MANAGEMENT SCIENCE
Vol. 53, No. 1, January 2007, pp. 61–77
ISSN 0025-1909 | EISSN 1526-5501 | 07 | 5301 | 0061

informs

DOI 10.1287/mnsc.1060.0598

Operations Systems with Discretionary Task Completion

Wallace J. Hopp, Seyed M. R. Iravani, Gigi Y. Yuen
Department of Industrial Engineering and Management Sciences, Northwestern University,
2145 Sheridan Road, Evanston, Illinois 60208-3119
{hopp@northwestern.edu, s-iravani@northwestern.edu, gigiyuen@northwestern.edu}

Most performance evaluation models in the operations management literature implicitly assume that tasks possess standardized completion criteria. However, in many systems, particularly service and professional work, judgment is frequently required to determine how much time to allocate to a task. In this paper, we show that introducing discretion in task completion adds a fourth variability buffer, quality, to the well-known buffers of capacity, inventory and time. To gain insight into the managerial implications of this difference, we model the work of one- and two-worker systems with discretionary task completion as controlled queues. After characterizing the optimal control policy and identifying some practical heuristics, we use this model to examine the differences between discretionary and nondiscretionary work. We show that in systems with discretionary task completion, (i) adding capacity may actually increase congestion, and (ii) task variability in service time can improve system performance. This implies that it may be suboptimal to expect shorter delays as a result of a capacity increase, and that task variability reduction may not be an appropriate goal in systems with discretionary task completion. We also find that the benefit of queue pooling is smaller in systems with discretionary task completion than in systems with nondiscretionary task completion.

Key words: service operations; work systems; white collar work; flexibility; Markov decision process
History: Accepted by Candace A. Yano, operations and supply chain management; received January 25, 2005.
This paper was with the authors 8 months for 2 revisions.

1. Introduction

Since the 1980s, the American economy has steadily shifted toward services, with 34.2%, 37.1%, and 42.7% of the national gross domestic product (GDP) coming from the service sector[1] in 1987, 1994, and 2001, respectively. The manufacturing sector[2] constituted 25.2%, 22.5%, and 20.3% of national GDP in these same years (Bureau of Economic Analysis 2001). This trend is being accompanied by an increase in the number and size of service-oriented firms. Moreover, even within manufacturing organizations, there is an increasing emphasis on professional work due to the proliferation of lean manufacturing and outsourcing practices. For example, in 1972, semiskilled operators (e.g., line workers) represented 55% of the General Motors workforce, while nonmanagement professionals (e.g., engineers) represented 5%. By 2001, operators represented only 44%, while professionals had increased to 14% (General Motors 1974, 2002).

A large portion of research on operations systems has focused explicitly or implicitly on manufacturing oriented work systems. Since the early work of Frederick W. Taylor at the beginning of the 20th century, in papers ranging from time and motion studies to the modeling of production flows, industrial engineers have concentrated on developing better methods for managing manufacturing systems. Considerably less research has been devoted to service systems and very little industrial engineering work has systematically studied professional work.

A virtually universal assumption in past models of manufacturing work systems is that they involve well-defined tasks with *nondiscretionary* completion criteria (i.e., determined by objective standards). However, in practice, many operations systems involve less-defined tasks with *discretionary* completion criteria (i.e., determined by a worker's subjective standards). In particular, service and professional systems frequently require workers' judgment concerning when tasks are completed. Discretionary task completion introduces a degree of flexibility into process times, which allows a worker to adjust quality of his/her output to manage workload.

Traditional manufacturing work and routine service work (e.g., bank tellers, checkout clerks) tend to have *nondiscretionary task completion* (NDTC) characteristics, while professional work and complex service

[1] Includes sectors of finance, insurance, real estate, and services.

[2] Includes sectors of mining, construction, and manufacturing.

work (e.g., engineers, physicians, software developers, financial analysts) tend to have *discretionary task completion* (DTC). For example, consider the task of installing a car seat into a sedan on an assembly line. The worker has to follow a standardized procedure to secure the seat to the car frame. The output quality of the work has well-defined metrics (e.g., whether the seat is secure) and management can easily specify the completion standard. Hence, this task has NDTC features. Various service industry tasks, such as transferring money between bank accounts, follow similarly well-defined structures, and are hence also regarded as NDTC work in our terminology.

In contrast, consider a call-center agent of a mail-order catalog company who processes orders and is also tasked with upselling (i.e., persuading customers to buy additional or more expensive items; see Aydin and Ziya 2005, Armony and Gurvich 2005, Aksin and Harker 1999). Beyond the minimum amount of time required to perform the basic order processing, the agent can choose the number of products to promote and the corresponding upselling strategy. As such, the agent has considerable influence over the call duration. Moreover, in such a service environment, it is nearly impossible to enforce a clear and objective completion standard. Hence, we regard this task as DTC work in our terminology. For a review of call-center research, see Gans et al. (2003).

Another example of DTC work occurs in a system managed by our industry collaborator General Motors. The engineers in the warranty repair group are given assignments to resolve problems that result in warranty costs. The solution process typically involves two phases: diagnostics and remediation. The more time an engineer spends diagnosing the problem, the more likely he/she is to identify an effective solution. Given the problem complexity, it is impossible to specify a priori task termination criteria. Hence, engineers are generally allowed a high degree of autonomy on how much time to spend on each problem. So again, the problem-solving tasks constitute DTC work.

One of the primary challenges in managing manufacturing and service systems is how to accommodate and mitigate variability. Process time variability is inherent in virtually all systems, whether they involve DTC or NDTC work. In manufacturing systems, this variability degrades system performance because of a well-known principle of factory physics which states that variability must be buffered by some combination of capacity, inventory, and time (Hopp and Spearman 2000). In service systems, only two of these variability buffers, capacity and time, are available because services cannot be stored. For example, variability in arrivals to a call center causes some mixture of queueing delay (time buffer) and idle agents (capacity buffer).

In this paper, we show that discretionary task completion introduces *quality* as an additional factor for buffering variability in DTC systems. To gain insights into the managerial implications of this newly introduced variability buffer, we examine one-server and two-server work systems with discretionary task completions. We first characterize the optimal control policy which specifies how long the worker should spend on each task in a single-job class system. We also introduce two simple threshold heuristics and show that they can be effective under certain conditions. Next, we compare our discretionary completion models with analogous nondiscretionary completion models in three different design and control aspects: (i) capacity, (ii) queue pooling, and (iii) input task variability. This leads us to two interesting and counterintuitive phenomena, which are distinctive to systems with discretionary task completion. In contrast to well-known NDTC system behavior, in DTC systems, increasing capacity can intensify congestion and adding input task variability may improve system performance. We also show that the benefit gained from pooling multiple queues in a two-server system decreases with the degree of discretion in completion times.

2. Literature Review

Extensive analytical research has been devoted to the design and control of NDTC work systems. This work covers a wide range of topics, including work flow design, production scheduling, and inventory management (see, e.g., Buzacott and Shanthikumar 1992, Altiok 1996, Hopp and Spearman 2000, Askin and Goldberg 2001).

Researchers have also conducted empirical and experimental studies to understand the impacts of human behavior on work systems (see Bailey 1998, Banker et al. 2001, Doerr et al. 1996). For example, Schultz et al. (1998) conducted laboratory experiments to investigate how motivation affects worker processing time. A survey and discussion of research at the interface of operations management and human resources management is given in Boudreau et al. (2003).

Many of the principles that have come out of the above research implicitly or explicitly focus on systems in which tasks are routine and well defined. Hence, they are applicable to what we term "NDTC settings." Some researchers have attempted to extend these NDTC insights to DTC environments in applications ranging from large-scale reengineering of a corporation (Hammer and Champy 1993) to small-scale process improvement of a product development process (Adler et al. 1995, Krishnan et al. 1997, Loch and Terwiesch 1999). For instance, Adler et al. (1995) suggests thinking of product development as a process

in which projects move through the knowledge-work equivalent of a job shop, while Loch and Terwiesch (1999) argue that the product development process has many manufacturing-like activities. Although this work has unquestionably yielded useful results, it is limited by the implicit assumptions of NDTC characteristics.

Some recent modeling work has been done specifically with discretionary task completion in mind. This can be classified in two major streams: (i) models that allow adjusting service rate to dynamically control queueing systems, and (ii) models that represent the relationship between dynamic pricing and adjustable service rate.

The first stream of research focuses on the characterization and computation of optimal policies and provides limited managerial insights. The two papers most closely related to our work are Stidham and Weber (1989) and George and Harrison (2001), which both show that under an optimal policy, service rate is nondecreasing in queue length. Both papers allow the service rate to be changed only at job arrival or departure epochs. In contrast, this paper allows service time to be adjusted at any time during processing and hence gives more flexibility to the server. This is realistic in many settings because servers often do not commit to a particular time of completion at the beginning of each service.

The second stream of research studies systems in which servers determine the optimal combination of pricing and service rate. This work includes Debo et al. (2004) and Ata and Shneorson (2006) and provides interesting results on interactions between service providers and customers. However, it does not explore the implications of discretionary task completion for system design or worker management.

The only paper of which we are aware that addresses DTC work from the perspective of generating managerial principles is Owen and Jordan (2003). They propose an S-curve model that describes the time dependence of output quality in white-collar work systems. Using simulation, they examine the performance of various scheduling policies in managing jobs with due dates for a single-server system.

In this paper, we adopt an optimal control approach as a step toward developing analytic principles of DTC workforce management. In §3, we define a time-dependent revenue curve to capture discretionary task completion characteristics of DTC tasks. In §4, we study a single-station, single-job class model of a discretionary task completion system, and derive the structure of the optimal control policy. We also conduct a series of numerical studies that examines the efficiency and robustness of two practical heuristic policies. In §5, we investigate the impact of discretionary completion times on how systems respond to an increase in capacity. In §6, we introduce a two-server, single-job class model and use it to study the effect of queue pooling in multiserver systems. In §7, we use a two-task class model to introduce task variability. We examine the interactions between task variability and discretion in DTC systems. Concluding remarks are presented in §8.

