Dynamically scheduling and maintaining a flexible server

Jefferson Huang[1], Douglas G. Down[2], Mark E. Lewis[3], and Cheng-Hung Wu[4]

[1]Operations Research Department, Naval Postgraduate School, Monterey, CA 93943-5098, USA

[2]Department of Computing and Software, McMaster University, Hamilton, Ontario L8S 4L7, Canada

[3]School of Operations Research and Information Engineering, Cornell University, Ithaca, NY 14853-3801, USA

[4]Institute of Industrial Engineering, National Taiwan University, 1, Sec. 4, Roosevelt Rd., Taipei 106, Taiwan

Abstract

Deciding how to jointly schedule jobs and perform preventive maintenance is a fundamental problem in flexible manufacturing systems, particularly those arising in semiconductor manufacturing. At the same time, past work in this area shows that, even when there is only one station and one type of job, identifying policies that minimize the amount of work-in-process (WIP) is also a difficult problem. In this paper, we study a single-station version of this problem wth two types of jobs, with the objective of minimizing average maintenance costs plus the weighted average amount of WIP. We identify conditions under which it suffices to schedule jobs according to both a server-state-dependent version of the $c\mu$-rule, and a static $c\mu$-rule where the average service rates are used. One of these conditions states that the ratio between the service rates should remain constant as the server deteriorates. When this assumption does not hold, scheduling with the $c\mu$-rule can in fact lead to an unstable system; we illustrate this using a simple example. On the other hand, we also present numerical evidence that $c\mu$-based scheduling performs well compared to other scheduling rules, and relative to an "optimal" policy based on solving a Markov decision process.

1 Introduction

The yield of a manufacturing process, defined as the fraction of output that is of sufficient quality, is a key economic performance indicator; see e.g., [3]. In semiconductor manufacturing, yield improvement has been recognized as an effective means of managing costs and sustaining profitability [2]. In particular, yield increases on the order of even 1-2% can lead to significant savings in wafer manufacturing costs [9].

One of the key determinants of yield is the health of the machines processing the jobs that eventually become finished products. As the underlying condition of a given machine deteriorates, the increased frequency of significant process deviations (as identified by, e.g., statistical process control procedures [24]) leads to more re-work and tuning, which in turn reduces the rate at which good products are produced (i.e., the "service rate" of the machine). Eventually, it can become worthwhile to take the deteriorated machine offline for maintenance, after which the service rate is improved.

The need for good maintenance policies, and the increasing prevalance of sensorized equipment in the semiconductor and other advanced manufacturing industries, has led to the emergence of condition-based maintenance (CBM) as a potentially cost-effective alternative to more commonly used age or job-based maintenance rules [8, 13]. At the same time, almost every machine used to process jobs in the semiconductor manufacturing setting is flexible, in the sense that it can be used to process more than one type of job. Hence a fundamental problem in semiconductor manufacturing, and more generally in flexible manufacturing systems with deteriorating equipment, is how to simultaneously (1) allocate jobs to flexible machines that deteriorate over time, and (2) perform preventive maintenance on those machines.

When there is a single operation to be performed, Kaufman and Lewis [17] show the difficulty in developing control policies that use as inputs both the current workload in the system and the condition of the machine. This leaves a broad class of manufacturing system configurations, which includes configurations arising in the semiconductor industry, without guidance on how to consider the trade-offs between resource allocation and resource maintenance. In this paper, we consider the question of joint maintenance and scheduling in a parallel queueing setting. When there are two classes of jobs, the manager can decide to assign a single resource (henceforth called a *server*) to either class, or to begin preventive maintenance. The goal is to provide adequate service (in the form of minimizing weighted queue-lengths) to each class, while noting that a deteriorated server works slower. Since [17] shows in the single queue setting that the usual monotonicity properties of an optimal control do not hold, there is little hope of finding simple solutions to the scheduling/maintenance pairing. Instead, we seek insights into the following questions:

1. Given a choice between prioritizing scheduling or maintenance, where should a decision-maker focus his/her efforts?

2. Under what conditions can classic scheduling/maintenance results be used to create useful heuristics?

We provide an answer to both questions by presenting conditions under which scheduling with a natural extension of the classic $c\mu$-rule is without loss of optimality (Theorem 5), and numerical results indicating that this heuristic performs well more generally (Section 5).

