WILEY
InterScience®
DISCOVER SOMETHING GREAT

Modeling and Analysis of Uncertain Time-Critical Tasking Problems

Donald P. Gaver,[1] Patricia A. Jacobs,[1] Gennady Samorodnitsky,[2] Kevin D. Glazebrook[3]

[1] *Operations Research Department, Naval Postgraduate School, Monterey, CA 93943*

[2] *School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY 14853*

[3] *Department of Mathematics and Statistics and the Management School, Lancaster University, Lancaster LA1 4YX, United Kingdom*

Received 11 April 2006; accepted 11 April 2006
DOI 10.1002/nav.20162
Published online 22 May 2006 in Wiley InterScience (www.interscience.wiley.com).

Abstract: This paper describes modeling and operational analysis of a generic asymmetric service-system situation in which (a) Red agents, potentially threatening, but in another but important interpretation, are isolated friendlies, such as downed pilots, that require assistance and "arrive" according to some partially known and potentially changing pattern in time and space; and (b) Reds have effectively limited unknown deadlines or times of availability for Blue service, i.e., detection, classification, and attack in a military setting or emergency assistance in others. We discuss various service options by Blue service agents and devise several approximations allowing one to compute efficiently those proportions of tasks of different classes that are successfully served or, more generally, if different rewards are associated with different classes of tasks, the percentage of the possible reward gained. We suggest heuristic policies for a Blue server to select the next task to perform and to decide how much time to allocate to that service. We discuss this for a number of specific examples. © 2006 Wiley Periodicals, Inc.* Naval Research Logistics 53: 588–599, 2006.

1. THE PROBLEM

This paper addresses modeling and operational analysis of a generic asymmetrical service-system situation in which (a) Red agents, such as military or facility-destructive hostile threats, arrive according to some partially known and possibly changing pattern in time and space; and (b) Reds have effectively limited unknown deadlines, or times of availability for Blue service in a military setting, detection, classification, and attack. Cases of known deadlines are important and analogous; see [15–17, 8].

Think of the Reds as presenting tasks to be performed or be subjects to be served. In a military context Reds are perceived enemy targets, but in a medical emergency room setting they are arriving casualties. In a call center they are requests for information; see [2]. In a Homeland Security (HLS) scenario a Red may be a container ship approaching a port or a truck approaching a border, either possibly carrying explosives or chemical–biological offensive agents. Blue has the problem of processing such Red tasks effectively and efficiently under time constraints and limited information, hence, the necessity to control the amount of service given so as to keep waiting times short.

Appropriate service effort typically differs between task classes; it may not always be completely provided and may be partial and incomplete, owing to deficiency of time, information, or resources. In general, task service is by a Blue force of task-server agents also of various classes, possibly varying in number and organization, and at different locations, but which attempt to share information and the service burden. Such complex agent systems are considered elsewhere, using insights provided in this paper.

The Blue military objective is to successfully service as many tasks as possible rather than to minimize queues, while hostile Reds attempt to avoid "service," at least until they can accomplish their purpose, often to thwart Blue. The models presented and analyzed suggest Blue force requirements and capability combinations for confronting specified challenges with acceptable success rates.

To summarize, such service-system issues arise ubiquitously in military operations of all kinds, as well as in HLS and in military force protection, downed pilot recovery, emergency management situations, and many natural haz-

Correspondence to: D. P. Gaver (dgaver@nps.edu); P. A. Jacobs (pajacobs@nps.edu); G. Samorodnitsky (gennady@orie.cornell.edu); K. D. Glazebrook (K.glazebrook@lancaster.ac.uk)

ard response scenarios, such as after earthquakes or tidal waves.

Except under the most simple and unrealistic scenarios, exact analytical solutions of the models we will consider are either impossible or so computationally challenging as to be impractical for real-time use. For example, some of the decision problems discussed may be regarded as restless bandit problems, a class of decision processes introduced by Whittle [27] and shown to be almost certainly intractable by Papadimitriou and Tsitsiklis [20]. Therefore, good approximations and/or heuristics are in order, and to display and evaluate these is one goal of this paper. In Section 2 we describe the basic model: A several-class Red task arrival stream confronts a single Blue server that can process one Red task at a time. The Red tasks each have independent randomly limited availability time for processing; so if the availability time for a task exceeds the delay, (any waiting plus service time), then the service is delivered. Otherwise, the task is lost (leaks through defenses, dies while awaiting treatment, etc.). In many applications it is natural to assume that the server can observe the loss of a Red task while in service, e.g., the death of a patient or the completion of service, e.g., stabilizing the patient. In other applications the server is unable to observe completion of service or defection. In the military context not observing completion of service may mean that a Blue agent may not be able to observe, in real time, whether the target has been destroyed. Similarly, not observing defection during service may mean that a Blue agent cannot see in real time whether the enemy has already accomplished his task. Our results are for an M/G/1-type queueing model where the tasks are lost after independent exponential times; see [1]. In Section 2 we present an exact expression for the probability that an arriving task is successfully served. We also present two useful approximations for the probability that an arriving task is successfully served that are easier to evaluate numerically than the exact expression.

In Section 3 we examine Blue defense's decision options so as to achieve maximum rate of successful task service, i.e., minimum leakage probability. These are (a) to select for next service the waiting task with the greatest chance of survival to be serviced and (b) to allocate service resources so as to balance time spent on the currently served task against losing tasks waiting. We study policies with fixed/constant allocated processing times for each task class. The probability that a task is served successfully is related to the allocated processing time: if the availability time for a task exceeds any waiting plus allocated processing time, then service is delivered with a possibly successful outcome; service will be successful if in addition the allocated processing time exceeds the service time of the task. The operational tradeoff is to allocate enough processing time to a task to have a large enough probability of serving it successfully versus allocating too much processing time and losing too many tasks that are waiting in the queue. Approximate heuristic control policies are proposed that may then be refined and evaluated by use of heuristic search procedures such as Genetic Algorithms and by adaptations of Dynamic Programming. Simulation results are presented to explore the effectiveness of the present heuristic policies.

2. THE PROBABILITY OF SUCCESSFUL TASK COMPLETION

Consider the case of a single Blue service-providing agent (BSA) confronting a random stream of loss-susceptible Red service-requiring agents (RSAs). In this section we will present an exact (but computationally intensive) expression for the probability that an arriving task will be served successfully. This case will be followed by two approximate procedures of increasing refinement that are computationally more attractive. These approximations will then be used in Section 3 in heuristic procedures to determine constant processing times to allocate to tasks that are starting service so as to maximize the long-run fraction of tasks that are served successfully.