3. Discretionary vs. Nondiscretionary Task Completion

In a manufacturing or routine service environment, output is typically measured as the rate of task completions (e.g., jobs per day or customers per hour). The reason is that the work content of tasks is known (at least in expectation). For example, consider a technician who assembles bicycles in a bike shop. Suppose that the difference in price between an assembled and an unassembled bicycle is b and the expected assembly time is τ_0. Then, because the assembly fee cannot be collected until all components have been assembled and adjusted, the expected value to the shop as a function of time looks like Figure 1(a). For our purposes, a step-function value-time curve defines NDTC. It is precisely the fact that the technician has no discretion over the value added to the bicycle—either it is assembled or it is not—that causes the value-time curve to resemble Figure 1(a).

However, the completion criteria in some tasks are not so clear. For example, in the previously described call-center upselling example, a call can be made short or long at the agent's discretion. Assuming that the agent makes rational use of call time, the expected revenue generated will be an increasing function of time. However, because the total revenue obtainable from a caller is finite, the value versus the time curve must be eventually concave. While many forms are possible, the simplest relationship between processing time and expected value is a concave curve, as depicted in Figure 1(b). Because a curve like this presents the agent with a decision of when to terminate processing, we refer to this as DTC.

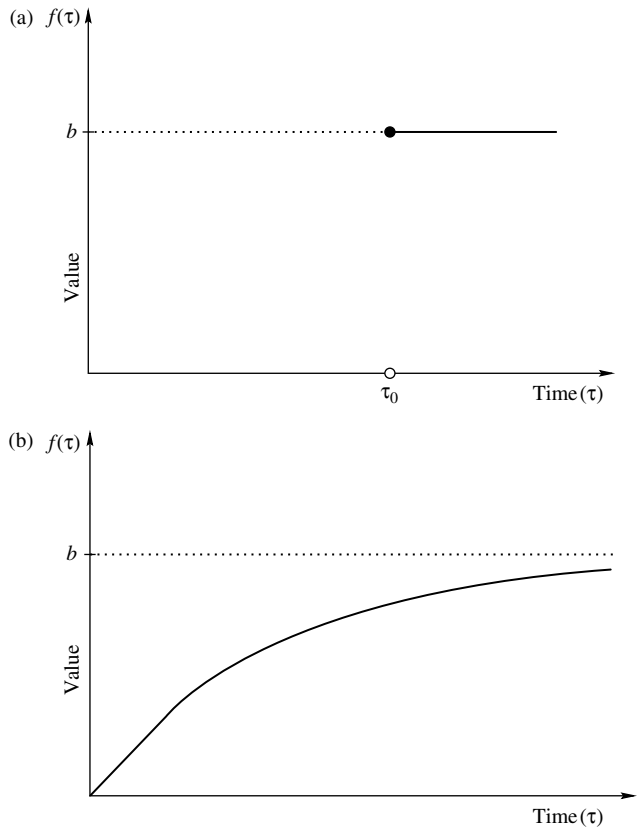
A simple way to model the concave curve in Figure 1(b) is the exponential function

$$f(\tau) = b(1 - e^{-a\tau}). \tag{1}$$

The parameter a determines the rate of increase of the value function $f(\tau)$. The larger a is, the faster $f(\tau)$ approaches its upper limit, b. Hence, a characterizes the processing rate, while b represents the highest possible task value.

In practice, it may be difficult to estimate the value-time curve. Nevertheless, in high volume systems such as call centers, it may be feasible to determine the general shape and use a simple family, such as the exponential, as a good approximation. If so, using

Figure 1 (a) Value vs. Time for Nondiscretionary Completion Tasks; (b) Value vs. Time for Discretionary Completion Tasks



data on the proportion of customers that are satisfied within x minutes is sufficient to estimate a, while b can be set to reflect the relative value of different call types. As we discuss in the remainder of this paper, even if a curve cannot be estimated in practice, we can still draw basic insights into how DTC workers should behave and how DTC work differs from NDTC work by using such a curve to describe the underlying dynamics of a DTC system.

4. Control Policy

To develop a model of work with discretionary task completion, we consider a worker who staffs a single station that receives jobs that arrive according to a Poisson process with rate λ. Jobs have the DTC behavior depicted in Figure 1(b) and are processed in first-come-first-served (FCFS) order. A holding cost h per unit time is charged for each task while it is in the system. The value generated by a task is $f(\tau)$, where τ is the service time chosen by the worker and $f(\tau)$ is given in Equation (1). To guarantee a minimum amount of time spent on a task (e.g., customer welcome, setup, minimum response, etc.), we assume that $\tau_{\min}$ represents the smallest allowable processing time. Note that increasing $\tau_{\min}$ restricts the ability of the worker to adjust the service time of each task.

In our call-center example, this model represents an agent who is responsible for customers that require a minimum of $\tau_{\min}$ service time to process a purchase. Beyond this minimum time, the agent can choose to spend additional time talking to the customer in hopes that they will buy a more expensive version of the item or other related items. The expected revenue generated from upselling is represented by the value-time curve. The holding cost reflects the cost of the inbound 1-800 call plus possibly a loss of customer goodwill, measurable as loss of future sales, caused by extending the call.

In our warranty repair example, this model represents an engineer who receives assignments to resolve problems with vehicles in the field. Each problem requires a minimum expected amount of $\tau_{\min}$ time to conduct an analysis and propose a solution. By spending additional time, the engineer can consider more candidate solutions and find one that more effectively reduces warranty costs. The cost saving generated by spending additional time to figure out a better design is represented by the value-time curve, while warranty cost from an unresolved problem awaiting attention represents a holding cost.

In this section, our goal is to find a task completion policy that maximizes the value generated per unit of time. In the upselling example, this means finding how much time the call-center agent should spend upselling to the current caller given knowledge of the number of customers waiting in queue. In the warranty repair example, this means identifying how much time the engineer should spend working on the current problem given the knowledge of the number of pending problems.

4.1. Model Formulation

To develop an optimization model, we discretize the time horizon into equal, nonoverlapping, infinitesimal intervals $\delta\tau$, where $\delta\tau$ is small enough to ensure that the probability of having more than one arrival during $\delta\tau$ is almost zero. Thus, the probability of having one arrival during interval $\delta\tau$ is $\lambda\delta\tau$. We define $t_{\min} = \tau_{\min}/\delta\tau$. We can then formulate the problem as a Markov decision process (MDP) in which:

- *State space* $\mathcal{S}$ includes states (n, t), where n is the number of jobs in the system, and t is the number of time intervals (i.e., number of periods of length $\delta\tau$) that the job under service has been worked on. Both n and t are nonnegative integers.
- *Decision epochs* are the beginning of every period.
- *Action space* $\mathcal{A}$ includes actions *keep* and *release*.

Action "keep" dictates that the worker should continue working on the current job for one more period, while action "release" calls for the worker to stop working on the current job and release it.

Without loss of generality, we assume that the worker can release a job at the beginning of a period, but jobs arrive only at the end of a period. We define $V(n,t)$ as the value function in state (n,t) and write the optimality equation of the MDP model for $n \geq 1$ and $t \geq t_{\min}+1$ as

$$\delta\tau g + V(n,t) = \max \begin{cases} -nh\delta\tau + (1-\lambda\delta\tau)V(n,t+1) \\ \qquad + \lambda\delta\tau V(n+1,t+1)\text{: keep,} \\ -(n-1)h\delta\tau + f(t\delta\tau) + (1-\lambda\delta\tau)V(n-1,\mathbf{1}_n) \\ \qquad + \lambda\delta\tau V(n,\mathbf{1}_n)\text{: release,} \end{cases}$$

where g is the optimal average profit per unit time, and $\mathbf{1}_n = 0$ if $n = 1$ and $\mathbf{1}_n = 1$ if $n \geq 2$. For $n \geq 1$ and $t \leq t_{\min}$, we have

$$\delta\tau g + V(n,t) = -nh\delta\tau + (1-\lambda\delta\tau)V(n,t+1) + \lambda\delta\tau V(n+1,t+1),$$

and for $n = 0$,

$$\delta\tau g + V(0,0) = (1-\lambda\delta\tau)V(0,0) + \lambda\delta\tau V(1,0).$$

4.2. Optimal Control Policy

Because task completion time is determined by the worker, utilization in a DTC system is also controlled by the worker. However, because of our assumption that all jobs must receive at least minimal processing, we define $\rho_{\min} = \lambda\tau_{\min}$. It is clear that the system is only stable when $\rho_{\min} < 1$.

The structure of the optimal service policy in a stable system is characterized by Theorems 1 and 2. For ease of presentation, we omit $\delta\tau$ in the value function $f(t\delta\tau)$. Therefore, in the rest of this paper, $f(t)$ represents the value when the job is released after $t\delta\tau$ units of time. Omitted proofs can be found in Appendix 5; all appendices are available in the e-companion.[3]

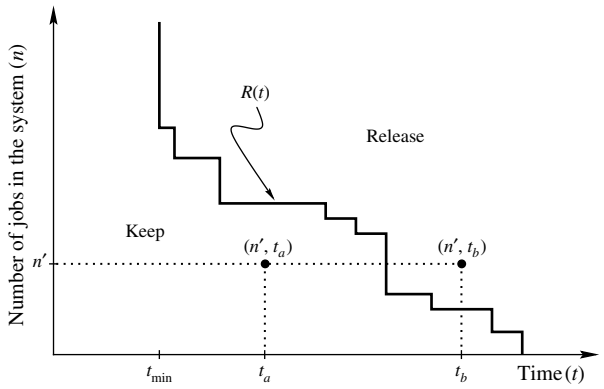
THEOREM 1. *If $f(t) < h\delta t$ for all t, then the optimal policy is to spend the minimum amount of time ($t_{\min}$) on each job.*

THEOREM 2. *If $f(t) > \delta ht$ for some t, then there exists a nonincreasing threshold function $n = R(t)$ such that it is optimal to keep the job in state (n,t) if $n \leq R(t)$ and release the job if $n > R(t)$.*

Figure 2 shows the typical structure of the optimal service policy. The boundary that separates the "keep" region from the "release" region is composed of state-dependent thresholds that define the optimal decision in different states. For example, in state

[3] The e-companion to this paper, which is part of the online version, is available at http://mansci.pubs.informs.org/.