1.1 Related Literature

The two types of decisions described above, namely maintaining a deteriorating machine and scheduling jobs in a queueing system, have typically been considered separately in the literature. In particular, the majority of papers in the maintenance literature do not consider the effect that the amount of work in the system (i.e., the queue length) may have on optimal maintenance decisions; see e.g., the surveys [18, 20, 22, 25]. Two papers where such effects are accounted for, via models that are very closely related to the one presented in Section 2, are Kaufman and Lewis [17] and Cai et al. [5], which we describe in more detail below. Moreover, while previous work such as that of Andradóttir et al. [1] and Wu et al. [26, 27] has accounted for server failures in the context of queueing models of flexible manufacturing systems, with the exception of Cai et al. [5] we are not aware of any other work in this area that combines scheduling with maintenance decisions. We note that there has been recent work on joint scheduling and maintenance in the contexts of deterministic scheduling [14], developing metaheuristics [7], and mixed-integer programming that incorporates constraints specific to semiconductor manufacturing [6, 28].

Kaufman and Lewis [17] analyze the structure of optimal maintenance policies for the server of an M/M/1 queue with only one type of job, where the service rate deteriorates according to a pure-death process. In particular, [17, Example 3.6] shows that the optimal policy under the average cost criterion may not be monotone in the queue lengths. For certain deterioration levels it may be optimal to perform maintenance when there are no queued jobs, not perform maintenance when there are few queued jobs, and to perform maintenance for all sufficiently large queue lengths. On the other hand, [17, Theorems 3.2, 4.2; Proposition 4.10] provide conditions under which there is an optimal policy that is monotone in the state of the server. This means that there is an optimal policy with the following structure: For each fixed number of queued jobs i, there is a threshold s_i^* such that maintenance is performed if and only if the deterioration level is worse than s_i^*. Finally, numerical experiments are presented [17, Section 5] that illustrate some pitfalls associated with using some simple and natural heuristics, underscoring the difficulty of the problem.

Cai et al. [5] consider an $M/G/1$ queueing model with at most two types of jobs, which is motivated by potential semiconductor manufacturing applications. In this model, the service and deterioration dynamics differ from those in [17]. In particular, jobs cannot be preempted while they are being served, and deterioration events can only occur when a service completion occurs. On the other hand, while in [17] it is assumed that at each deterioration event the server moves to the "next-worse" state with probability 1, the model in Cai et al. [5] allows the server to move, as a result of a single deterioration event, to any state that is worse. For this model, analogous results to the ones in [17] hold. Namely, the optimal policy may not be monotone in the number of jobs [5, Section 5], but under certain conditions there exists an optimal policy that is monotone in the state of the server [5, Theorems 3.3, 4.3]. In addition, for the case of two types of jobs, it is shown that it may be suboptimal to always prioritize a job type that seems, from a cost and deterioration perspective, to be superior to the other [5, Section 4.2]. A monotonicity result [5, Theorem 4.4] on the value function when one job type is superior to the other in the aforementioned sense is also provided. Finally, numerical results [5, Sections 5,6] are presented to illustrate the savings that an optimal joint scheduling and maintenance policy can provide, relative to first-in first-out scheduling and maintenance after a fixed number of jobs have been completed.

In this paper, we consider joint scheduling and maintenance in the context of a

G/M/1 queue with two types of jobs[1]. As in Kaufman and Lewis [17], and in contrast to Cai et al. [5], we assume that jobs can be preempted by the decision-maker, or interrupted by a failure event. We also consider deterioration dynamics that are more general than those in [17], and which differ from those in [5]. In particular, in [5] it is assumed that deterioration events must coincide with service completions, but that deterioration rates can depend on which type of job is worked on. In contrast, we assume that deterioration events can happen at any time, but that the deterioration rates are the same for both job types. While Cai et al. [5] argue that work-dependent deterioration is important for certain semiconductor manufacturing applications, Sloan and Shanthikumar [23] note that for some wafer fabrication processes, such as in etch operations, it is reasonable to assume that deterioration does not depend on the type of job.

1.2 Contributions and Outline

The main contributions of the paper are as follows. After presenting the scheduling and maintenance model in Section 2, we consider the problem of scheduling in the presence of a deteriorating server in Section 3, without preventive maintenance. We provide a condition (Assumption CR) under which it is optimal to schedule the jobs according to a static priority rule, when the service rates are modulated according to a (possibly non-Markovian) point process; see Theorem 2 and Remark 3. In addition, Example 1 shows that, when the conditions of Theorem 2 do not hold, scheduling according to the aforementioned priority rule can in fact lead to an unstable system, in the sense that the average number of queued jobs grows to infinity. From the perspective of system design, this provides a strong incentive to

[1]All of our results extend to the case of a finite number of job classes

invest in ensuring that Assumption CR below holds. Next, in Section 4 we return to the joint scheduling and maintenance problem. We use the results in Section 3 to provide conditions under which it suffices to search for an optimal policy among those that schedule according to a priority rule and that monotone in the state of the server (Theorem 7). In Section 5, we provide numerical results indicating that the priority-rule based scheduling policies considered in Section 4 can perform well across a range of system parameters. The numerical results also underscore the value of good maintenance policies, and of incorporating service rate information in scheduling and maintenance policies. This latter point was also observed by Iravani and Duenyas [16], in the context of a single job type. Finally, conclusions and future research directions are presented in Section 6. Unless otherwise indicated, proofs of stated results are provided in Appendix A.