Next let tasks from J ($J \geq 1$) task classes (or types) arrive at a service facility according to independent Poisson processes with rate λ_j for the jth task class; let $\lambda = \sum_{j=1}^{J} \lambda_j$. Service times are independent, and service times for each task class are identically distributed; let $\mathbf{C}_j$ denote a generic service time for class j. Each task of the jth class has an exponentially distributed deadline with the positive finite mean $1/\theta_j$, with the usual independence assumptions. If the service discipline is first-come first-served (FCFS), an arriving task will eventually start service if its virtual waiting time $\mathbf{W}$ (the sum of remaining service times of all tasks that are waiting for or in service when the task arrives), is less than its time until loss; the steady-state probability that an arriving task of class j will start service is $\psi(\theta_j) = E[e^{-\theta_j \mathbf{W}}]$. This model with one task class has already been considered by Baccelli et al. [1]. We start with the case where tasks whose deadlines have elapsed when they reach the server are not served, but no defection occurs while in service. A task whose deadline is shorter than the waiting time at the moment of arrival does not enter the system, i.e., is lost.

2.1. Solution to a Modification of the Takaçs–Beneš Equation for Multiple Task Classes

Forward Kolmogorov equation arguments analogous to those carried out in the case of a single class of tasks (see [1]) show that the Laplace transform of the steady-state virtual waiting time in the system satisfies

$$\psi(s;\ \boldsymbol{\theta}) = p_0(\boldsymbol{\theta}) + \sum_j \rho_j\ \delta_j\ (s)\psi(s + \theta_j;\ \boldsymbol{\theta}), \qquad (2.1)$$

where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_J)$ and $p_0(\boldsymbol{\theta})$ is the limiting probability the virtual waiting time is equal to 0;

$$\delta_j\ (s) = \frac{1 - b_j^*(s)}{sE[\mathbf{C}_j]}, \qquad (2.2)$$

with $\rho_j = \lambda_j E[\mathbf{C}_j]$, and $b_j^*(s) = E[e^{-s\mathbf{C}_j}]$. Here $\mathbf{C}_j$ is a generic service time of a class j task; put $\rho = \sum_{\mathrm{j}=1}^{\mathrm{J}} \rho_{\mathrm{j}}$. Furthermore,

$$p_0(\boldsymbol{\theta}) = 1 - \sum_j \rho_j \psi(\theta_j;\ \boldsymbol{\theta}). \qquad (2.3)$$

An iterative procedure similar to the one used in the single task class case shows that the probability $\psi(\theta_j) \equiv \psi(\theta_j;\ \boldsymbol{\theta})$ that a task of class j will start service (not be lost while in queue) satisfies, for each $n = 1, 2, \ldots,$ the equation

$$\psi(\theta_j) = A^{(j)}(0,\ \boldsymbol{\theta}) + \ldots + A^{(j)}(n,\ \boldsymbol{\theta}) - [A^{(j)}(0,\ \boldsymbol{\theta}) + \ldots$$
$$+ A^{(j)}(n,\ \boldsymbol{\theta})] \sum_{i=1}^{J} \rho_i \psi(\theta_i) + E_j(n + 1;\ \boldsymbol{\theta}), \qquad (2.4)$$

where

$$A^{(j)}(0,\ \boldsymbol{\theta}) = 1$$

$$A^{(j)}(n,\ \boldsymbol{\theta}) = \sum_{k_1=1}^{J} \rho_{k_1}\ \delta_{k_1}\ (\theta_j) \sum_{k_2=1}^{J} \rho_{k_2}\ \delta_{k_2}\ (\theta_j + \theta_{k_1})$$
$$\times \ldots \times \sum_{k_n=1}^{J} \rho_{k_n}\ \delta_{k_n}\ (\theta_j + \theta_{k_1} + \ldots + \theta_{k_{n-1}})$$

and

$$E^{(j)}(n;\ \boldsymbol{\theta}) = \sum_{k_1=1}^{J} \rho_{k_1}\ \delta_{k_1}\ (\theta_j) \ldots \sum_{k_n=1}^{J} \rho_{k_n}\ \delta_{k_n}\ (\theta_j + \theta_{k_1} + \ldots$$
$$+ \theta_{k_{n-1}})\psi(\theta_j + \theta_{k_1} + \ldots + \theta_{k_{n-1}} + \theta_{k_n})$$

for $n = 1, 2, \ldots.$ Since $E^{(J)}(n;\ \boldsymbol{\theta}) \to 0$ as $n \to \infty$, we obtain a system of linear equations for success probabilities,

$$\psi(\theta_j) = \sum_{k=0}^{\infty} A^{(j)}(k;\ \theta) - \sum_{k=0}^{\infty} A^{(j)}(k;\ \theta) \sum_{i=1}^{J} \rho_i \psi(\theta_i), \qquad (2.5)$$

for $j = 1, \ldots, J$, which we can solve by replacing the infinite sums by their finite approximations.

For the model with one task class, Baccelli et al. [1] solve the equation corresponding to (2.5) to obtain

$$\psi(\theta) = \frac{\sum_{k=0}^{\infty} A(k;\ \theta)}{1 + \rho \sum_{k=0}^{\infty} A(k;\ \theta)}, \qquad (2.6)$$

where

$$A(0,\ \theta) = 1$$

$$A(n;\ \theta) = \rho^n\ \delta\ (n\theta)\ \delta\ ((n - 1)\theta) \times \ldots \times \delta\ (\theta). \qquad (2.7)$$

If a task deadline's elapse is detectable during service and the task is then terminated, then the distribution of service time, $\mathbf{C}$, must be replaced by that of $\mathbf{C}_T = \min(\mathbf{C}, \text{deadline})$, the allowed service time. Consequently, the service times that contribute to the virtual waiting time are, thanks to the exponential deadline assumption, iid with mean

$$E[\mathbf{C}_T] = \frac{1 - E[e^{-\theta\mathbf{C}}]}{\theta} \qquad (2.8)$$

and tail-transform now

$$\delta_T\ (s;\ \theta) = \frac{1 - E[e^{-(\theta+s)\mathbf{C}}]}{(\theta + s)E[\mathbf{C}_T]} = \frac{\delta\ (\theta + s)}{\delta\ (\theta)}. \qquad (2.9)$$

These replace $E[\mathbf{C}]$ in ρ and $\delta\ (s)$ in the previous solution, (2.6).

Solving the system of Eqs. (2.5) is a computationally intensive proposition. Computationally attractive alternatives are self-thinning approximations described next.

2.2. Approximation I

If $\mathbf{W}$ has the steady-state virtual workload distribution, then the limiting probability that an arbitrary arriving task of class j will eventually start service (its deadline does not expire while it is waiting in queue), is $\psi(\theta_j) = E[e^{-\theta_j \mathbf{W}}]$ in the long run. We neglect the dependence between the fates of different tasks by pretending that whether an arriving task will start service is decided via a sequence of independent coin tosses. The resulting system becomes an M/G/1 queue with traffic intensity $\sum_{j=1}^{J} \rho_j \psi(\theta_j)$.