Figure 2 Typical Structure of a DTC Worker's Optimal Policy



(n', t_a) where $t_{\min} \leq t_a$, the current job has already been processed for t_a units of time. In this state, because the potential value generated by working on the job for one more period outweighs the holding cost of keeping n' jobs in the system, it is optimal to keep the job. On the other hand, in state (n', t_b), because the potential benefit of spending one more period processing the current job does not exceed the holding cost of keeping n' jobs in the system, the optimal decision is to release the current job.

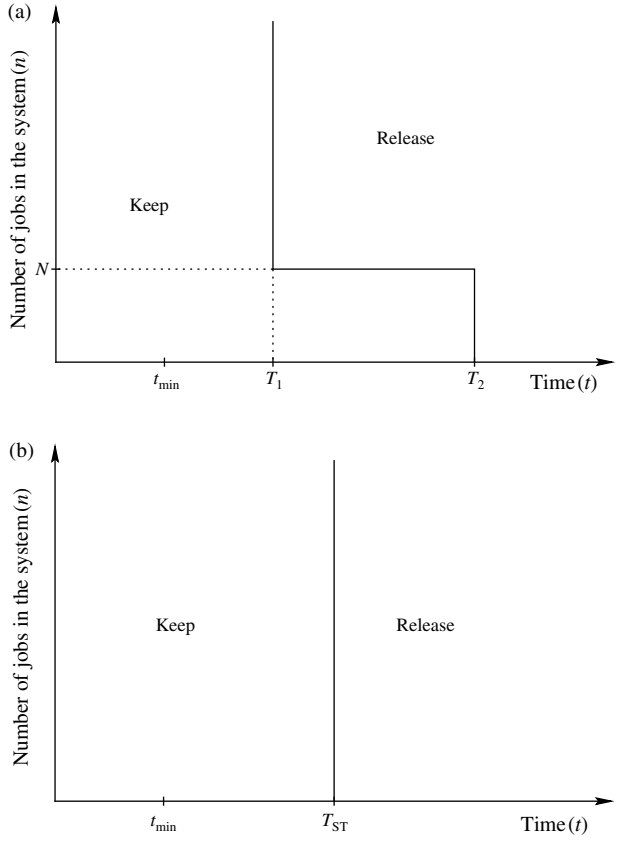
The structure of the optimal control policy demonstrates how DTC systems mitigate congestion by varying the quality (or revenue) of outputs. When the queue is long, the system lowers product quality by processing jobs for a shorter amount of time. This helps the system reduce congestion and keeps the holding cost low. In a warranty repair group, this implies that an engineer tends to limit the amount of time spent diagnosing a problem when the number of repair requests is large to prevent the queue from growing too long. Because adjusting the completion time has a direct impact on the output quality level, we describe this behavior as using quality as a variability buffer.

In a call center, our results imply that during peak demand times an agent is likely to conduct less upselling (via shorter calls) than during low demand times to keep the customer waiting time reasonable. In this case, adjusting the completion time has a direct impact on revenue generation, so revenue is the variability buffer. However, because in a majority of service and professional work the link between process time and revenue is not this direct, we use "quality" to describe the new variability buffer introduced by DTC work.

4.3. Performance of Heuristics Policies

The optimal policy in Figure 2 presents two major difficulties: (i) it has a complex structure even for our simple case; for real-world systems, the optimal policy would be at least as complex; and (ii) jobs

Figure 3 **(a) The Double-Threshold (DT) Policy; (b) The Single-Threshold (ST) Policy**



are released at different points of time during their processing, resulting in inconsistent quality among customers which could erode long-term customer satisfaction. These issues motivated us to consider heuristic policies that may be better suited to practice.

The first heuristic is called the *double-threshold* (DT) *policy* and has its structure illustrated in Figure 3(a). The DT policy has three parameters (N, T_1, T_2), where $T_2 \geq T_1 \geq t_{\min}$. Under this policy, the server works on the current job for T_2 units of time as long as there are less than or equal to N jobs in the system. Otherwise, the current job is processed for T_1 units of time. For example, in a warranty repair group under the DT policy, depending on the number of pending problems, engineers can choose between a minimal or an extended solution procedure.

The second heuristic, illustrated in Figure 3(b), is called the *single-threshold* (ST) *policy*, and is a special case of the DT policy for which $T_{ST} = T_1 = T_2 \geq t_{\min}$. Under this policy, and all jobs are processed for the same amount of time, T_{ST}. Hence, the ST heuristic is similar to the policy for NDTC work systems in which processing times are independent of the work backlog. The worker selects a fixed processing time in the beginning of the horizon and processes all jobs according to that. Hence, there is discretion over the completion time of a task for every job but the worker chooses not to use it. For example, the warranty repair engineer might choose to spend a standard amount of time solving a problem regardless of the number of pending requests.

To describe the conditions under which the DT and ST heuristics perform well, we studied the performance of these heuristics relative to that of the optimal policy in various systems. We used a value iteration algorithm for each case to obtain the optimal profit, as well as the profit under the DT and the ST heuristics. We define $\tau_{\max} = \arg_\tau\{f(\tau) = 0.99b\}$ and $t_{\max} = \tau_{\max}\delta\tau$. Therefore, $t_{\max}$ is the maximum number of time intervals the worker is allowed to spend on a job, which is also the time required to obtain 99% of the maximum profit (we use 99% instead of 100% because of the asymptotic property of the exponential function). The parameters we investigated are:

- *Shape of value function A.* Define A as

$$A = \frac{F(\tau_{\max}) - \tau_{\max}(b/2)}{\tau_{\max}(b/2)},$$

where $F(x) = \int_0^x f(\tau)\,d\tau$. This measure characterizes the shape of the value function. As illustrated in Figure 4, the A parameter represents the ratio of the shaded area and the area under the triangle $[(0, 0), (0, \tau_{\max}), (0.99b, \tau_{\max})]$. The bold line represents the value-time function taken into consideration, $\tau_{\min}$. At $A = 0$, the value-time curve is linear. As A increases, value increases more rapidly with time. We considered values $A = 0.24$, 0.59, and 0.79 to represent situations ranging from strongly DTC work (a fairly flat concave function) to nearly NDTC work (very close to a step function).

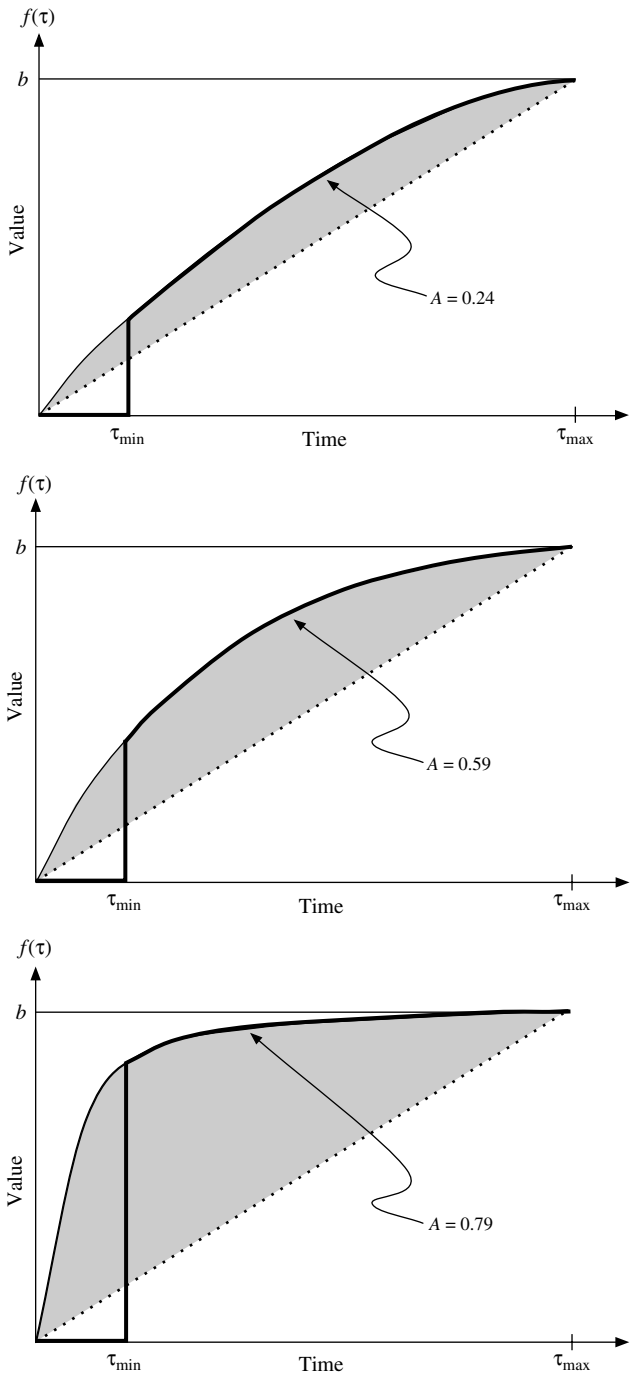
- *Minimum traffic intensity* $\rho_{\min}$. The parameter $\rho_{\min}$ is a relative measure of the minimum level of system congestion. We considered values from 0.1 to 0.55 in increments of approximately 0.15, depending on A and $f(\tau_{\min})$.