2 Joint Scheduling and Maintenance Model

Two classes of jobs arrive randomly over time. Each arriving job requires a random amount of work, and all incoming work is processed by a single server. The arrival times of the jobs are modeled by independent point processes on $\mathbb{R}_+ := [0, \infty)$, while the amount of work required by each arriving job is assumed to be exponentially distributed with unit mean, independently of the other jobs. It is assumed that the arrival process is *regular*, in the sense that with probability 1, there can be at most a finite number of arrivals during any finite interval of time.

Jobs of the same class are homogeneous. However, both the cost incurred by a waiting job and the time required to complete a job depend on the job's class. For $k = 1, 2$, the cost incurred by a waiting job of class k is assumed to accumulate continuously at a constant rate c_k.

As time progresses, the health of the server deteriorates. This is modeled by assuming that when the server is able to perform work, it spends a random amount of time in its current state $s \in \{0, 1, \ldots, S\}$ before deteriorating to a state that is at least as bad. In particular, lower numbered states indicate worse health. In addition, given that a deterioration event has occured, the probability that the server then transitions to state ℓ from its current state s is $q(\ell|s)$, where $q(\ell|s) = 0$ if $\ell > s$. Once the server's state reaches 0, it undergoes maintenance for a random amount of time, after which it returns to state S. We assume that the times at which the server changes state are independent of both the amount and the nature of the work that the server has completed. In particular, the random times at which the server changes state are modeled by a point process on $\mathbb{R}_+$ that is independent of the arrival processes and work requirements. Like the arrival processes, this process of deterioration times is also assumed to be regular.

The rate at which the server can complete work of a given class depends on the server's health. If the state of the server is s, then the rate at which it can complete work of class k is μ_k^s. We assume that $\mu_k^0 = 0 < \mu_k^1 \leqslant \cdots \leqslant \mu_k^S < \infty$ for each class $k = 1, 2$. In other words, the server cannot complete any work while it is undergoing maintenance (i.e., is in state 0), and higher-numbered states indicate less deterioration.

For ease of exposition, the server is referred to as being *online* if its state is not 0, and *offline* if its state is 0. In addition, the server is said to *deteriorate* if it transitions from state $s \geqslant 2$ to $\ell \geqslant 1$, and *fails* if it transitions to state 0 without the influence of the decision-maker. Whenever a failure occurs, *corrective maintenance* (CM) is initiated at cost $K_c \geqslant 0$. When the server is online, the decision-maker can initiate *preventive maintenance* (PM), which is modeled as an instantaneous transition of the server state to 0 at cost $K_p \geqslant 0$.

In addition, when the server is online and there is a job in the system, the server may be assigned to work on that job. Since jobs of the same class are assumed to be homogeneous, we equate selecting a job to work on with selecting which class to assign the server to.

The decision-maker is only able to exert control over the server at decision epochs, which occur whenever one of the following events occurs:

- A job arrives and the server is online.
- A job is completed and the server is online.
- The server deteriorates or fails.
- The server comes back online (from state 0).

Accordingly, jobs that are currently in service may be preempted.

At each decision epoch, the decision-maker knows the current state of the system, i.e., the number i_k of jobs of each class $k \in \{1, 2\}$ present and the state $s \in \{0, 1, \dots, S\}$ of the server. Let $\mathbb{X} := \{0, 1, \dots\}^2 \times \{0, 1, \dots, S\}$ denote the set of all possible system states. In addition to the current state, the decision-maker also knows the history of the system (i.e., the past queue lengths, server states, and event times) up to the current decision epoch. In deciding whether to serve one of the classes or initiate PM, the decision-maker follows a policy π that prescribes (possibly in a randomized way) the action to take at each decision epoch, given the current state and history of the system. We restrict attention to policies that are *non-idling* (i.e., never call for an online server to idle when there is work to do) and *non-anticipative* (i.e., do not depend on future information). Let Π denote the set of all such policies. Of particular interest are the *deterministic stationary* policies; under such a policy π, the action $\pi(x)$ is taken whenever the system is in state $x \in \mathbb{X}$.

We compare policies on the basis of the long-run average cost per unit time incurred from a given initial state. To define this optimality criterion, fix any $\pi \in \Pi$. Let $Q_1^\pi(t)$ (resp. $Q_2^\pi(t)$) denote the number of jobs in class 1 (resp. 2), including those in service, at time t. Also, let $S^\pi(t)$ denote the state of the server at time t under π, and let $M_c^\pi(t)$ (resp. $M_p^\pi(t)$) equal 1 if CM (resp. PM) is initiated at time t under π, and let $M_c^\pi(t)$ (resp. $M_p^\pi(t)$) equal 0 otherwise. Finally, for $n = 1, 2, \ldots$ let t_n^π denote the n^{th} decision epoch under π.