We start with a self-thinning approximation to the entire aggregation of tasks. Let p be the overall proportion of tasks that start service. Suppose we thin arriving tasks with probability p. Then the Pollaczek–Khinchine (P–K) formula for

M/G/1 queues yields the transform of the virtual waiting time in queue:

$$E[e^{-s\mathbf{W}}] = \frac{1 - p\sum_{j=1}^{J} \lambda_j E[\mathbf{C}_j]}{1 - \sum_{j=1}^{J} \lambda_j p\left[\dfrac{1 - E[e^{-s\mathbf{C}_j}]}{s}\right]}. \quad (2.10)$$

On the other hand, the probability that an arriving task will eventually start service is

$$p = \sum_{j=1}^{J} \frac{\lambda_j}{\lambda} E[e^{-\theta_j \mathbf{W}}] = \sum_{j=1}^{J} \frac{\lambda_j}{\lambda}\left[\frac{1 - p\rho}{1 - p\sum_{k=1}^{J} \rho_k\, \delta_k\,(\theta_j)}\right] \equiv B(p). \quad (2.11)$$

Note that $B(p)$ follows from (2.10) and is decreasing in p on $[0, 1/\rho]$ and is always between 0 and 1. Hence, the above equation always has a unique solution $\tilde{p}$ in [0, 1]. The approximation for the probability a task of class j starts service is

$$\psi(\theta_j) = E[e^{-\theta_j \mathbf{W}}] = \frac{1 - \tilde{p}\rho}{1 - \tilde{p}\sum_{k=1}^{J} \rho_k\, \delta_k\,(\theta_j)}. \quad (2.12)$$

In the special case of a model with one task class, the approximation becomes the solution to a quadratic equation. If $\mathbf{W}$ has the steady-state virtual workload distribution, then the probability that an arriving task is successfully served is $\psi(\theta) = E[e^{-\theta \mathbf{W}}]$. The approximation that whether an arriving task will start service is decided by an independent coin flip results in an M/G/1 queue with traffic intensity $\rho\psi(\theta)$ and the P–K formula yields

$$\psi(\theta) \equiv E[e^{-\theta\mathbf{W}}] = \frac{1 - [\psi(\theta)\lambda]E[\mathbf{C}]}{1 - [\psi(\theta)\lambda]E[\mathbf{C}]\left\{\dfrac{1 - E[e^{-\theta\mathbf{C}}]}{\theta E[\mathbf{C}]}\right\}}. \quad (2.13)$$

This simple formula differs somewhat from the one task version of (2.1) of the modified Takaçs–Beneš (T–B) equation for the same assumed arrival–queue interaction, but is in handy closed form.

The expression (2.13) is a quadratic in the desired probability, the solution of which is

$$\psi(\theta) = \frac{2}{1 + \rho + \sqrt{(1+\rho)^2 - 4\rho\,\delta\,(\theta)}}, \quad (2.14)$$

where $\rho = \lambda E[\mathbf{C}]$ as usual, and $\delta\,(\theta)$ given by (2.2) is the transform of the service/completion time tail or survivor distribution. The approximate probability of successful transit to the server given by this simple expression is unity when $\theta \to 0$ (no degradation or infinite deadline), as long as $\rho < 1$; if $\theta \to \infty$ then, since deadlines are now stringent, the only hope of initiating service is to arrive when there is no server activity, i.e., with probability $1/(1 + \rho)$, and this time any (positive) ρ-value is permitted. In general, there are no restrictions on ρ in (2.14): a long queue generates many rejections and thus does not ever remain long or grow indefinitely; in practice this situation would be not tolerated for long. Empirically, the simple expressions, (2.14) and (2.27) below, supply a lower bound to the exact solution of such a reneging or refusal model. Note that the same logic gives an approximation for the transform of virtual waiting time of non-refused tasks, $\mathbf{W}$, with the formula

$$\psi(\xi;\, \theta) = \frac{1 - \lambda\psi(\theta)E[\mathbf{C}]}{1 - \lambda\psi(\theta)E[\mathbf{C}]\left\{\dfrac{1 - E[e^{-\xi\mathbf{C}}]}{\xi E[\mathbf{C}]}\right\}}. \quad (2.15)$$

2.3. Approximation II

A more refined approximation makes use of the fact that the task that arrives when the server is idle always starts service. First of all,

$$E[e^{-s\mathbf{W}}] = P\{\mathbf{W} = 0\} + [1 - P\{\mathbf{W} = 0\}]E[e^{-s\mathbf{W}}|\mathbf{W} > 0]. \quad (2.16)$$

Hence, using the P–K formula for $E[e^{-s\mathbf{W}}]$ results in

$$E[e^{-s\mathbf{W}}|\mathbf{W} > 0] = \frac{1-\rho}{\rho}\,\frac{\sum_{j=1}^{J} \rho_j\, \delta_j\,(s)}{1 - \sum_{j=1}^{J} \rho_j\, \delta_j\,(s)}. \quad (2.17)$$

We view ρ in this expression as the traffic intensity during a busy period. Then the same logic as the one used in derivation of Approximation I above says that the acceptance probability during a busy period should approximately satisfy (2.17) with ρ replaced by ρp, where p is the proportion of tasks arriving during a busy period that reach the server. In other words, an auxiliary randomization (biased coin flip) adjusts for the imposition of the deadline, as before in Approximation I, but in a somewhat more refined manner. The approximation is

$$E[e^{-s\mathbf{W}}|\mathbf{W} > 0] = \frac{1 - \rho p}{1 - \sum_{j=1}^{J} p\rho_j\, \delta_j\,(s)}\,\frac{\sum_{j=1}^{J} \rho_j\, \delta_j\,(s)}{\rho}. \quad (2.18)$$

Further, p must satisfy the relation

$$p = \sum_{j=1}^{J} \frac{\lambda_j}{\lambda} E[e^{-\theta_j \mathbf{W}} | \mathbf{W} > 0]$$

$$= \sum_{j=1}^{J} \frac{\lambda_j}{\lambda} \frac{1 - \rho p}{1 - \sum_{k=1}^{J} p\rho_k \, \delta_k\,(\theta_j)} \frac{\sum_{k=1}^{J} \rho_k \, \delta_k\,(\theta_j)}{\rho}. \quad (2.19)$$

The same argument as before shows that this equation has a unique solution $\tilde{p}_b$ in $(0, 1 \wedge \rho^{-1})$.

Let

$$\psi_+(\theta_j) = \frac{1 - \tilde{p}_b\rho}{1 - \sum_{k=1}^{J} \tilde{p}_b\rho_k \, \delta_k\,(\theta_j)} \frac{\sum_{k=1}^{J} \rho_k \, \delta_k\,(\theta_j)}{\rho} \quad (2.20)$$

for $j = 1, \ldots, J$.