- *Holding cost to maximum value ratio* h/b. This ratio characterizes holding cost relative to (maximum) job value. We considered values of 0.01, 0.02, and 0.04. A small ratio (i.e., 0.01) represents situations where holding cost generated from a long wait is small compared to the value gained from the best possible service.

- *Minimum quality* $f(\tau_{\min})/b$. This ratio measures the minimum quality a customer is willing to accept relative to the maximum quality possible. We considered values of 20%, 40%, and 60%.

- *Process flexibility* ϕ. Define $\phi = (\tau_{\max} - \tau_{\min})/\tau_{\max}$. This measure represents the amount of discretion a worker has in choosing processing time. We considered values ranging from 0.6 to 1 in increments of 0.2, depending on A and $f(\tau_{\min})$. The larger this value is, the more discretion a worker has in selecting processing time.

Figure 4 Value-Time Functions with Different Values of *A*



For computational purposes, the queue length was truncated at $N = 30$ after determining by experiment that increasing N beyond 30 does not have a significant effect on the average profit. We set the unit time $\delta\tau$ such that the probability of more than one arrival during a period of length $\delta\tau$ is less than 0.0025. Hence, $\delta\tau = 0.64$ when the arrival rate $\lambda = 0.1$; $\delta\tau = 0.32$ when the arrival rate $\lambda = 0.2$, and so on. The number of time units T was truncated at τ_{max}, which corresponds to the time when the task reaches 99% of its maximum value b. For example, when $A = 0.79$ and $\lambda = 0.2$, it is straightforward to compute $\tau_{max} = 6.2$, and hence $T = 6.2/0.32 = 20$. By repeating this process for different values of (N, T_1, T_2) and T_{ST}, we obtained the profit under the *best* DT and ST policies.

Our numerical study of 108 cases showed that the DT heuristic performs well under most parameter settings. We observed that the DT heuristic results in an average of only 0.5% less profit per unit of time than the optimal policy. The maximum difference we observed was 2%. The DT heuristic is very close to optimal under most conditions because it is able to adjust the quality level to respond to changes in the work backlog. Evidently, even the crude adjustment afforded by only two process time settings is enough to take advantage of quality flexibility. The actual parameters of the DT heuristics in practice can be determined based on experience and historic data. Then, the exact levels of the "regular" and "hurry up" mode can be fine tuned using trial and error.

The ST heuristic, however, performs well under a more restricted range of parameter settings. Our numerical study showed that the ST heuristic results in an average of 5% less profit per unit time than the optimal policy, with a maximum percent difference of around 18%. We found that the ST heuristic performs well in: (i) systems with high A (which represent work systems that approach their maximum value quickly), (ii) systems with small h/b (where holding cost is not significant compared to the potential value earned), (iii) systems with small process flexibility (ϕ), and (iv) systems with fairly low minimum traffic intensity (ρ_{min}). One of the key factors that affects performance of the ST heuristic is the shape of the value-time function $f(\tau)$. The closer $f(\tau)$ is to a step function (e.g., large A and small process flexibility ϕ), the better the ST heuristic performs. Given its simplicity, ST heuristics is a very reasonable policy to adopt when the tasks are fairly similar to NDTC tasks (i.e., servers have limited discretion on time completion). We summarize the results of these tests in the following observations.

Observation 1. The DT heuristic performs almost as well as the optimal service policy in single-station single-class DTC systems.

Observation 2. The ST heuristic performs reasonably well in DTC systems only when the tasks are fairly similar to NDTC tasks.

The difference in process time control between the heuristic policies and the optimal policy results in different levels of quality variation. Workers under the optimal policy have the highest variation in task processing time because it can range from t_{min} to t_{max} depending on what is the best release time based on queue length. The average coefficient of variation (CV) of service time we observed in our tests was 0.24.

In contrast, the ST policy imposes absolute control on processing time and results in no quality variation at all. The DT policy has an intermediate level of quality variation. Its average coefficient of variation of service time was 0.20 in our tests. If such variation affects customer perceptions of fairness, and thereby has a long-term effect on profitability, the DT heuristic might actually be preferred to the optimal policy in practice.

5. Capacity

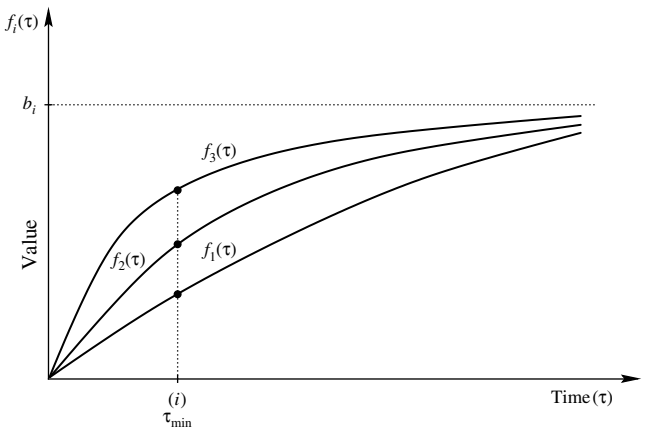
It is well known that increasing system capacity reduces congestion in NDTC environments (Hopp and Spearman 2000, Chapter 9). One might reasonably expect the same to be true in DTC systems. However, because DTC systems introduce an additional variability buffer (quality), we need to look closely at the factors that affect system congestion to understand the role of capacity. In this section, we perform numerical experiments to investigate the question: Does a capacity increase in a DTC work system always decrease average queue length?

In single-worker NDTC systems, an increase in capacity corresponds to a decrease in the average effective processing time of a job. In DTC systems, because capacity is defined by the value-time curve, an increase in capacity can be measured in various ways. In the following section, we investigate two types of capacity increase: (i) improved worker productivity, and (ii) increased number of staff.

5.1. Improving Worker Productivity

We define an increase in capacity via improving worker productivity as an upward shift in the value-time curve (see Figure 5). That is, the new curve (f_2) completely dominates the old one (f_1) because at every point in time, the value earned from working on a job with f_2 is greater than (or equal to) the value earned from working on a job with f_1. Formally, $f_1(\tau) \le f_2(\tau) \le f_3(\tau)$ for all τ. For an exponential value-time function, an increase in capacity can be represented by an increase in the parameter a. In practice, such an increase in productivity could be the result of training.

Figure 5 Modeling an Increase in Capacity via Improved Worker Productivity



To investigate the effects of increasing capacity via improving productivity, we conducted another set of numerical tests using an exponential value-time function with values of $a = 0.25$, 0.5, and 1, which refer to as slow, medium, and fast workers, respectively. To isolate the effect of capacity, we considered value-time functions with the same $\tau^i_{\min}$ and b_i for $i = 1, 2, 3$. We used MDP to compute the optimal service policy for single-station DTC systems with the various capacities. We used the convergence requirement and truncated state space described in §4. To calculate the average queue length, we fed the optimal policy back into the MDP as the decision structure but set $f(\tau) = 0$ for all t, and $h = 1$. Hence, the average profit obtained from the second value iteration returned the average queue length.

Table 1 presents the results of our experiments and illustrates some surprising behavior. In sharp contrast to the behavior of NDTC work systems, an increase in capacity in a DTC work system sometimes results in an increase in average queue length (and hence average waiting time). Table 1 shows that this counterintuitive phenomenon is more prevalent in systems with: (i) high minimum traffic intensity, $\rho_{\min}$, and (ii) relatively low holding cost, h/b. Hence, we can make the following observation.

Observation 3. In DTC systems, increasing capacity may intensify congestion.

The underlying reason for this behavior is the presence of quality as a variability buffer in discretionary task completion systems. This makes it possible to take advantage of an increase in capacity to either reduce queueing (and hence holding cost), or increase average quality level (and hence value), or both. The optimal mix of these depends on the relative costs. When the ratio of holding cost to value (h/b) is high, it makes sense to apply the additional capacity to reduce congestion. But when h/b is low, the extra capacity should be used to improve quality. In some cases, it can even make sense to increase queueing in response to a capacity increase to facilitate an even greater increase in quality.

By comparing the last two columns of Table 1, we observe that the increase in congestion is more significant when capacity is increased from low to medium ($a = 0.25$ to 0.5) than when it is increased from low to high ($a = 0.25$ to 1.0). The explanation for this is that because the value-time function is concave, using capacity to increase value exhibits diminishing returns. Hence, as the capacity increase grows larger,

Table 1 The Effects of Increased Capacity on Queue Length in DTC Systems

Minimum traffic intensity ρ_{min}	Holding cost ratio h/b	Rate parameter a	Average profit rate	Average service time	Utilization	Expected queue length	% change in E[queue] Compared with $a = 0.25$	% change in E[queue] Compared with $a = 0.5$
0.3	0.02	0.25	12.6	2.85	0.98	1.35	—	—
		0.5	19.5	2.76	0.92	1.49	**11**	—
		1	25.0	2.49	0.81	1.26	−6	−16
	0.04	0.25	10.3	2.63	0.92	1.05	—	—
		0.5	16.9	2.49	0.84	1.11	**6**	—
		1	22.8	2.21	0.71	0.96	−8	−13
	0.06	0.25	8.3	2.44	0.84	0.90	—	—
		0.5	14.9	2.30	0.77	0.94	**4**	—
		1	21.1	1.96	0.64	0.78	−13	−17
0.4	0.02	0.25	14.2	2.20	0.99	1.36	—	—
		0.5	23.2	2.18	0.97	1.68	**24**	—
		1	31.7	2.04	0.89	1.54	**13**	−8
	0.04	0.25	11.8	2.07	0.96	1.10	—	—
		0.5	20.4	2.00	0.92	1.23	**12**	—
		1	29.0	1.85	0.81	1.17	**7**	−5
	0.06	0.25	9.7	2.01	0.92	1.02	—	—
		0.5	18.1	1.88	0.87	1.05	**3**	—
		1	26.8	1.70	0.76	0.98	−4	−7
0.5	0.02	0.25	15.3	1.81	0.99	1.41	—	—
		0.5	26.1	1.78	0.98	1.73	**22**	—
		1	37.4	1.73	0.94	1.85	**31**	**7**
	0.04	0.25	12.8	1.73	0.98	1.18	—	—
		0.5	23.1	1.68	0.95	1.33	**13**	—
		1	34.3	1.59	0.88	1.36	**15**	**2**
	0.06	0.25	10.5	1.72	0.96	1.15	—	—
		0.5	20.7	1.58	0.91	1.11	−4	—
		1	31.8	1.47	0.83	1.12	−3	**1**

the optimal policy will eventually dedicate some of the additional capacity to reducing queue length.