If the system is in state $(i_1, i_2, s) \in \mathbb{X}$ at time 0, then the long-run expected *average cost* per unit time that is incurred by following the policy π is

$$w^\pi(i_1, i_2, s) := \limsup_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{n:t_n^\pi \leqslant T} \left[K_c M_c^\pi(t_n^\pi) + K_p M_p^\pi(t_n^\pi)\right] + \int_0^T \sum_{k=1}^{2} c_k Q_k^\pi(t)\, dt \,\middle|\, Q_1^\pi(0) = i_1,\ Q_2^\pi(0) = i_2,\ S^\pi(0) = s\right].$$

A policy $\pi_* \in \Pi$ is *optimal* if $w^{\pi_*}(i_1, i_2, s) = \min_{\pi\in\Pi} w^\pi(i_1, i_2, s)$ for every initial state $(i_1, i_2, s) \in \mathbb{X}$.

3 Scheduling Without Preventive Maintenance

We first consider the case where the decision-maker cannot perform PM, and can only schedule jobs in the presence of a deteriorating server. In this setting, the server can only go offline (i.e., enter state 0) via a failure. Note that, since preventive maintenance is not permitted and the server state evolves independently of the scheduling decisions, the maintenance costs are independent of the policy used. The decision-maker's objective is therefore to find a scheduling policy π that minimizes the weighted long-run average expected number of jobs in the system

$$\limsup_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\int_0^T \sum_{k=1}^{2} c_k Q_k^{\pi}(t)\, dt\right]. \tag{1}$$

in the presence of uncontrollable server deterioration.

3.1 $c\mu$-Rules

Without server deterioration, it follows from [19, Theorem 2.1] that it is optimal to schedule according to the $c\mu$-*rule*. According to this rule, if the service rate for class k jobs is μ_k, then priority is given to any class k^* where $c_{k^*}\mu_{k^*} \geqslant c_k\mu_k$ for every class k.

When the service rate depends on the state of the server, it is natural to consider prioritizing the jobs according to a *state-dependent* $c\mu$-*rule*. Namely, if the state of the server is s, prioritize any class k^* where $c_{k^*}\mu_{k^*}^s \geqslant c_k\mu_k^s$ for every class k. Alternatively, letting $\nu(s)$ denote the long-run expected fraction of time that the server spends in state s, one could employ the following *average* $c\mu$-*rule*: Assign priority to any class k^* for which $c_{k^*}\bar{\mu}_{k^*} \geqslant c_k\bar{\mu}_k$ for every class k, where $\bar{\mu}_k := \sum_{s=0}^{S}\nu(s)\mu_k^s$ is the *average service rate* for class k jobs.

3.2 Instability of $c\mu$-Rules

Observe that if $c_1\mu_1^s \geqslant c_2\mu_2^s$ for every server state s, then both the state-dependent and average $c\mu$-rules described above would prioritize class 1, regardless of the server state. While it is tempting to conjecture that prioritizing class 1 in this situation is optimal, the following example shows that doing so could in fact be *very* suboptimal.

Example 1. *Suppose jobs of classes 1 and 2 arrive according to Poisson processes with rates* $\lambda_1 = 5$ *and* $\lambda_2 = 0.8$*, respectively. There are only two server states,* $q(1|2) = 1$*, and the inter-deterioration times are exponential with rate 1. The service rates are* $\mu_1^1 = \mu_1^2 = 10$ *and* $\mu_2^s = s$ *for* $s = 1, 2$*. Finally, corrective maintenance occurs instantaneously.*

Under both the state-dependent and average $c\mu$*-rules, class 1 is given priority regardless of the server state. Note that under this policy, the average service rate for class 2 jobs is* $0.5(0.5 \cdot 2 + 0.5 \cdot 1) = 0.75 < 0.8 = \lambda_2$*. This indicates that, regardless of the initial state, the system is unstable and has infinite long-run expected average cost. A formal proof that the system is unstable when class 1 is always prioritized is given in Appendix A.1.*

On the other hand, consider the policy that prioritizes class s *jobs when the server state is* s*, for* $s = 1, 2$*. Since* $\lambda_2 = 0.8 < (0.5)(2) = 1$*, this policy incurs a finite long-run expected average cost regardless of the initial state. This can be proved by showing that the associated fluid model is stable; see Appendix A.2 for details.*

3.3 Optimality of $c\mu$-Rules

The following condition guarantees that it is optimal to always prioritize one class over the other.

