $u(a) < 1$ implies that there exists a $\theta > 0$ such that $u(a) = e^{-\theta}$. □

Detailed Descriptions of Heuristics

The GREEDY Policy. At each arrival epoch, t_k: (1) substitute Q_{t_k} in the right-hand side of (2) and solve; call the optimal solution to this linear program x; (2) starting at t_k, use a convex combination of configurations in which $U_t^j \overset{\text{def}}{=} x_j/\mathbf{1}^\top x$; (3) process the backlog using U_t until $(t_k + \min\{\mathbf{1}^\top x, T_{k+1}\})$, the earlier of the time of next arrival epoch and the time that the queue is cleared.

The CENTER Policy. Before the simulation begins, define a "center ray", $C \in \mathbf{R}^m$, in the interior of the cone of B: (1) let $d \in \mathbf{R}^m$ equal $B^{-1}\gamma$; (2) let $e \in \mathbf{R}^m$ be the "inverse" of d, where for each element, i, $e_i = 1/d_i$; then (3) let $C = Be$.

Then at each arrival epoch, t_k:

• if Q_{t_k} is in the cone of B, then (A) solve the linear program:

$$W_C = \min \sum_{j=1}^{m} x_j,$$
$$\text{s.t.}$$
$$Bx + \alpha C = Q_{t_k},$$
$$x \geqslant \mathbf{0}, \tag{19}$$

and call the optimal solution to (19) $(x, \alpha) \in \mathbf{R}^{m+1}$; (B) starting at t_k, process the backlog using a convex combination of configurations in which $U_t^j \overset{\text{def}}{=} x_j/\mathbf{1}^\top x$; (C) process the backlog using U_t until time $(t_k + \min\{W_C, T_{k+1}\})$, the earlier of the time of the next arrival and the time the backlog equals αC; (D) if, in turn, $(W_C < T_{k+1})$, then

(a) let $x \in \mathbf{R}^n$ equal $B^{-1}C$; (b) process the backlog using a convex combination of configurations in which $U_t^j \overset{\text{def}}{=} x_j/\mathbf{1}^\top x$; (c) process the remaining backlog using U_t until the next arrival epoch or the backlog is eliminated, whichever time comes first.

• if, however, Q_t is *not* in the cone of B, then: (A) solve the linear program:

$$W_{out} = \max \sum_{j=1}^{m} x_j$$
$$\text{s.t.}$$
$$Bx \leqslant Q_{t_k}$$
$$x \geqslant \mathbf{0}; \tag{20}$$

(if the optimal solution is not unique, choose an optimal solution that maximizes $y^{*\top}Bx$) and call the optimal solution to (20) $x \in \mathbf{R}^m$; (B) starting at t_k, process the backlog using a convex combination of configurations in which $U_t^j \overset{\text{def}}{=} x_j/\mathbf{1}^\top x$; (C) process the backlog using U_t until, $(t_k + \min\{W_{out}, T_{k+1}\})$, the earlier of the next arrival epoch and the time that the server can no longer use any column of B to process the backlog; (D) if, in turn, $(W_{out} < T_{k+1})$, then call Q_{out} the remaining backlog at time $(t_k + W_{out})$:

(a) put Q_{out} in the right-hand side of (2), solve, and call the optimal solution of the linear program $x \in \mathbf{R}^n$; (b) process the remaining backlog, using a convex combination of configurations in which $U_t^j \overset{\text{def}}{=} x_j/\mathbf{1}^\top x$; (c) process the backlog using U_t until either the backlog is exhausted or the next arrival occurs, whichever time comes first.

ACKNOWLEDGMENT

We thank Karl Sigman for his help in correcting an error in Step 2 of Theorem 3, as well as two anonymous referees and the associate editor for many helpful suggestions on an earlier draft of this paper. The research of both authors was supported by an internal grant from the Columbia University Graduate School of Business.

REFERENCES

Akella, R. and P. R. Kumar. 1986. Optimal Control of Production Rate in A Failure Prone Manufacturing System. *IEEE Trans. Automatic Control*, **AC-31**, **2**, 116–126.

Balinski, M. and R. Quandt. 1964. On an Integer Program for a Delivery Problem. *Opns. Res.* **12**, 300–304.

Bambos, N. and J. Walrand. 1993. Scheduling and Stability Aspects of a General Class of Parallel Processing Systems. *Advances in Applied Probability*, **25**, 176–202.

Bazaraa, M. S., J. Jarvis, and H. D. Sherali. 1990. *Linear Programming and Network Flows*. John Wiley & Sons, New York.

Bitran, G. R. and S. Dasu. 1992. A Review of Open Queueing Network Models of Manufacturing Systems. *Queueing Systems*, **12**, 95–134.

Brill, P. H. and L. Green. 1984. Queues in which Customers Receive Simultaneous Service from a Random Number of Servers: A System Point Approach. *Mgmt. Sci.* **30**, 51–68.

Buzacott, J. A. and J. G. Shanthikumar. 1992. Design of Manufacturing Systems Using Queueing Models. *Queueing Systems*, **12**, 135–214.

Ceria, S., N. Paolo, and A. Sassano. 1995. A Lagrangian-Based Heuristic for Large-Scale Set Covering Problems. Working Paper, Columbia University.

Charnes, A. and M. Miller. 1956. A Model for the Optimal Programming of Railway Freight Train Movements. *Mgmt. Sci.* **3**, 74–92.