From Table 1, we also note that utilization decreases with server productivity regardless of congestion level. This is because utilization is driven by the probability of idling, which increases when productivity is improved. The reason is that the marginal revenue per unit time spent on a job in a high productivity system diminishes at a faster rate than that in a low productivity system. Hence, when there is only one job in the system, it is optimal to spend less time on it, and therefore the server becomes idle more quickly in a high productivity system than in a low productivity system. But this increased probability of idling does not necessarily mean that average queue length will be shorter. In some cases, the improved productivity system also has a higher probability of long queues (presumably because this does not present as big a risk of incurring large holding costs as when the server is less efficient). As a result, the average queue length can be larger or smaller as a result of the productivity increase. Because server behavior affects the shape of the queue-length distribution, utilization tells us less about DTC systems than about NDTC systems.

Because many service (e.g., call-center) and professional (e.g., engineering) systems involve multiple servers, it is worthwhile to ask whether the behavior observed here is somehow an artifact of our single-server model, or whether it can be expected to occur more generally in multiserver systems. While the curse of dimensionality prevents us from directly modeling systems with many servers, we can extend our analysis of the impact of an increase in productivity to two-server systems and draw inferences about the role the number of servers play. To do this, we developed a single-queue two-server MDP model with the state of the system given by (n, t_1, t_2), where n is the number of jobs in the system and t_1 (t_2) are the amount of time server 1 (server 2) has spent on the job. The details of the MDP model can be found in Appendix 1. We studied the two-server system using the same set of input parameters for the value-time function, minimum quality requirement, and holding cost that we used for the single-server system. However, to ensure that the single-server and

Table 2 The Effects of Capacity in Single-Server and Two-Server DTC Systems for Cases with Minimum Traffic Intensity $\rho_{\min} = 0.3$

$\rho_{\min} = 0.3$		Single-server system				Two-server system			
Holding cost ratio h/b	Rate parameter a	Average profit rate	Average service time	Expected queue length	% change in E[queue] from $a = 0.25$	Average profit rate	Average service time	Expected queue length	% change in E[queue] from $a = 0.25$
0.02	0.25	12.6	2.87	1.35	—	27.8	3.01	1.97	—
0.02	0.5	19.5	2.74	1.49	**11.0**	41.7	2.91	2.26	**15.0**
0.02	1	25.0	2.49	1.26	−6.0	51.9	2.66	1.94	−1.3
0.04	0.25	10.3	2.61	1.05	—	24.2	2.87	1.63	—
0.04	0.5	17.0	2.49	1.11	**6.0**	37.7	2.75	1.78	**9.1**
0.04	1	22.8	2.23	0.96	−8.0	48.4	2.38	1.53	−6.5
0.06	0.25	8.4	2.43	0.99	—	21.0	2.75	1.48	—
0.06	0.5	14.9	2.31	0.94	**4.0**	34.4	2.56	1.49	**0.5**
0.06	1	21.1	2.01	0.78	−13.0	45.6	2.16	1.25	−15.5

two-server systems have the same minimum utilization $\rho_{\min}$, we set the arrival rate in the two-server system to be double the single-server's arrival rate.

The results of these tests are summarized in Table 2, which shows that a capacity increase (via productivity improvement) may intensify congestion in the two-server system, as well as in the single-server system. Specifically, depending on the shape of the value-time curve and the cost parameter, systems may take advantage of the productivity improvement to increase quality instead of reducing holding cost.

Hence, as we observed in the examples in Table 2, this phenomenon does not depend critically on the number of servers. Increasing the number of servers may make the system more prone or less prone to increasing congestion due to capacity increase. From a practical standpoint, this means that the counterintuitive behavior we have observed here may occur in multiserver real-world systems.

From a managerial perspective, this result is potentially significant. For example, adding capacity might be expected to shorten customer waiting times in a call center and to decrease the number of pending problems in an engineering group. But our results suggest that if management insists on observing shorter delays, they may induce a suboptimal solution. Hence, it is important for management to measure and control both quality and responsiveness in setting improvement goals.

5.2. Increasing the Number of Workers

We also studied the effect of increasing capacity by adding an additional server to a single-server system. Using our original MDP model for single-server DTC systems with state (n, t) and single-queue two-server DTC systems with state (n, t_1, t_2) we introduced in the previous section, we studied a range of examples with the same input parameters. For each case, we determined the optimal policy and the corresponding queue length in both systems. However, even though the two-server system had twice as much capacity and the same arrival rate, it sometimes occurred that the average queue length was longer in the two-server system. Hence, again, the presence of the quality buffer from capacity increase may intensify congestion.

Because the results were similar to those of the previous section, we omit the detailed summary for briefness (a numeric summary is available in Appendix 3).

6. Queue Pooling

It is well known that queue pooling improves system performance in NDTC systems by reducing idleness and waiting time. A natural question, therefore, is whether the same holds for DTC systems. If so, we would like to know how the benefits in a DTC system compare to those in the analogous NDTC system.

To address this question, we consider a single-job-class system with two servers, where the servers may have different speeds. In the unpooled system, servers receive independent Poisson arrivals λ_q, where $q = 1, 2$. Each server q has its own queue and service policy. The state of the system is (n_q, t_q) for $q = 1, 2$, where n_q represents the number in queue and t_q represents the number of periods the current customer has been in process of server q. The MDP model for each server in the unpooled system is identical to that in §4.1.

In the pooled system, arrivals to the two servers are combined into a single queue (where $\lambda = \lambda_1 + \lambda_2$). The state of the system is (n, t_1, t_2), where n is the combined queue length (similar to §5.2). Details of the MDP can be found in Appendix 1. Similar to §4.3, we study the following parameter settings to the system:

- *Minimum traffic intensity* $\rho_{\min}$. We define the minimum traffic intensity of the two-server system to be $\rho_{\min} = \lambda_1/\mu_1 + \lambda_2/\mu_2$. We considered values from 0.1 to 0.6 in increments of approximately 0.1, depending on the value of A and $f(\tau_{\min})$.
- *Holding cost to maximum value ratio* h/b. We considered values of 0.01, 0.02, and 0.04.

• *Minimum quality* $f(\tau_{\min})/b$. We considered values of 20%, 40%, and 60%.

Moreover, we examined different ratio of arrival rates, and each server q has its individual value function shape A_q and process flexibility ϕ_q as follows:

• *Shape of value function* A_q. We considered values of $A = 0.24$, 0.59, and 0.79.

• *Process flexibility* ϕ_q. We considered values ranging from 0.6 to 1 in increments of 0.2 for each server q, depending on A_q and $f^q(\tau_{\min})$.

We found in all 216 cases that queue pooling results in an increase in system profit in the DTC system, with the average and maximum profit increase to be 8.9% and 48%, respectively. We found that, as with NDTC systems, the increase in profit from queue pooling is the greatest for cases that have large differences in arrival rates and in high minimum traffic intensity $\rho_{\min}$ (see Appendix 4 for details). Table 3 illustrates the average percentage increase in system performance from unpooled to pooled queues. Our results show that as the difference in the process flexibility of the two servers increases (i.e., as $|\phi_2 - \phi_1|$ increases), the benefit of queue pooling increases.

Next, to compare the benefit of queue pooling in DTC and NDTC environments, we specified a step function for the NDTC case (see Figure 1(a)) that "matches" the value-time function in the DTC case. For each case, we fitted a step function to the value-time function for each server such that the step function matched the optimal ST policy in the corresponding DTC system.

We found that system improvement is much greater in NDTC systems than in DTC systems. The increase in profit from queue pooling in NDTC systems is on average 32% greater than that in DTC systems, with a maximum difference of 80% (see Appendix 4 for a detailed summary). Hence, we can make the following observation.

Observation 4. The benefit of queue pooling in DTC environments is significant but less significant than that in NDTC environments.

The reason for this is that the flexibility of processing time in DTC systems allows the worker to balance server idling and customer waiting time. Because NDTC systems lack this flexibility, they must rely more heavily on queue pooling to reduce idleness and waiting time. This suggests that queue pooling is more attractive for tasks that are closer to NDTC than DTC in nature. For example, in a call center, queue pooling is likely to generate more benefit when order taking is the agent's primary responsibility (NDTC in nature) than when upselling is the primary responsibility (DTC in nature). Similarly, in an engineering group, cross-training workers to capture the benefit of queue pooling is likely to yield better results for NDTC tasks like drafting than for DTC tasks like design problem resolution.

Given the limited benefit offered by queue pooling in DTC systems, managers should closely study whether the disadvantages of queue pooling (e.g., free riding; see Olson 1965 and Hardin 1968) outweigh the benefits. Researchers have noted that the risk of free riding increases in groups (Latane et al. 1979, McGrath 1984). Moreover, in many service and professional environments, customers may prefer one server over another. Hence, combining all customers into a single queue may lower overall customer satisfaction and erode long-term customer loyalty. These issues suggest that managers should be very cautious when implementing queue pooling in DTC systems.