Assumption CR (Constant Ratio)**.** *Every state* $s \in \{1, \ldots, S\}$ *(i.e., where the server is online) satisfies*

$$\mu_1^{s-1}\mu_2^s = \mu_1^s\mu_2^{s-1}. \tag{2}$$

Assumption CR states that the *ratio* of the service rates for class 1 and class 2 jobs remains constant as the server changes state. It can be interpreted as saying that the different service capabilities of the flexible server are *affected equally by deterioration*. Note that Assumption CR implies, but is not equivalent to, the condition that $c_1\mu_1^s \geqslant$

$c_2\mu_2^s$ for every server state s. For instance, Example 1 satisfies $c_1\mu_1^s \geqslant c_2\mu_2^s$ for every $s \in \{0, 1, \dots, S\}$, but does not satisfy Assumption CR.

The following theorem, which is the main result of this section, states that Assumption CR guarantees the optimality of the state-dependent and average $c\mu$-rules described at the beginning of this section.

Theorem 2. *If Assumption CR holds, then it is optimal to always prioritize one class over the other. In particular, consider any server state $s' \geqslant 1$. If*

$$c_1\mu_1^{s'} \geqslant (\textit{resp.} \leqslant)\ c_2\mu_2^{s'}, \tag{3}$$

then Assumption CR implies that (3) holds when s' is replaced with any $s \in \{0, 1, \dots, S\}$, and that it is optimal to prioritize class 1 (resp. 2).

Proof. We use Assumption CR to adapt the interchange argument in Nain [19, Proof of Theorem 2.1] to our setting.

The first step is to show that, for every $T \geqslant 0$, the problem of minimizing the finite-horizon expected weighted queue lengths

$$\mathbb{E}\left[\int_0^T \sum_{k=1}^{2} c_k Q_k^\pi(t)\ dt\right] \tag{4}$$

can be reduced to a reward-maximization problem that is amenable to analysis via an interchange argument. To do this, we define some processes of interest. Consider any policy $\pi \in \Pi$ and fixed time $t \in [0, \infty)$. Let $U^\pi(t)$ denote the job class that the server is assigned to at time t under the policy π, and let[2]

$$a_k^\pi(t) := \mathbf{1}\{Q_k^\pi(t-) > 0,\ U^\pi(t) = k\}, \qquad k = 1, 2.$$

[2]Given a function $f : [0, \infty) \to \mathbb{R}$, let $f(t-) := \lim_{u \uparrow t} f(u)$ for $t > 0$.

Also, recalling that we are considering the case of no preventive maintenance, let $S(t)$ denote the state of the server at time t, and let

$$\phi^{\pi}(t) := \int_0^t \sum_{k=1}^{2} c_k \mu_k^{S(u)} a_k^{\pi}(u)\, du.$$

To reduce the problem of minimizing (1) to that of maximizing

$$\mathbb{E}\left[\int_0^T \phi^{\pi}(t)\, dt\right], \tag{5}$$

consider the queue-length processes $Q_k^{\pi}(t)$, $k = 1, 2$, under π, and let $A_k(t)$ denote the cumulative number of class k arrivals during the time interval $[0, t]$. Using an argument analogous to that in [19, Proof of Lemma 2.1] (replace μ_k with $\mu_k^{S(u)}$ and the fact that [4, Partial Result, p. 24] holds for Poisson processes with rates that depend on $S(t)$),

$$\mathbb{E}\left[\int_0^T \sum_{k=1}^{2} c_k Q_k^{\pi}(t)\, dt\right] = \mathbb{E}\left[\int_0^T \sum_{k=1}^{2} c_k [Q_k(0) + A_k(t)]\right] - \mathbb{E}\left[\int_0^T \phi^{\pi}(t)\, dt\right]. \tag{6}$$

Since the first term on the right-hand side of (6) does not depend on π, it follows that minimizing (4) is equivalent to maximizing $\mathbb{E}\left[\int_0^T \phi^{\pi}(t)\, dt\right]$.

The next step is to show that the $c\mu$-rule, denoted by $\pi_{c\mu}$, maximizes (5) for every finite horizon $T \geqslant 0$, i.e., that

$$\mathbb{E}\left[\int_0^T \phi^{\pi_{c\mu}}(t)\, dt\right] \geqslant \mathbb{E}\left[\int_0^T \phi^{\pi}(t)\, dt\right]. \tag{7}$$

for all $\pi \in \Pi$ and $T \geqslant 0$. To prove this, we use a slight modification of the sample-path-based construction in [19, Proof of Theorem 2.1] to show that in fact,

$$\int_0^T \phi^{\pi_{c\mu}}(t)\, dt \geqslant \int_0^T \phi^{\pi}(t)\, dt \tag{8}$$

holds with probability 1 (written w.p.1).