The expected length of a busy period satisfies the approximate relation

$$E[\mathbf{B}] = \sum_{j=1}^{J} \frac{\lambda_j}{\lambda} E[\mathbf{C}_j] + \tilde{p}_b \, \rho E[\mathbf{B}] \quad (2.21)$$

and so,

$$E[\mathbf{B}] = \frac{\sum_{j=1}^{J} \frac{\lambda_j}{\lambda} E[\mathbf{C}_j]}{1 - \tilde{p}_b\rho}. \quad (2.22)$$

Since

$$P\{\mathbf{W} = 0\} = \frac{\lambda^{-1}}{\lambda^{-1} + E[\mathbf{B}]}, \quad (2.23)$$

our final approximation is

$$\psi_B(\theta_j) = P\{\mathbf{W} > 0\}\psi_+(\theta_j) + P\{\mathbf{W} = 0\}$$

$$= \frac{1 - \tilde{p}_b\rho + \rho\psi_+(\theta_j)}{1 + \rho(1 - \tilde{p}_b)}. \quad (2.24)$$

If there is only one task class, the above simplifies. Let

$$\psi_+(\theta) = E[e^{-\theta \mathbf{W}} | \mathbf{W} > 0] \quad (2.25)$$

be the marginal long-run rate of task acceptance given that the server is busy. From the P–K formula

$$E[e^{-s\mathbf{W}} | \mathbf{W} > 0] = \left[\frac{(1 - \rho)\rho \, \delta\,(s)}{1 - \rho \, \delta\,(s)}\right] \frac{1}{\rho} = \frac{(1 - \rho) \, \delta\,(s)}{1 - \rho \, \delta\,(s)}. \quad (2.26)$$

Equations (2.18) and (2.19) become

$$\psi_+(\theta) = (1 - \rho\psi_+(\theta)) \frac{\delta\,(\theta)}{1 - \rho\psi_+(\theta) \, \delta\,(\theta)}. \quad (2.27)$$

The solution of (2.27) is

$$\psi_+(\theta) = \frac{2 \, \delta\,(\theta)}{(1 + \rho \, \delta\,(\theta)) + \sqrt{(1 + \rho \, \delta\,(\theta))^2 - 4\rho \, \delta\,(\theta)^2}}. \quad (2.28)$$

Equation (2.22) becomes

$$E[\mathbf{B}] = E[\mathbf{C}] + \rho\psi_+(\theta)E[\mathbf{B}] = E[\mathbf{C}]/(1 - \rho\psi_+(\theta)). \quad (2.29)$$

Consequently, an alternating renewal process argument gives us, as the long-run proportion of time that the server is idle,

$$P\{\mathbf{W} = 0\} = \frac{\lambda^{-1}}{\lambda^{-1} + E[\mathbf{B}]} = \frac{1 - \rho\psi_+(\theta)}{1 + \rho[1 - \psi_+(\theta)]}. \quad (2.30)$$

The probability that an arriving task is admitted (not refused and eventually served) of Eq. (2.24) simplifies

$$\psi_2(\theta) = P\{\mathbf{W} = 0\} + (1 - P\{\mathbf{W} = 0\})\psi_+(\theta)$$

$$= \frac{1}{1 + \rho[1 - \psi_+(\theta)]}, \quad (2.31)$$

which differs from (2.14) owing to the more refined conditioning imposed.

Table 1 compares the exact probability of an arriving task reaching the server and the two approximations for a model with one task class. The tasks arrive according to a Poisson process with rate 1 per minute. The times until task loss are independently and identically distributed, having an exponential distribution with mean 10. The service time distributions considered are exponential with mean 1.11; a constant service time equal to 1.05; and the exponentially truncated stable law; see [10]. The order of the stable law is $\alpha = 0.1$ and scale $\nu = 0.1$; e.g., the Laplace transform of the stable service time is $E[e^{-s\mathbf{S}}] = \exp\{-(\nu s)^{\alpha}\}$; see [9]; the mean of the exponential truncation time is 5; the service time is the minimum of the truncation time and the stable law time. The service times are chosen so that the probability of successful service completion by a task that starts service is 0.9. Task loss is observable during service; if task loss occurs before service completion then the service ends unsuccessfully at the time of task loss. The stable law is chosen because it exhibits a very long right tail (no mo-

Table 1. Comparison of approximations to exact results: loss times are exponential with mean 10; Poisson arrivals with rate 1.

Service time distribution	T–B for probability arriving task starts service	Approximation (2.14) for probability arriving task starts service	Approximation (2.31) for probability arriving task starts service
Exponential mean 1.11	0.81	0.77	0.79
Constant = 1.05	0.85	0.82	0.83
Exponentially truncated stable law	0.97	0.97	0.97

ments exist for the untruncated version, so truncation is required). Correspondingly, the truncated version has a high concentration of small values, which accounts for the noticeably high success probability for that case in Table 1. Approximation II, (2.31), improves slightly on Approximation I, (2.14), in all cases explored numerically to date, but approximation (2.14) is somewhat easier to evaluate numerically.

Table 2 below displays results of the two approximations for the probability that an arriving task will start service and the results from a simulation for a model with three task classes; see [6]. The simulation results are for 50 replications with 30,000 tasks of each class per replication. Both service completion and task loss while in service are observable. Task loss in queue is also observable. Results for three cases are displayed in Table 2. For each case, the three task classes each have gamma distributed service times with parameters chosen so that the probability of a task of class 1 (respectively, class 2, class 3) starting service will complete it successfully is 0.95 (respectfully, 0.91 and 0.67). Both approximations give very reasonable lower bounds for the simulation results.

3. ALLOCATION OF SERVICE TO ARRIVING TASKS

In this section we consider the management of a single server queue with arrivals of tasks having independent random deadlines. Each task that is served before its deadline elapses earns a reward; the amount of reward can depend on the class of the task. We suggest dynamic policies for choosing at each decision epoch (these being the conclusion of each allocated service and the time of any arrival to an empty system) which task among those waiting should next be processed together with the duration of the processing time allocated so as to maximize the percentage

Table 2. Probability that arriving task will start service and will complete service successfully.

Task class	Task arrival rate	Mean task service time	Task shape parameter	Task loss rate	Approx. prob. (2.12) task will start service [successfully complete service]	Approx. prob. (2.24) task will start service [successfully complete service]	Simulation fraction of tasks that will start service (std. error) [successfully complete service (std. error)]
1	0.2	1.1	0.2	0.05	0.56 [0.54]	0.60 [0.57]	0.65 (0.0007) [0.62 (0.0007)]
2	0.2	3.66	0.2	0.03	0.64 [0.58]	0.68 [0.62]	0.74 (0.0006) [0.67 (0.0006)]
3	0.2	13.19	0.2	0.1	0.45 [0.30]	0.47 [0.31]	0.49 (0.0007) [0.33 (0.0006)]
1	0.2	1	1	0.05	0.62 [0.59]	0.65 [0.62]	0.70 (0.0006) [0.67 (0.0005)]
2	0.2	3	1	0.03	0.70 [0.63]	0.73 [0.66]	0.78 (0.0004) [0.71 (0.0004)]
3	0.2	5	1	0.1	0.48 [0.32]	0.50 [0.33]	0.53 (0.0005) [0.35 (0.0004)]
1	0.2	0.98	10	0.05	0.65 [0.62]	0.67 [0.64]	0.73 (0.0005) [0.69 (0.0005)]
2	0.2	2.87	10	0.03	0.73 [0.66]	0.75 [0.68]	0.80 (0.0003) [0.73 (0.0004)]
3	0.2	4.14	10	0.1	0.50 [0.33]	0.52 [0.34]	0.55 (0.0004) [0.37 (0.0004)]

of reward received. A dynamic programming approach to this problem is computationally challenging for any problem of reasonable size. Thus, we consider heuristic service allocation policies. One heuristic uses the self-thinning approximations of Section 2 to ease the computational burden. The second heuristic is a myopic policy.