Table 3 Average Percentage Increase in DTC System Profit from Queue Pooling

Process flexibility of server 1 (ϕ_1)	Process flexibility of server 2 (ϕ_2) (%) 0.6	0.8	1.0	Average
0.6	5.2	8.3	9.7	7.7
0.8	9.7	6.2	9.3	8.4
1.0	12.9	9.9	7.8	10.1
Average	9.3	8.1	9.2	8.9

7. Task Variability

In this section, we investigate how discretion in task completion interacts with task variability to generate further insights into the differences between DTC and NDTC systems. To do this, we compare the performance of systems with no discretion (NDTC), partial discretion (DTC with DT heuristic), and full discretion (DTC) in environments with and without task variability.

7.1. Task Variability in DTC and NDTC Systems

It is well known that increasing task variability degrades system performance in NDTC systems (Hopp and Spearman 2000). For example, in traditional manufacturing systems, measures of cycle time, work in process, and on-time delivery grow worse as task variability increases.

It is reasonable to ask whether such an increase in task variability in DTC systems also results in a deterioration of system performance. For example, in a call center with upselling, customers may have similar revenue potential but, due to different personalities and time constraints, may require different amounts of conversation time to achieve this potential. Similarly, in an engineering group, warranty repair problems may have similar cost saving potential but different levels of complexity. Therefore, the problems may require different amounts of diagnostic time to achieve a certain level of design efficiency.

We examine this question by comparing DTC systems with a single-task type (one value-time curve) and multiple-task types (two value-time curves). We assume that the value-time functions for the multitask-type system ($f_1(\tau)$ and $f_2(\tau)$) have different rates of increase (i.e., different values of a) but have the same minimum service requirement (i.e., same $\tau_{\min}$) and converge to the same level in the long run (i.e., the same maximum value b). This represents a system that processes two types of tasks that share the same revenue potential but require different times to obtain that revenue, for example, customers in a call center with upselling that have the same potential for upselling but require different amounts of time to be convinced to make a purchase. We also assume that customers with each value-time function arrive according to independent Poisson processes.

To make the single-task-type and multitask-type systems comparable, we define the value-time function for the single-task-type system as the weighted average of the functions in the multitask-type system such that

$$f_{\text{single}}(\tau) = f_1(\tau)\gamma + f_2(\tau)(1-\gamma),$$

where γ is the fraction of arrivals corresponding to the value-time function $f_1(\tau)$.

To model the multitask-type DTC system, we constructed an MDP with state space (n, t, q), where n is the number of jobs in the system, t is the number of time intervals the current job has been under service, and q is the type of task under service ($q = 1, 2$). Similar to most NDTC models with multiple task types, we assume that the task type becomes clear to the server once he/she starts to perform the task. To isolate the effect of task variability from that of queue information, we also assume that the server has no information on the composition of task types in queue. The details of the MDP and parameter inputs for the numerical study are in Appendix 2.

In contrast to the principle that variability *always* degrades performance in NDTC systems, we observed that increasing task variability in DTC systems *may* improve performance. In the 243 cases we studied, we found that 24% of them showed greater system profit, and 26% have less congestion in the multitask-type system than in the single-task-type system. The magnitude of the profit improvement can be as large as 4%. We also noted that an increase in profit via higher task variability is often coupled with a reduction in queue length. The greatest queue reduction was as large as 33%.

In Table 4, we summarize the performance of the system profit and congestion level based on the average flexibility level and the difference in value-time function shapes. We define $\bar{\phi}$, the average flexibility[4] between the two tasks in the multitask-type system, as follows:

$$\bar{\phi} = \phi_1\gamma + \phi_2(1-\gamma),$$

where ϕ_i is the flexibility of task i and γ is the percentage of type 1 arrivals.

Table 4 Effect of Task Variability on System Performance in DTC Systems

Average process flexibility $\bar{\phi}$	Ratio of shape parameters (A_1/A_2)	% difference in system profits			% difference in average queue length		
		Min.	Avg.	Max.	Min.	Avg.	Max.
0.6	1.33	−1.3	0.8	1.1	0.7	11.9	24.7
	2.5	−1.8	0.3	0.4	−1.2	7.5	25.2
	4.0	−2.4	0.1	0.3	−1.6	3.0	20.1
0.7	1.33	0.0	0.1	0.6	−0.3	1.9	17.6
	2.5	0.1	0.5	0.9	−1.1	7.6	25.2
	4.0	1.4	1.8	2.3	1.0	11.8	23.3
0.8	1.33	0.1	0.2	0.5	−1.3	0.5	3.0
	2.5	0.3	0.7	1.1	−2.9	1.7	9.5
	4.0	1.8	2.4	3.0	−4.0	2.8	13.1
0.9	1.33	0.2	0.3	0.4	−3.1	0.7	3.9
	2.5	0.5	0.9	1.3	−6.8	1.8	9.5
	4.0	2.3	2.9	3.6	−18.3	2.2	13.0
1.0	1.33	0.3	0.4	0.7	−4.8	0.8	4.8
	2.5	0.6	0.8	1.3	−10.7	1.9	9.6
	4.0	2.3	3.0	3.7	−32.7	1.6	13.0

We found that as the average process flexibility increases, the improvement in system performance between the multitask-type (high variability) system and single-task-type (low variability) system increases. Moreover, the effect of task variability is amplified (either positively or negatively) by differences in the shape parameter of the value-time functions.

The underlying reason for this interesting phenomenon (i.e., that task variability can be beneficial in DTC systems) is that DTC systems are able to take advantage of the information about task type and utilize it (by adjusting completion time) to improve system performance. In contrast, even when information about the task type is clear to the NDTC servers, they cannot adjust anything to compensate for the variable value-time functions.

For both DTC and NDTC systems, task variability leads to fluctuation in the amount of time required to generate certain revenue. Such *process fluctuation* has a negative effect on both types of system performance. However, task variability also has a positive effect on DTC systems, namely, that variability in the

[4] We verified in this study that average process flexibility represents the overall system discretion level just as well as considering the two process flexibilities individually. For simplicity, we only make use of average flexibility in this discussion.

value-time functions gives the DTC server more room to adjust process times. In other words, DTC servers have an *additional degree of freedom* when faced with two value-time functions than with only one value-time function. For instance, in the call-center example, agents could adjust their service time to upsell items only to "faster" customers but not to "slower" ones. Depending on the specific nature of the value-time curves, this could yield a better profit than the best upselling strategy for "medium" customers. The benefit of the additional discretion sometimes exceeds the inherent cost of process fluctuation and allows task variability to be beneficial to the DTC system. Hence, we conclude the following.

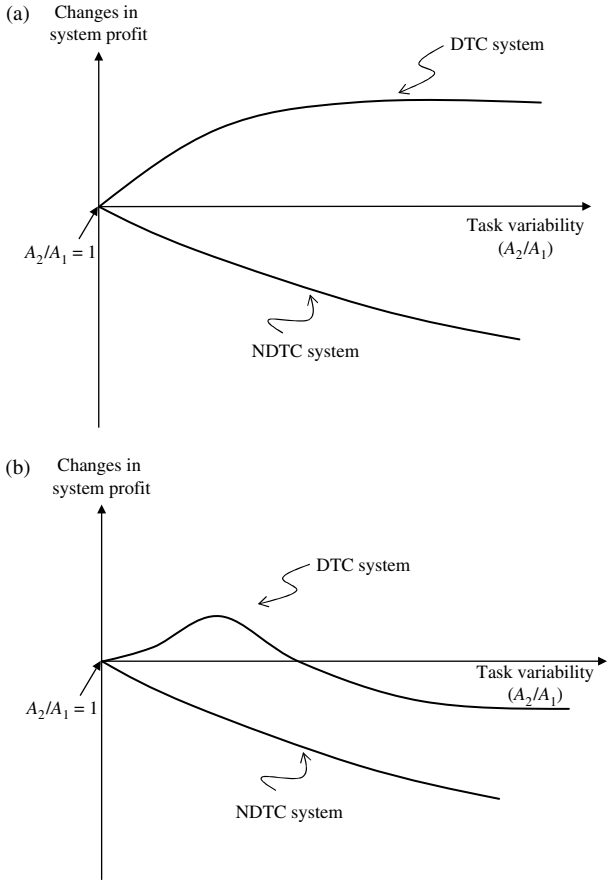
Observation 5. DTC systems take advantage of task variability, which may improve system performance.

In some cases, the queue reduction is relatively large.[5] This is because the increase in discretion allows the system to significantly reduce the processing time spent on each task. However, at the same time, the corresponding revenue generated also decreases greatly and hence moderates the queue's effect on the overall system profit.

As noted earlier, Table 4 suggests that a high value of shape parameter ratio (A_2/A_1) amplifies the effect of task variability. The shape parameter ratio represents the difference in value generating rates of the two tasks; the larger the ratio, the greater the difference in times required for the tasks to obtain their maximum values. To deepen our understanding of the impact of process flexibility under task variability, we examined performance of systems under an array of A_2/A_1 values for two scenarios: *Scenario* 1—high-process flexibility ($\bar{\phi} = 0.9$); *Scenario* 2—low-process flexibility ($\bar{\phi} = 0.3$). To accomplish that, we first chose a single-task-type system to represent each scenario. Then, we evaluated the performance changes of a set of equivalent multitask-type systems with different A_2/A_1 values (such that $f_1(\tau)\gamma + f_2(\tau)(1-\gamma) = f_{\text{single}}(\tau)$).