Noting that every policy is optimal if $T = 0$, fix $\pi \in \Pi$ and $T > 0$. Suppose Assumption CR holds, and assume the job classes are numbered so that $c_1\mu_1^s \geqslant c_2\mu_2^s$ (see the comments following the definition of Assumption CR). Then the $c\mu$-rule stipulates that class 1 should be prioritized. Consider the random time

$$\sigma_\pi := \inf\{t > 0 \mid Q_1^\pi(t-) > 0,\ U^\pi(t) = 2\},$$

which denotes the first time that the policy π does not follow the $c\mu$-rule. If $\sigma_\pi \geqslant T$ w.p.1, then (8) holds w.p.1., since in this case the policy π follows the $c\mu$-rule during $[0, T]$.

On the other hand, suppose $\sigma_\pi < T$ with positive probability, in which case there is a positive probability with which π does not follow the $c\mu$-rule during $[0, T]$. Let Task A denote the class 1 job that is assigned to the server at time σ_π under the $c\mu$-rule and Task B be the class 2 job the server is assigned to under π.

Let $\Pi_\infty \supset \Pi$ denote the set of all possibly anticipative policies. We will now construct a policy $\pi_+ \in \Pi_\infty$ that follows the $c\mu$-rule at time σ_π and satisfies

$$\int_0^T \phi^{\pi_+}(t)\, dt \geqslant \int_0^T \phi^\pi(t)\, dt \qquad w.p.1. \tag{9}$$

First, for all times $t \in [0, \sigma_\pi)$, let π_+ coincide with the $c\mu$-rule. To define π_+ for times $t > \sigma_\pi$, consider the random variable

$$\sigma_\pi^* := \min\{T_A, \tau_\pi\},$$

where T_A denotes the amount of time required to complete Task A when the server state is $S(\sigma_\pi)$, and $\tau_\pi := \inf\{t \geqslant \sigma_\pi \mid U^\pi(t) \neq 2 \text{ or } S(t) \neq S(\sigma_\pi)\}$ is the first time after time σ_π that either under π the server stops working on Task B or the server changes state. Assume the policy π_+ works on Task A during $[\sigma_\pi, \sigma_\pi + \sigma_\pi^*)$.

To complete the "interchange" of Tasks A and B, we will complete the definition of π_+ so that after some time ν_π, the number of queued jobs under both π_+ and π agree w.p.1. In particular, let ν_π denote the time when Task A is completed under the policy π. During $[\sigma_\pi + \sigma_\pi^*, \nu_\pi)$, let π_+ mimic the actions taken under π with the following exception: Whenever π works on Task A, but Task A has already been completed under π_+, the latter policy works on Task B instead. Finally, let π_+ mimic the actions taken under π at all times $t \geqslant \nu_\pi$.

We claim that at time ν_π, both the queue lengths and the amount of work remaining in the system are the same under both π and π_+. To verify this, let

$$\kappa := \frac{\mu_1^{s-1}}{\mu_1^s} = \frac{\mu_2^{s-1}}{\mu_2^s}, \qquad s \geqslant 1,$$

and let $I_n = [\theta_n, \theta_n')$ denote the n^{th} time interval in $[\sigma_\pi + \sigma_\pi^*, \nu_\pi)$ during which π_+ serves class 2 while π serves class 1 and the server state does not change. Observe that under π_+, the amount of work done on Task A during $[\sigma_\pi, \sigma_\pi + \sigma_\pi^*)$ is the same as the amount of work done on this job during $\cup_n I_n$ under π. Note that by Assumption CR,

$$\mu_k^r = \kappa^{s-r} \mu_k^s, \qquad k = 1, 2,\ r, s \geqslant 1; \tag{10}$$

this is because when $r > s \geqslant 1$,

$$\mu_k^r = \frac{\mu_k^r}{\mu_k^{r-1}} \cdots \frac{\mu_k^{r-(r-s-1)}}{\mu_k^s} = \kappa^{s-r} \mu_k^s,$$

and when $s > r \geqslant 1$,

$$\mu_k^r = \frac{\mu_k^r}{\mu_k^{r+1}} \cdots \frac{\mu_k^{r+(r-s-1)}}{\mu_k^s} = \kappa^{s-r} \mu_k^s.$$

Using (10), the amount of work done on Task A during $[\sigma_\pi, \sigma_\pi + \sigma_\pi^*)$ can be written as

$$\mu_1^{S(\sigma_\pi)} \sigma_\pi^* = \sum_n \mu_1^{S(\theta_n)} (\theta_n' - \theta_n) = \mu_1^{S(\sigma_\pi)} \sum_n \kappa^{S(\sigma_\pi) - S(\theta_n)} (\theta_n' - \theta_n). \tag{11}$$