In Subsection 3.1 we develop a heuristic policy for a model with one task class. The policy allocates a constant processing time τ to each task that begins service. If the service time of the task ends before the allocated constant processing time and the task has not been lost, the task is served successfully; otherwise, it is not served successfully. In Subsection 3.2 we consider the much more difficult problem with multiple task classes.

3.1. One Task Class: A Markovian Heuristic for Service Allocation

Consider a model in which allocated processing times are independent random times having an exponential distribution with mean $1/\mu$. We will choose the μ that maximizes the probability of successful service for an arriving task when the task is allocated an exponential processing time. Our policy will be to use the maximizing exponential allocation rate to set the constant allocated processing time $\tau = 1/\mu$. The policy will be to allocate to each task that starts service the constant processing time $\tau = 1/\mu$. We assume that neither service time completions nor task losses from service are observed. Actual service times have distribution function F and hence a task that starts service will be served successfully if the actual service finishes before that task is lost and before the exponentially distributed allocated processing time ends.

$$p(\mu) = P\{\text{allocated service is successful}\} = \int_0^\infty F(dt)e^{-(\theta+\mu)t} \quad (3.1)$$

Example. If the actual service time has a gamma distribution with shape parameter r and mean r/ν, then $p(\mu) = [\nu/(\nu + \theta + \mu)]^r$.

Under the policy of allocated processing time, which is $\exp(\mu)$, then the process of the number of tasks waiting or being served on the assumption that neither losses nor actual service completions are observable is a birth–death process with

$$P\{X(t+h) = n-1 | X(t) = n\} = (\mu + \theta(n-1))h + o(h), \quad \text{for } n > 0$$

$$P\{X(t+h) = n+1 | X(t) = n\} = \lambda h + o(h), \quad \text{for } n \geq 0. \quad (3.2)$$

If the loss of the currently served task is observable but the actual service completion is not observable then the system remains a birth–death process with all birth rates equal to λ and death rate when there are n tasks waiting or being served, $\mu + \theta n$. If both loss from service and actual service completions are observable then the system is only Markov if the actual service times are exponential with some mean $1/\zeta$. In this case the number of tasks waiting for service and being served is a birth–death process with all birth rates equal to the arrival rate λ and death rate when there are n tasks waiting or being served equal to $\mu + \zeta + \theta n$.

Assume that neither losses from service nor service completion are observable. The rate of allocated process completions from the system is $\mu P\{\text{system is non-empty}\}$. Using standard birth–death process results, the rate of allocated process completions is

$$\mu[1 - \pi_0(\mu)] = \mu\left[\frac{\sum_{n=1}^{\infty}\prod_{k=0}^{n-1}\dfrac{\lambda}{\mu + k\theta}}{1 + \sum_{n=1}^{\infty}\prod_{k=0}^{n-1}\dfrac{\lambda}{\mu + k\theta}}\right]. \quad (3.3)$$

To obtain the rate of successful service completions we simply multiply the expression in (3.3) by $p(\mu)$. To obtain the proportion of arriving tasks that are served successfully, $p_F(\lambda, \theta, \mu)$, multiply (3.3) by $p(\mu)/\lambda$.

The design goal is to choose μ to maximize the proportion of arriving tasks that are served successfully, $p_F(\lambda, \theta, \mu)$. If the service time distribution does not have an atom at 0, then $\mu \to 0$, $p_F(\lambda, \theta, \mu) \to 0$ since $p_F(\lambda, \theta, \mu) \leq \mu/\lambda$. As $\mu \to \infty$, then note from (3.3) that if $\lambda < \mu$ then

$$p_F(\lambda,\theta,\mu) \leq \frac{\mu p(\mu)}{\lambda}\left[\frac{\sum_{n=1}^{\infty}\left[\dfrac{\lambda}{\mu}\right]^n}{1 + \sum_{n=1}^{\infty}\left[\dfrac{\lambda}{\mu}\right]^n}\right] = p(\mu) \quad (3.4)$$

and $p(\mu) \to 0$, $\mu \to \infty$ and so $p_F(\lambda, \theta, \mu) \to 0$, $\mu \to \infty$. Note that $p_F(\lambda, \theta, \mu)$ is continuous in μ and thus, the maximum is achieved. We infer that for given (λ, θ) the μ maximizing $p_F(\lambda, \theta, \mu)$ must lie in some specified finite range $(\underline{\mu}, \bar{\mu})$.

Examples. Table 3 displays the constant allocated processing times that maximize the probability of successful service for an arriving task. The service times considered

Table 3. Service allocation time and probability of successful service.

Gamma service time			Thinning approximation (2.14)		Birth–death		T–B	
Mean	Shape	Mean time to loss	Max alloc. time	Exact prob. succ. service	Max alloc. time	Exact prob. succ. service	Max alloc. time	Exact prob. succ. service
0.6	3	10	0.95	0.71	1.02	0.71	1	0.71
0.6	10	10	0.91	0.78	1.07	0.76	0.94	0.78
1.0	3	10	1.26	0.50	1.21	0.50	1.31	0.51
1.0	10	10	1.35	0.56	1.32	0.56	1.37	0.56
10	3	10	8.55	0.03	5.01	0.03	8.46	0.03
10	10	10	11.94	0.03	8.18	0.02	11.90	0.03
0.6	1	1	0.61	0.42	0.73	0.42	0.67	0.42
0.6	3	1	0.80	0.37	0.87	0.37	0.84	0.37
0.6	10	1	0.84	0.38	1	0.37	0.86	0.38

have gamma distributions. The time to task loss has an exponential distribution with mean $1/\theta$. Tasks arrive according to a Poisson process with rate $\lambda = 1$. Task losses while in queue are observable. Task losses during service and actual service completion are not observable. The thinning approximation (2.14), the birth–death process model (3.3), and the exact T–B formula (2.6) are used to calculate the constant allocated processing time that maximizes the calculated probability of arriving task service completion. Table 3 also displays the resulting maximum probability of successful service for an arriving task using the exact probability of successful service using the T–B formula for each policy.