Figure 6 shows the improvement in system profit when a single-task-type system is replaced by its equivalent multitask-type system. The example is constructed with the following parameters: $h/b = 0.02$, $A_{\text{single}} = 0.5$, $b = 100$, and $\gamma = 0.5$. Figure 6(a) illustrates cases with high-process flexibility ($\bar{\phi} = 0.9$, Scenario 1) and shows that the improvement of the multitask-type system increases quickly in the A_2/A_1 ratio (i.e., difference in value generating rates), but levels off as the ratio becomes large. This is because, initially, the DTC system can greatly benefit from the increased flexibility to adjust process times provided by having two value-time functions. However, as the two tasks becomes very different, the incremental gain of task variability diminishes. When A_2/A_1 becomes very large, it becomes optimal for the server to spend $\tau_{\min}$ on both tasks (see Figure 7(a)). The reason is that the server would not spend extra time on the low-revenue (small A) task because of the extremely low marginal benefit, while he/she would not spend extra time on the high-revenue (high A) task because almost all of the potential benefit is obtained within $\tau_{\min}$ units of time. As a result, the two concave functions can be simplified to two step functions with the same processing time but different revenue levels (i.e., the dotted lines in Figure 7). The system can be approximated by a simple multitask-type NDTC system.[6] Hence, as the two value-time functions become extremely different (A_2/A_1 grows

Figure 6 Relative Impact of Changing Task Variability in Single-Task-Type and Multitask-Type Systems



Note. (a) Scenario 1 (high-process flexibility); (b) Scenario 2 (low-process flexibility).

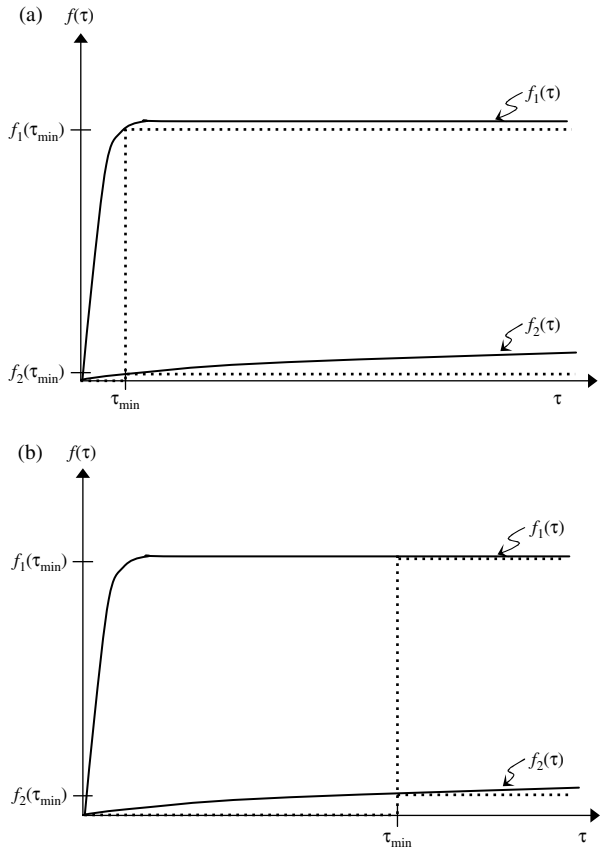
[5] The conditions for this to happen are: (1) the average process flexibility is large, and (2) the two tasks are quite different (large A_2/A_1 ratio).

[6] Congestion of this system can be approximated by an $M/D/1$ queue with service time $\tau_{\min}$, while the revenue of the system per unit time can be computed as the weighted average of $(1/(\lambda_1 + \lambda_2))[\lambda_1 f_1(\tau_{\min}) + \lambda_2 f_2(\tau_{\min})]$.

large), the performance difference between the single-task-type and the multitask-type systems stabilizes.

Figure 7(b) illustrates the results for cases with low-process variability ($\bar{\phi} = 0.3$, Scenario 2). Under these conditions, the multitask-type DTC system exhibits a small increase in system profit relative to the single-task-type system when the A_2/A_1 ratio is small. However, system performance quickly deteriorates as the A_2/A_1 ratio grows larger. The percentage profit change soon becomes negative and eventually levels off when the ratio becomes very high. This is because low-process flexibility limits the ability of the worker to take advantage of the additional degree of freedom provided by having two value-time curves, and hence moderates the benefit of task variability in DTC systems. As the difference between the two tasks increases, the negative effect of process fluctuation (i.e., queueing) quickly exceeds the benefit from the additional degree of freedom. As a result, the multitask-type system becomes inferior to the single-task-type system. Similar to the multitask-type DTC system in Scenario 1, when the difference between the two tasks becomes large enough, the two value-time concave functions become very similar to two step-functions with the same minimum processing time ($\tau_{\min}$) but different revenue levels (see the dotted lines in Figure 7(b)). However, due to the high $\tau_{\min}$, congestion is more severe under Scenario 2 and hence task variability tends to deteriorate DTC system performance.

Figure 7 **Value-Time Functions of a Multitask-Type System with Extremely High Task Variability (Large A_2/A_1)**



Note. (a) Scenario 1 (high-process flexibility); (b) Scenario 2 (low-process flexibility).

We also plotted the relative profit changes of NDTC systems in Figure 7. We considered NDTC tasks that share the same parameter input as their corresponding DTC tasks, including the arrival rate λ, potential revenue b, holding cost h, and minimum quality $\tau_{\min}$. In multitask-type DTC systems, the ratio of A_2/A_1 represents the difference in value generating rates of the two functions. The corresponding measure in NDTC systems is the ratio of the two processing times $q\tau_2/\tau_1$. Hence, for each value of A_2/A_1, we chose a pair of step functions (whose processing times are τ_1 and τ_2) such that $\tau_2/\tau_1 = A_2/A_1$. As expected, system profit of the multitask-type NDTC system deteriorates at a much greater rate as the A_2/A_1 ratio increases for Scenarios 1 and 2. It is clear that task variability tends to be significantly more beneficial (or less harmful) in DTC systems than NDTC systems.

These two scenarios illustrate the dynamics of how performance is driven by the level of process flexibility. Moreover, because of the structure of the value-time functions, performance of DTC systems is fairly robust to changes in the level of task variability. While the direction of the task variability effect can be positive or negative, the resulting size is relatively small (in our numerical study, the percentage changes in system profit were all within ±4%). We conclude the following:

Observation 6. Process flexibility amplifies the effect of task variability in DTC systems, but system profit is relatively insensitive to changes in task variability.

Multiserver Systems. To see whether the interaction between task variability and discretion is similar in multiserver systems, we developed a single-queue two-server MDP with two task types. The state of the system is (n, t_1, t_2, q_1, q_2), where q_1 (q_2) is the type of value-time function of the job currently under process by server 1 (server 2). We compared systems with one and two value-time functions for the same set of parameters used in the single-server tests. Of the 243 cases we considered, 27% exhibited improved performance with multitask-type systems and the maximum improvement was 4.5%. The behavior in the two-server case was almost identical to that in the one-server case, suggesting that the interaction between variability and discretion is not strongly influenced by the number of servers.

Partial Discretion Model. Finally, we examined whether the relationship between task variability and

discretion is similar in DTC systems with partial discretion. This is of interest because practical DTC systems are likely to use some form of simplified policy to adjust service times. We considered the DT heuristic, whose effectiveness was illustrated earlier in §4.3. We compared systems with one and two value-time functions for the same set of parameters used in the single-server tests. For systems with a single-task type (low variability), we computed the optimal DT policy (N, T_1, T_2). For systems with two task types (high variability), we computed the optimal DT for each of the value-time functions, $(N, T_1^1, T_2^1, T_1^2, T_2^2)$, where T_1^q (T_2^q) is the T_1 (T_2) threshold for a customer of type q $(q = 1, 2)$.

We observed that the system with two task types outperforms the system with a single-task type in 55% of the 243 cases we studied. We found that the relative improvement from the single-task type to multitask-type systems can be as large as 6.8%. This implies that, if anything, the variability in value-time functions is valuable when a heuristic control policy is used. Evidently, the flexibility gained from defining thresholds for each value-time function is effective in exploiting the information inherent in the value-time function variability. Finally, we also observed that partial discretion systems with multiple servers benefit from variability in the value-time functions to an even greater extent.

We summarize our results in Figure 8. The shaded arrows in the figure specify the direction of system improvement. In the environment with no server discretion (NDTC), the shaded arrow pointing downward indicates that a single-task-type system always performs better than its equivalent multitask-type system. However, in the environments with partial (DT heuristics) or full server discretion (optimal policy), the two-headed shaded arrows indicate that a single-task-type system may perform better or worse than an equivalent multitask-type system.

This result is significant because it implies that in sharp contrast to NDTC systems, reducing task variability may not be an appropriate goal in DTC systems. Moreover, the fact that variability can improve performance even under a DT heuristic suggests that this insight applies to practical settings where the value-time curves are only partially observable and servers adjust their policy by trial and error (e.g., regular mode when the queue is short, hurry-up mode when the queue is long).

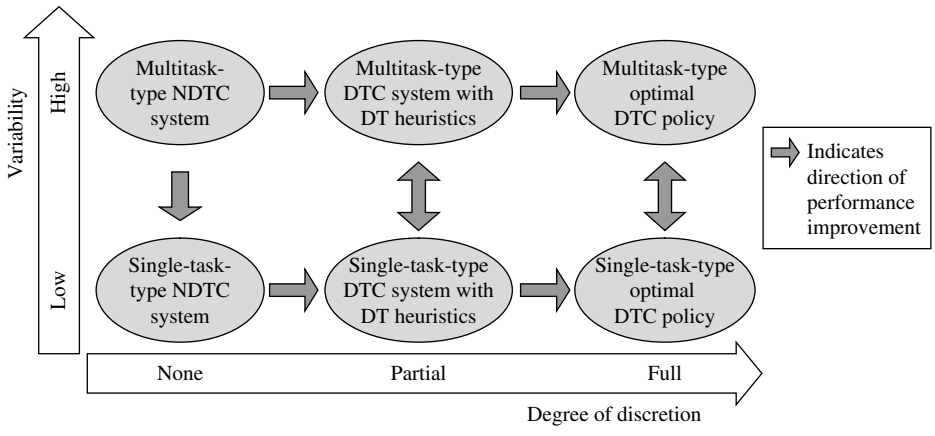
7.2. Discretion in Low and High Variability Environments

Now that we have a model for high variability (i.e., multitask-type) DTC systems, we can investigate the question of whether discretion is more valuable in systems with high or low variability. We do this by comparing the profit gain between no server discretion and full server discretion in systems with a single-task type (low variability) and in systems with two task types (high variability).