From (11), we conclude that

$$\sigma_\pi^* = \sum_n \kappa^{S(\sigma_\pi)-S(\theta_n)}(\theta_n' - \theta_n). \tag{12}$$

Hence the amount of work that is done on Task B during $\cup_n I_n$ under π_+ is

$$\sum_n \mu_2^{S(\theta_n)}(\theta_n' - \theta_n) = \mu_2^{S(\sigma_\pi)} \sum_n \kappa^{S(\sigma_\pi)-S(\theta_n)}(\theta_n' - \theta_n) = \mu_2^{S(\sigma_\pi)}\sigma_\pi^*,$$

which is precisely the amount of work done on Task B during $[\sigma_\pi, \sigma_\pi + \sigma_\pi^*)$ under the original policy π. Since π_+ selects exactly the same actions as π at all times $t \in [\sigma_\pi + \sigma_\pi^*, \nu_\pi) \setminus \cup_n I_n$, it follows that both the queue lengths and the remaining amount of work in the system at time ν_π are the same under both π and π_+.

Since the policies π and π_+ couple at time ν_π, the validity of (9) and the optimality of the $c\mu$-rule for every finite horizon T can be proved by following [19, Proof of Theorem 2.1] and using the preceding definitions of $\phi^\pi(t)$, σ_π^*, and the intervals $[\theta_n, \theta_n')$. It follows *a fortiori* that the $c\mu$-rule is optimal under the average-cost criterion (1). $\square$

Remark 3. *The proof of Theorem 2 does not rely on the assumption that deterioration events always send the server to a state that is worse. In particular, it holds when the service rates are simply assumed to be modulated (not necessarily in a Markovian way) according to the point process that describes the deterioration process. Hence the proof of Theorem 2 implies that, for a two-class* $G/M/1$ *queue with modulated service rates that satisfy Assumption CR, it is optimal to schedule according to the* $c\mu$*-rule.*

4 Scheduling with Preventive Maintenance

We now consider the problem of optimally making both scheduling and preventive maintenance decisions. In Section 4.1, we provide conditions under which it suffices to schedule jobs according to the state-dependent $c\mu$-rule, or the average $c\mu$-rule, described in Section 3.1. Then, in Section 4.2 we present conditions under which optimal maintenance decisions are monotone in the state of the server. When the conditions hold, these results simplify the computation of optimal policies. At the same time, when one or more of the conditions do not hold, they suggest heuristics that may still perform well. The performance of scheduling with the $c\mu$-rule, when the conditions of Theorem 4 in this section do not hold, is considered numerically in Section 5.

4.1 Optimal Scheduling

In this section, a *maintenance policy* is a rule that stipulates, given the current state of the system, whether or not to initiate maintenance. If maintenance is not initiated, a *scheduling policy* determines which customer class (if any) should be served. The set of all stationary deterministic maintenance policies is identified with the set of all functions $f : \{0, 1, \dots\}^2 \times \{0, 1, \dots, S\} \to \{0, 1\}$ where $f(i_1, i_2, s) = 1$ (resp. $= 0$) if and only if the maintenance policy f calls for maintenance to be initiated (resp. no maintenance) when the state is (i_1, i_2, s). Note that $f(i_1, i_2, 0) = 1$ for all $i_1, i_2 \in \{0, 1, \dots\}$.

According to Theorem 2 in Section 3, if the ratio between the service rates for class 1 and class 2 jobs remains constant as the server changes state (i.e., Assumption CR holds), then the $c\mu$-rule is the optimal scheduling policy in the presence of a deteriorating server that cannot be preventively maintained. In the context of joint scheduling and maintenance, Theorem 2 can be generalized to Theorem 4 below. To state this theorem, a maintenance policy f is said to be *queue-oblivious* if there exists a function $g : \{0, 1, \ldots, S\} \to \{0, 1\}$ satisfying

$$f(i, j, s) = g(s) \qquad \text{for all } \; (i, j, s) \in \{0, 1, \ldots\}^2 \times \{0, 1, \ldots, S\}.$$

In other words, a queue-oblivious maintenance policy is stationary, deterministic, and does not depend on any queue-length information. Examples of queue-oblivious maintenance policies include *server threshold* policies (where the server is maintained if and only if its state is below a certain threshold), *job-based* policies (where the server is maintained whenever a certain fixed number of jobs have been completed), and *calendar-based* policies (where the server is maintained whenever a certain fixed amount of time has elapsed).