Discussion. The two heuristic policies result in probabilities of successful service that are very close to those of the exact (T–B) policy. Thus, there is little lost by using either of the two heuristics. There is some suggestion that the thinning approximation results in constant allocated processing times, resulting in somewhat higher probability of successful service than those of the birth–death process. The constant allocated processing times found using the thinning approximation are closer to those found by the exact T–B calculations.

3.2. Service Allocation for Several Task Classes

Here we consider a much more realistic and difficult scenario: a single BSA confronting a random stream of different loss-susceptible RSAs. The arrival process of RSAs of class j is Poisson (λ_j) (j is a member of $(1, \ldots, J)$) independent of the other task classes. Service times for tasks of class j are independent and have a distribution F_j. Assume the times until loss are independent with those for RSAs of class j, having an exponential distribution with mean $1/\theta_j$. A successfully served task of class j results in a reward r_j.

Realistically, both in emergency room applications and in military applications, the server(s) must decide which customer to serve next and how much time to serve that customer. The decision epochs are the times of service completion if there are tasks in the queue or the time of arrival of a task if the server is idle. We propose below heuristic policies that assist the decision maker (server) to do that. The first heuristic is a Markovian heuristic that uses the self-thinning approximations of Section 2 to ease the computational burden. The second heuristic is a myopic policy.

3.2.1. A Markovian Heuristic Policy with First-Come First-Served Service Discipline

In this subsection we first consider a heuristic policy with FCFS service discipline. In the next subsection we consider a heuristic policy with a simple class priority service discipline. Both heuristics assign a constant processing time to a task that starts service. The constant processing time can depend on the task's class. The constant processing time is chosen as the mean of an exponential processing time that is chosen to maximize the long-run average reward received.

Assume each task j that enters service is allocated an exponential service/processing time with mean $1/\mu_j$. Given a task of class j starts service, the probability it will successfully complete service is

$$p_j(\mu_j) = \mathrm{P}\{\text{an allocated service is successful for a task of class } j \text{ that starts service}\} = \int_0^\infty F_j(dt)e^{-(\theta_j+\mu_j)t}, \quad (3.5)$$

where F_j is the service time distribution for task j. Note that (3.5) is the Laplace transform of the service distribution F_j evaluated at $\theta_j + \mu_j$. Use the approximation (2.24) to estimate the probability an arriving task of class j starts service, $\psi_B(\theta_j)$, for the case in which the exponential allocated processing time is observable but the task service time is not; this calculation can be done for the case in which task loss during service is observable and the case in which task loss during service is not observable. The resulting approximate long-run average reward earned is

$$\sum_{j=1}^{J} \psi_B(\theta_j) p_j(\mu_j) r_j. \tag{3.6}$$

Find those allocated service rates $\{\mu_j;\ j = 1, \ldots, J\}$ that maximize the long-run average reward. The heuristic policy is to serve the arriving tasks FCFS and give each task of class j that enters service a constant allocated processing time equal to $1/\mu_j$. This heuristic balances task congestion at the server with the expected reward received. This heuristic is computationally attractive when the Laplace transforms of the service time distributions are easier to calculate than the distribution functions. Another policy is to find those constant allocated processing times that maximize (3.6) with the modified (3.5).

3.2.2. A Markovian Heuristic Policy with Task Priorities

In this subsection we modify the FCFS service discipline of Section 3.2.1 by a simple task class priority service discipline. The resulting heuristic policy allocates constant processing times to each member of a task class in accordance with the Markovian heuristic of Subsection 3.2.1. The task class priorities are as follows. If the task losses in the queue are not observable and there is more than one task in the queue when a service ends, then the next task to be served will be a member of the class represented within the tasks currently present with the largest values of the index $r_j\theta_j p_j(\tau_j)/\tau_j$, where τ_j is the allocated constant processing time for a task of class j and $p_j(\tau_j)$ is the probability of successfully service; if task losses in the queue are observable, the denominator of the index is modified to be $E[\min(\mathbf{L}_j,\ \tau_j)]$, where $\mathbf{L}_j$ has an exponential distribution with mean $1/\theta_j$; see Glazebrook et al. [12], who developed the above class index in the context of a simpler model for the optimal service of a collection of impatient tasks that does not incorporate arrivals.

3.2.3. Myopic Policies for Choice of the Next Task Class to Serve and the Amount of Service to Provide

Suppose at some decision epoch, the system state is the number of tasks $\bar{n} = (n_1, n_2, \ldots, n_J)$ of various classes in the queue, with n_j being the number of tasks of class j waiting in the queue. We will choose a class $j(\bar{n})$ from which the next task to be processed should be taken together with an allocated processing time $\tau(\bar{n})$. While it is in principle possible to use DP together with a discretization of the action space to find such policies that are close to optimal, this is unrealistic in problems of reasonable size. Note that the standard theory of stochastic dynamic programming (see, for example, [21]) indicates that there exists an optimal policy for such a formulation that is stationary (makes use of current state information only). Further, the full detail of any optimal policy is likely to be sufficiently complex as to make implementation very difficult. The goal is to heuristically obtain as large a long-run average reward as possible.

As discussed previously, we will consider different possibilities, depending on whether the server observes the completion of service and/or defection of the customer being served.

Suppose first that loss during service and the completion of service are not observable. To see the logic of the proposed policies, suppose we decide to serve a waiting task of class j and allocate $\tau_j > 0$ units of time to serve (process) that task. Recall that the time to task loss (impatience time) is exponential with mean $1/\theta_j$. The expected reward received during service of that customer is

$$R_E(j,\tau_j;\bar{n}) = r_j \int_0^{\tau_j} e^{-\theta_j y} F_j(dy). \tag{3.7}$$

This is, clearly, an increasing function of the allocated service/processing time τ_j. However, the longer the allocated service/processing time is, the more reward we could have obtained by serving alternatively other tasks, some of which may defect during the time τ_j. Notice that the expected reward leaving the system during the service time is

$$R_L(j,\tau_j;\bar{n}) = r_j + r_j(n_j - 1)A_j(\tau_j;\theta_j) + \sum_{k\neq j} r_k n_k A_j(\tau_j;\theta_k) + \sum_{k=1}^{J} r_k \lambda_k B_j(\tau_j;\theta_k), \tag{3.8}$$

where

$$A_j(\tau;\theta) = [1 - e^{-\theta\tau}] \tag{3.9}$$

$$B_j(\tau,\theta) = \tau - \frac{1}{\theta}[1 - \exp\{-\theta\tau\}]. \tag{3.10}$$

Here $A_j(\tau, \theta)$ is the probability that a presently waiting customer with impatience rate θ will defect during the time allocated to serve a customer of class j if τ units of time have been allocated. In the present case, with both service completion and defection during service being unobserved, the subscript j is, of course, redundant. We use this cumbersome notation for ease of comparing with similar policies under other scenarios. Similarly, $B_j(\tau, \theta)$ is proportional to the expected number of customers with impatience rate θ arriving during the time allocated to serve a customer of class j if τ units of time have been allocated that will defect during that service time. Once again, we keep the subscript j for easy comparison with other cases.