To make a fair comparison between DTC and NDTC systems, we used the same approach as in §6 and specified the step function for the NDTC case that corresponds to the best single-threshold heuristic for the DTC case. Using these step functions as the value-time curves, we computed the profit in the NDTC environment for both the single-task-type and multitask-type cases by using the appropriate MDP model (from §4.1 and §7.1, respectively).

With a series of numerical tests, using the same parameters from the variability study of §7.1, we found that the changes in system performance resulting from increased discretion are very similar in systems with low variability and high variability. The average increase in profits in low and high variability systems were 4.3% and 4.4%, respectively, with a maximum increase of around 18% for both cases. Hence, task variability does not significantly affect the positive impact of server discretion.

Figure 8 Interaction of Task Variability and Server Discretion



This result is important from a practical standpoint because, while the existence of value-time functions is a reasonable assumption, the use of a single curve to represent all customers probably is not. Hence, it is useful to know that increasing server discretion is helpful even when customers are nonhomogeneous. This result is summarized in Figure 8. The fact that discretion produces a similar degree of benefit in high and low variability settings is represented by the shaded arrow pointing from left to right. The figure summarizes that while discretion is unambiguously helpful in all systems and variability is clearly harmful in NDTC (no discretion) systems, variability may be helpful in DTC (partial or full discretion) systems.

8. Conclusions

In this paper, we have presented analytical models that capture one important characteristic of many operations systems, particularly those involving service and professional work—discretionary task completion. Our main insight is that in operations systems with discretionary task completion times, it is attractive to adjust processing time, and hence job quality, in response to system congestion. In factory physics terminology, quality is a variability buffer (along with capacity, inventory, and time). The presence of this fourth type of variability buffer causes systems with discretionary completion times to exhibit some very different behaviors from systems with nondiscretionary completion times.

Most strikingly, in contrast with NDTC systems, increasing capacity in DTC systems may intensify congestion. This occurs when the value gained from higher quality outweighs the increase in holding cost. Another distinctive phenomenon we found in DTC systems is that task variability can be beneficial. This is because discretion allows the server to take advantage of the difference in task types to increase flexibility. The managerial implications for DTC systems of these counterintuitive results are that (a) output quality, together with responsiveness, must be incorporated into evaluation of improvement initiatives; and (b) reducing task variability may not always be an appropriate goal. We also showed that DTC systems benefit less from queue pooling than NDTC systems, which implies that pooling strategies are more appropriate for systems with NDTC characteristics.

The above insights were obtained by examining the optimal service policy for single-station DTC work systems. However, the optimal policy is probably too complex to implement in real-world settings, particularly because value-time curves can only be approximated. We showed that a double-threshold heuristic achieves nearly optimal performance, while a single-threshold heuristic performs reasonably well only when tasks are fairly similar to NDTC tasks. Both of these policies would be simple to implement (possibly by trial and error) in practice. These results imply that the benefits and behaviors of DTC systems are still valid in systems that use practical heuristic control policies.

The work in this paper represents a first step toward modeling and understanding work systems with discretionary process times. To develop a more comprehensive set of principles concerning design and management of such work systems, further modeling work is needed.

For instance, it would be useful to extend the single-station model to network models of multistage DTC work systems. An example of such a network is a problem resolution team in which different experts work together to solve design and manufacturing problems. A problem resolution project has to go through several stages, such as data collection, diagnostics, solution generation, analysis, and implementation. If different people are responsible for different stages, then the system constitutes a tandem network of DTC tasks. These could be iterative if complications arise during the process.

In multistage systems, each stage has a distinct purpose and requires a different skill set. This raises questions about interactions and the impact of discretionary task completion at the different stages; for example, what is the best way to accommodate variable quality from upstream stages at downstream stages? How do we define the bottleneck in such DTC systems?

Another useful area to explore in the context of DTC systems would be the setting where the objectives of the employee and the firm do not align. For example, the firm is interested in maximizing the long-term average value of the system, while the agent is more interested in his/her short-term utility gain. This could be modeled as a version of the classic principal-agent problem and used to investigate interesting questions such as: What is an effective yet robust incentive compensation plan? Are the utility functions of workers in the DTC system distinctly different from those in the NDTC system? How do factors like an agent's personality and experience affect the characteristics of the agent's utility function?

The long-term goal of research on DTC work systems is to gain insights into the best ways to design, manage, and improve operations in which workers have discretion in task completion. By understanding the interactions between capacity, variability, information, and customer service (determined by both quality and responsiveness), DTC workers can make better time management decisions, while managers can make better decisions regarding cross-training, information sharing, and collaboration policies.

An e-companion to this paper is available as part of the online version that can be found at http://mansci.pubs.informs.org/.

Acknowledgments

The authors thank the department editor, the associate editor, and the two anonymous referees for many insightful comments that greatly improved this paper. This work was supported by the National Science Foundation under grant DMI-0423048.

References

Adler, P., A. Mandelbaum, V. Nguyen, E. Schwerer. 1995. From project to process management: An empirically-based framework for analyzing product development time. *Management Sci.* **41**(3) 458–484.

Aksin, O. Z., P. T. Harker. 1999. To sell or not to sell: Determining the tradeoffs between service and sales in retail banking phone centers. *J. Service Res.* **2**(1) 19–33.

Altiok, T. 1996. *Performance Evaluation of Manufacturing Systems*. Springer-Verlag, New York.

Armony, M., I. Gurvich. 2005. When promotions meet operations: Cross-selling and its effect on call-center performance. Working paper, Stern School of Business, New York University, New York.

Askin, R. G., J. B. Goldberg. 2001. *Design and Analysis of Lean Production Systems*. Wiley, Hoboken, NJ.

Ata, B., S. Shneorson. 2006. Dynamic control of an $M/M/1$ service system with adjustable arrival and service rates. *Management Sci.* **52**(11) 1778–1791.

Aydin, G., S. Ziya. 2005. Upselling a promotional product using customer purchase information. Working paper, The University of Michigan, Ann Arbor, MI.

Bailey, D. E. 1998. Comparison of maufacturing performance of three team structures in semiconductor plants. *IEEE Trans. Engrg. Management* **45**(1) 20–32.

Banker, R., J. Field, K. Sinha. 2001. Work-team implementation and trajectories of manufacturing quality: A longitudinal study. *Manufacturing Service Oper. Management* **3**(10) 25–42.

Boudreau, J., W. Hopp, J. McClain, I. J. Thomas. 2003. On the interface between operations and human resources management. *Manufacturing Service Oper. Management* **5**(3) 179–202.

Bureau of Economic Analysis. 2001. Gross domestic product by industry. U.S. Department of Commerce. Retrieved December 7, 2004, http://www.bea.gov/bea/dn2/gdpbyind_data.htm.

Buzacott, J. A., J. G. Shanthikumar. 1992. Stochastic models of manufacturing systems. Prentice Hall, New York.

Debo, L. G., L. B. Toktay, L. N. Van Wassenhove. 2004. Queueing for expert services. Working paper, Tepper School of Business, Carnegie-Mellon University, Pittsburgh, PA.

Doerr, K. H., T. R. Mitchell, T. D. Klastorin, K. A. Brown. 1996. Impact of material flow policies and goals on job outcomes. *J. Appl. Pysch.* **81**.

Gans, N., G. Koole, A. Mandelbaum. 2003. Telephone call centers: Tutorial, review, and research prospects. *Manufacturing Service Oper. Management* **5**(2) 79–141.

General Motors. 1974. GM Public Interest Report, Detroit, MI.

General Motors. 2002. GM Corporate Responsibity Report, Detroit, MI.

George, J. M., J. M. Harrison. 2001. Dynamic control of a queue with adjustable service rate. *Oper. Res.* **49**(5) 720–731.

Gross, D., C. M. Harris. 1985. *Fundamentals of Queueing Theory*, 2nd Ed. Wiley, New York.

Hammer, M., J. Champy. 1993. *Reengineering the Corporation: A Manifesto for Business Revolution*. Haper Business, New York.

Hardin, G. 1968. The tragedy of the commons. *Science* **162** 1243–1248.

Hopp, W., M. Spearman. 2000. *Factory Physics: Foundations of Manufacturing Management*, 2nd ed. McGraw-Hill, Burr Ridge, IL.

Krishnan, V., S. D. Eppinger, D. E. Whitney. 1997. A model-based framework to overlap product development activities. *Management Sci.* **43**(4) 437–351.

Latane, B., K. D. Williams, S. Harkins. 1979. Many hands make light the work: The causes and consequences of social loafing. *J. Personality Soc. Psych.* **37** 822–832.

Loch, C. H., C. Terwiesch. 1999. Accelerating the process of engineering change orders: Capactiy and congestion effects. *J. Product Innovation Management* **16**(2) 145–169.

McGrath, J. E. 1984. *Groups: Interaction and Performance*. Prentice-Hall, Englewood Cliffs, NJ.

Olson, M. 1965. *The Logic of Collective Action: Public Goods and the Theory of Groups*. Harvard University Press, Cambridge, MA.

Owen, S. H., W. C. Jordan. 2003. An extended framework for studying white-collar work. Working paper, GM Research Laboratories, Warren, MI.

Schultz, K. L., D. C. Juran, J. W. Boudreau, J. O. McClain, L. J. Thomas. 1998. Modeling and worker motivation in just in time production systems. *Management Sci.* **44**(12) 595–607.

Stidham, S., Jr., R. R. Weber. 1989. Monotonic and insensitive optimal policies for control of queues with undiscounted costs. *Oper. Res.* **87**(4) 611–625.