Coffman, E. G., Jr. and G. S. Lueker. 1991. *Probabilistic Analysis of Packing and Partitioning Algorithms*. John Wiley & Sons, New York.

Courcoubetis, C. and M. I. Reiman. 1987. Optimal Control of a Queueing System with Simultaneous Service Requirements. *IEEE Trans. Automatic Control*, **AC-32**, 717–727.

Courcoubetis, C., M. I. Reiman, and B. Simon. 1987. Stability of a Queueing System with Concurrent Service and Locking. *SIAM J. Comput.* **16**, 169–178.

COURCOUBETIS, C. AND U. G. ROTHBLUM. 1991. On Optimal Packing of Randomly Arriving Objects. *Math. O. R.* **16,** 176–194.

COURCOUBETIS, C. AND R. WEBER. 1994. Stability of Flexible Manufacturing Systems. *Opns. Res.* **42,** 947–957.

DESROCHERS, M., J. DESROSIERS, AND M. SOLOMON. 1992. A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows. *Opns. Res.* **40,** 342–354.

DURRETT, R. 1991. *Probability: Theories and Examples*. Brooks/Cole Publishing, Pacific Grove, CA.

FEDERGRUEN, A. AND L. GREEN. 1984. An *M/G/c* Queue in which the Number of Servers Required is Random. *J. Appl. Probability*, **21,** 583–601.

GANS, N. F. AND G. J. VAN RYZIN. 1996a. Optimal Scheduling of a Parallel Processing Queueing System. Working Paper, OPIM Department, The Wharton School, University of Pennsylvania.

GANS, N. F. AND G. J. VAN RYZIN. 1996b. Dynamic Vehicle Dispatching: Optimal Heavy Traffic Performance and Practical Policies. Working Paper: OPIM Department, The Wharton School, University of Pennsylvania.

GELENBE, E. AND G. PUJOLLE. 1987. *Introduction to Queueing Networks*. John Wiley & Sons, Chichester.

GERSHWIN, S. B. 1994. *Manufacturing Systems Engineering*. Prentice Hall, Englewood Cliffs, NJ.

GREEN, L. 1980. A Queueing System in which Customers Require a Random Number of Servers. *Opns. Res.* **28,** 1335–1346.

GREEN, L. 1981. Comparing Operating Characteristics of Queues in which Customers Require a Random Number of Servers. *Mgmt. Sci.* **27,** 65–74.

GREEN, L. 1984a. A Multiple Dispatch Queueing Model of Police Patrol Operations. *Mgmt. Sci.* **30,** 653–665.

GREEN, L. 1984b. A Queueing System with Auxiliary Servers. *Mgmt. Sci.* **30,** 1207–1216.

GREEN, L. 1985. A Queueing System with General-Use and Limited-Use Servers. *Opns. Res.* **33,** 168–182.

HARRISON, J. M. 1988. Brownian Models of Queueing Networks with Heterogeneous Customer Populations. *Stochastic Differential Systems, Stochastic Control Theory and Applications*. W. Fleming and P. L. Lions (eds.), IMA **10,** Springer-Verlag, New York, 147–186.

JACOBSON, P. A. AND E. D. LAZOWSKA. 1982. Analyzing Queueing Networks with Simultaneous Resource Possession. *Comm. ACM*, **25,** 142–151.

KAUFMAN, J. S. AND Y. T. WANG. 1989. A Conservation Law Based Approximate Analysis for a Class of Simultaneous Resource Possession Problems. *Performance Evaluation*, **10,** 263–280.

KIMEMIA, J. AND S. B. GERSHWIN. 1983. An Algorithm for the Computer Control of a Flexible Manufacturing System. *IIE Trans.* **15,** 4, 353–362.

KLEINROCK, L. 1976. *Queueing Systems*. Vol. 2, John Wiley & Sons, New York.

KRICHAGINA, E. V., R. RUBIO, M. I. TAKSAR, AND L. M. WEIN. 1992. A Dynamic Stochastic Stock Cutting Problem. Working Paper, Massachusetts Institute of Technology.

LAW, A. M. AND W. D. KELTON. 1982. *Simulation Modeling and Analysis*. McGraw-Hill, New York.

ROSS, K. W. AND D. H. K. TSANG. 1989. The Stochastic Knapsack Problem. *IEEE Trans. Comm.* **37,** 740–747.

ROSS, K. W. AND D. D. YAO. 1990. Monotonicity Properties for the Stochastic Knapsack. *IEEE Trans. Information Theory*, **36,** 1173–1179.

SAUER, C. H. 1981. Approximate Solution of Queueing Networks with Simultaneous Resource Possession. *IBM J. Res. Development*, **25,** 894–903.

SETHI, S. P. AND Q. ZHANG. 1994. *Hierarchical Decision Making in Stochastic Manufacturing Systems*. Birkhauser, Boston.

WALRAND, J. 1988. *An Introduction to Queueing Networks*. Prentice Hall, Englewood Cliffs, NJ.

WEIN, L. M. 1992. Dynamic Scheduling of a Multiclass Make-to-Stock Queue. *Opns. Res.* **40,** 724–735.

WHITT, W. 1985. Blocking when Service is Required from Several Facilities Simultaneously. *AT&T Technical J.* **64,** 1807–1856.

WHITT, W. 1989. Planning Queueing Simulations. *Mgmt. Sci.* **35,** 1341–1366.

WOLFF, R. W. 1989. *Stochastic Modeling and the Theory of Queues*. Prentice Hall, Englewood Cliffs, NJ.