The myopic policy is to select the j and τ_j that maximize the proportion of expected reward stream gained during the next service time. That is,

$$(j_0(\bar{n}), \tau_{j_0}(\bar{n})) = \underset{j,\tau_j}{\operatorname{argmax}} \frac{R_E(j,\tau_j;\bar{n})}{R_L(j,\tau_j;\bar{n})}. \tag{3.11}$$

The policy is myopic because it only optimizes the immediate gain.

Similar policies can be obtained for cases in which the service completion is observable and/or the loss of a task during service is observable. The overall logic of the procedures remains the same: we use the maximization in (3.11), but the expected amount of reward leaving system $R_L(j, \tau_j, \bar{n})$ is different under different scenarios.

If the service completion is observable but the loss of a task during service is not observable, then

$$A_j(\tau,\theta) = \int_0^\tau [1 - e^{-\theta y}]F_j(dy) + [1 - F_j(\tau)][1 - e^{-\theta\tau}] \tag{3.12a}$$

$$B_j(\tau,\theta) = \int_0^\tau \left[y - \frac{1}{\theta}[1 - e^{-\theta y}]\right]F_j(dy) + [1 - F_j(\tau)]\left[\tau - \frac{1}{\theta}[1 - e^{-\theta\tau}]\right], \tag{3.12b}$$

where F_j is the service time distribution for a task of class j.

If the service completion is not observable but the loss of a task during service is observable, then

$$A_j(\tau,\theta) = \frac{\theta}{\theta + \theta_j}[1 - e^{-[\theta+\theta_j]\tau}] \tag{3.13a}$$

$$B_j(\tau,\theta) = \frac{1}{\theta_j}[1 - e^{\theta_j\tau}] - \frac{1}{\theta_j + \theta}[1 - e^{-(\theta+\theta_j)\tau}]. \tag{3.13b}$$

If both the service completion and the loss of a task during service are observable, then

$$A_j(\tau,\theta) = \int_0^\tau e^{-\theta_j y}[1 - e^{-\theta y}]F_j(dy) + \int_0^\tau [1 - F_j(y)][1 - e^{-\theta y}]\theta_j e^{-\theta_j y}dy + e^{-\theta_j\tau}[1 - F_j(\tau)][1 - e^{-\theta\tau}] \tag{3.14a}$$

$$B_j(\tau,\theta) = \int_0^\tau \left[y - \frac{1}{\theta}[1 - e^{-\theta y}]\right]e^{-\theta_j y}F_j(dy) + \int_0^\tau \left[y - \frac{1}{\theta}[1 - e^{-\theta y}]\right][1 - F_j(y)]\theta_j e^{-\theta_j y}dy + e^{-\theta_j\tau}[1 - F_j(\tau)]\left[\tau - \frac{1}{\theta}[1 - e^{-\theta\tau}]\right]. \tag{3.14b}$$

Example. There are three task classes. The task classes have gamma service times with different means and shape parameters. The parameters of the gamma distributions are chosen so that if a task starts service, it will complete service successfully with probability 0.9. The times until task loss for task class j are independent and identically distributed having an exponential distribution for $j = 1, 2, 3$. Task losses from the queue are observable. Each simulation run simulates 30,000 tasks of each class for each replication and has 50 replications.

Table 4 displays the mean fraction of the total possible reward earned under various service policies obtained from the simulation [6]. One service policy is the myopic policy of Subsection 3.3.3; another is a FCFS policy that gives full service; a third policy is a FCFS policy that terminates service at constant times specified by the Markovian approximate model of Subsection 3.2.1 with losses during service observable. A fourth policy assigns priority to each task class as described in Subsection 3.2.2 and terminates service at constant times specified by the Markovian approximate model with losses during service observable. Note that the FCFS policy that gives full service is only applicable when service completion is observable.

Discussion. For the cases in which task loss during service and task service completion are not observable, the

Table 4. Mean fraction of total reward earned.

Mean of gamma service time task 1 [task2] (task3)	Shape of gamma service time dist. task 1 [task2] (task3)	Arr ival rate task 1 [task2] (task3)	Reward task 1 [task2] (task3)	Mean time to loss task 1 [task2] (task3)	Observable task loss	Observable service completion	Mean fraction: myopic policy (std. error)	Mean fraction: Markovian approx. FCFS (std. error)	Mean fraction: Markovian approx. with task priority	Mean fraction: FCFS (std. error)
2.24 [3.52] (1.07)	0.9 [0.5] (3)	1/5 [1/5] (1/5)	1 [1] (1)	20 [30] (10)	No	No	0.49 (0.0002)	0.51 (0.0002)	0.53 (0.0002)	NA
2.24 [3.52] (1.07)	0.9 [0.5] (3)	1/5 [1/5] (1/5)	1 [1] (1)	20 [30] (10)	Yes	Yes	0.70 (0.0003)	0.58 (0.0002)	0.58 (0.0002)	0.62 (0.0005)
2.24 [3.52] (1.07)	0.9 [0.5] (3)	[1/5] 1/5 (1/5)	2 [3] (1)	20 [30] (10)	No	No	0.44 (0.0003)	0.48 (0.0003)	0.52 (0.0002)	NA
2.24 [3.52] (1.07)	0.9 [0.5] (3)	1/5 [1/5] (1/5)	2 [3] (1)	20 [30] (10)	Yes	Yes	0.67 (0.0004)	0.55 (0.0003)	0.55 (0.0003)	0.66 (0.0004)
2.24 [3.52] (1.07)	0.9 [0.5] (3)	1/5 [1/5] (1/5)	2 [1] (3)	20 [30] (10)	No	No	0.57 (0.0002)	0.54 (0.0002)	0.58 (0.0002)	NA
2.24 [3.52] (1.07)	0.9 [0.5] (3)	1/5 [1/5] (1/5)	2 [1] (3)	20 [30] (10)	Yes	Yes	0.74 (0.0002)	0.62 (0.0002)	0.65 (0.0002)	0.59 (0.0005)
2.24 [3.52] (1.07)	0.9 [0.5] (3)	1/3 [1/3] (1/3)	2 [1] (3)	20 [30] (10)	No	No	0.47 (0.0003)	0.45 (0.0003)	0.48 (0.0002)	NA
2.24 [3.52] (1.07)	0.9 [0.5] (3)	1/3 [1/3] (1/3)	2 [1] (3)	20 [30] (10)	Yes	Yes	0.64 (0.0002)	0.51 (0.0002)	0.52 (0.0002)	0.35 (0.0006)
2.24 [3.52] (1.07)	0.9 [0.5] (3)	1/3 [1/3] (1/3)	2 [1] (3)	40 [60] (20)	No	No	0.50 (0.0002)	0.49 (0.0003)	0.51 (0.0002)	NA
2.24 [3.52] (1.07)	0.9 [0.5] (3)	1/3 [1/3] (1/3)	2 [1] (3)	40 [60] (20)	Yes	Yes	0.70 (0.0003)	0.53 (0.0002)	0.54 (0.0002)	0.33 (0.0006)

fraction of tasks that are served successfully is roughly the same for the myopic policy and the heuristic policies; the heuristic policy with task priorities yields slightly larger probabilities of successful task completion. In the cases for which the task loss during service and the task service completion are observable, the myopic policy results in the largest probabilities of successful task completion. The advantage of the heuristic Markovian policies is that they are much easier to compute than the myopic policy in these cases; the myopic policy requires numerical integration. Naive FCFS degenerates with higher traffic intensity and with heavy-tailed service time distributions; it is not applicable if service completion is not observable.

4. CONCLUDING REMARKS

Modeling uncertain time-critical service systems is a difficult but vitally important practical problem. Exact computations are often either impossible or very challenging computationally, especially with multiple task classes. Special challenges are present when deciding on a service policy in order to make the system as efficient as possible.

In this paper we have presented several approximation procedures that are computationally easy and, at least in the examples we have looked at, provide valuable information about the efficiency of the service system under different service options. An important feature of these approximations is that they stay computationally feasible even for many task classes and/or heavily loaded systems.

We have introduced a heuristic myopic service policy that attempts to maximize locally the system efficiency. This policy has performed well under scenarios we have considered. We have also introduced a Markovian heuristic that performs well when service completion and task loss during service are not observed.

A number of important issues are left for future work. One such issue is improving the myopic policies into (approximately) optimal policies. Another untouched issue is that of nonstationarity: what happens if the parameters of the system change with time and need to be constantly estimated in order to update the service policy and keep the system running efficiently. We hope to address these questions in the near future.

ACKNOWLEDGMENTS

The authors acknowledge useful comments by Gideon Weiss, John Hiles, Glen Takahara, and John Lehoczky. John Lehoczky suggested Approximation II. Professor Glazebrook acknowledges support received from the Engineering and Physical Sciences Research Council through the award of Grant GR/S45188/01. Professor Samorodnitsky acknowledges support from the National Research Council Associateship Program at the Naval Postgraduate School and National Science Foundation grant DMS-0303493. Professors Gaver and Jacobs acknowledge support from the Center for Defense Technology and Education for the Military Services (CDTEMS) at the Naval Postgraduate School. We acknowledge with thanks the contributions of Dr. Gregory Bullock.

REFERENCES

[1] F. Baccelli, P. Boyer, and G. Hebuterne, Single-server queues with impatient customers, Adv Appl Probab 16 (1984), 887–905.

[2] K.J. Becker, D.P. Gaver, K.D. Glazebrook, P.A. Jacobs, and S. Lawphongpanich, Allocation of tasks to specialized processors: A planning approach, Eur J Oper Res 126 (2000), 80–88.

[3] N.K. Boots and H. Tijms, A multi-server queueing system with impatient customers, Manage Sci 45(3) (1999), 444–448.

[4] N.K. Boots and H. Tijms, An M/M/C queue with impatient customers, presented at the First International Workshop on Retrial Queues, Universidad Complutense de Madrid, Madrid, Spain, September 22–24, 1998.

[5] M. Brown and F. Proschan, Imperfect repair, J Appl Probab 20 (1983), 851–859.

[6] G.L. Bullock, UTCT Simulation Model, A simulation implemented in C++, 2004.

[7] D.R. Cox and W.L. Smith, Queues, Chapman & Hall, London, UK, 1961.

[8] B. Doytchinov, L. Lehoczky, and S. Shreve, Real-time queues in heavy traffic with earliest-deadline-first queue discipline, Ann Appl Probab 11 (2001), 332–378.

[9] W. Feller, An Introduction to Probability Theory and Its Applications, Vol. II, Wiley, New York, 1966.

[10] D.P. Gaver and P.A. Jacobs, Waiting times when service times are stable laws: Tamed and wild, in Applied Probability and Stochastic Processes, J.G. Shanthikumar and U. Sumita (Editors), Vol. 19 in International Series in Operations Research and Management Science, Kluwer Academic, Boston, 1999.

[11] D.P. Gaver and P.A. Jacobs, Servicing impatient tasks that have uncertain outcomes, Naval Postgraduate School Technical Report, NPS-OR-00-001, Monterey, CA, 2000.

[12] K.D. Glazebrook, P.S. Ansell, R.T. Dunn, and R.R. Lumley, On the optimal allocation of service to impatient tasks, J Appl Probab 41 (2004), 51–72.

[13] Z. Jiang, T.G. Lewis, and J.-Y., Colin, Scheduling hard real-time constrained periodic tasks on multiple processors, J Syst Software 19(11) (1996), 102–118.

[14] L. Kleinrock, Queueing Systems, Vol. I: Theory, Wiley (Interscience), New York, 1976.

[15] J.P. Lehoczky, Real-time queueing theory, in Proceedings of the IEEE Real-Time Systems Symposium, December 1996, pp. 186–195.

[16] J.P. Lehoczky, Using real-time queueing theory to control lateness in real-time systems, Perform Eval Rev 25(1) (1997), 158–168.

[17] J.P. Lehoczky, Real-time queueing network theory, Proceedings of the IEEE Real-Time Systems Symposium, December 1997, pp. 58–67.

[18] C.L. Liu and J.W. Layland, Scheduling algorithms for multiprogramming in a hard real-time environment, J Auto Comput Mach 20(1) (1973), 40–61.

[19] J. Osmundson, A systems engineering methodology for information systems, Syst Eng 3(2) (2000), 68–81.

[20] C.H. Papadimitriou and J. Tsitsiklis, The complexity of queueing network control, Math Oper Res 24(2) (1999), 293–305.

[21] M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, Wiley, New York, 1994.

[22] S.I. Resnick, Extreme Values, Regular Variation, and Point Processes, Springer-Verlag, New York, 1987.

[23] R. Righter, The stochastic sequential assignment problem with random deadlines, Probab Eng Inform Sci 1 (1987), 189–202.

[24] R. Righter, Expulsion and scheduling control for multiclass queues with heterogeneous servers, Queueing Syst Theory Appl 34 (2000), 289–300.

[25] A.R. Ward and N. Bambos, On stability of queueing networks with job deadlines, J Appl Probab 40(2) (2003), 293–304.

[26] W. Whitt, Improving service by informing customers about anticipated delays, Manage Sci 45(2) (1999), 192–207.

[27] P. Whittle, Restless bandits: Activity allocation in a changing world, In a Celebration of Applied Probability (J Appl Prob Spec Vol 25A), J. Gani (Editor), Applied Probability Trust, Sheffield, 1988, pp. 287–298.