Proposition 4. *Suppose Assumption CR holds. Then under any queue-oblivious maintenance policy, it is optimal to prioritize one class over the other. In particular, consider any server state $s \geqslant 1$, and assume that the current state is one in which the maintenance policy calls for no maintenance. If*

$$c_1 \mu_1^s \geqslant (\textit{resp.} \leqslant)\; c_2 \mu_2^s, \tag{13}$$

then Assumption CR implies that (13) holds for all $s \in \{0, 1, \ldots, S\}$, and it is optimal to prioritize class 1 (resp. 2).

Proof. Under a queue-oblivious maintenance policy, the evolution of the server state does not depend on how the jobs are served. The theorem then follows from the proof of Theorem 2, which does not require any assumptions on where the server state transitions to when deterioration events occur (see Remark 3). □

Proposition 4 immediately implies the following theorem, which is the main result of this section.

Theorem 5. *If Assumption CR holds, and the decision-maker is restricted to queue-oblivious maintenance policies, then it is without loss of optimality to only consider joint scheduling and maintenance policies where jobs are scheduled according to the (static) priority policy described in Theorem 4.*

4.2 Optimal Maintenance Decisions

Up to this point, we have only assumed that the arrival processes for the two job classes are described by independent point processes on $\mathbb{R}_+$. Under Assumption M below, the problem can be formulated as a *semi-Markov decision process (SMDP)*. The main result in this section (Theorem 6) states that under this assumption and Assumption S below, the search for an optimal policy can be restricted to policies that are monotone in the server's health.

Assumption M (Markovian Arrivals and Deterioration)**.**

(i) The point processes modeling the arrival times of jobs of class 1 and 2 are independent Poisson processes with rates $\lambda_1 > 0$ *and* $\lambda_2 > 0$*, respectively.*

(ii) The server deteriorates according to a continuous-time Markov chain. In particular, if its current state is $s \in \{1, \ldots, S\}$*, then the time until the next deterioration event is exponentially distributed with rate* $\alpha_s > 0$*.*

(iii) The maintenance times (i.e., the times that the server spends in the offline state) are independent and identically distributed with common distribution $G(\cdot)$ *whose mean* $1/\alpha_0 := \int_0^\infty t \, dG(t)$ *satisfies* $0 < 1/\alpha_0 < \infty$*.*

Assumption S (Stability)**.**

(i) There is a server state $s^* \in \{1, \dots, S\}$ *satisfying*

$$\frac{\lambda_1}{\sum_{s=s^*}^{S}(\mu_1^s/\alpha_s)} + \frac{\lambda_2}{\sum_{s=s^*}^{S}(\mu_2^s/\alpha_s)} < \frac{1}{(1/\alpha_0) + \sum_{s=s^*}^{S}(1/\alpha_s)}.$$

(ii) The server can only deteriorate to the next-worse state, i.e.,

$$q(s-1|s) = 1 \qquad \forall s \geqslant 1.$$

A joint scheduling and maintenance policy is *monotone in the server's health* if, for every fixed number of class 1 and class 2 jobs in the system, PM is initiated whenever the server's health state is sufficiently low. The following proposition states that under Assumptions M and S, one can restrict the search for an optimal joint scheduling and maintenance policy to deterministic stationary policies that are monotone in the server's health. A proof is provided in Appendix A.3.

Proposition 6. *Suppose Assumptions M and S hold. Then there exists an optimal joint scheduling and maintenance policy that is deterministic, stationary, and monotone in the server's health.*

Combining the conclusions of Propositions 4 and 6 leads to the following theorem, which is the main result in this section.

Theorem 7. *Suppose Assumptions CR, M, and S hold. Then there exists an optimal deterministic stationary policy that is both monotone in the server's health and schedules jobs according to the* $c\mu$*-rule.*

Remark 8. *Under Assumptions M and S, there may not be an optimal policy that is monotone in the queue lengths. In particular, letting* $K_c = K_p = 0$, $c_1 = c_2 = 1$, $\lambda_1 = \lambda_2 = 1$,

$S = 4$, $\mu_1^1 = \mu_2^1 = 1/2$, $\mu_1^2 = \mu_2^2 = 1$, $\mu_1^3 = \mu_2^3 = 3/2$, $\mu_1^4 = \mu_2^4 = 2$, *and* $\alpha_s = 1/5$ *for* $s = 0, 1, 2, 3, 4$, *we obtain the model instance considered in [17, Example 3.6]. It was shown in [17] that the optimal policy for this model is such that, for server state 2, initiating maintenance is optimal when there are no jobs, not optimal when there are 1 to 11 jobs, and optimal when there are more than 11 jobs.*

5 Numerical Experiments

While we have only been able to prove the optimality of the $c\mu$-rule within the class of all joint scheduling and maintenance policies that are queue-oblivious, the simulation results presented in this section suggest that using the $c\mu$-rule remains nearly optimal when the assumptions of Theorem 4 are violated. Moreover, the results of the experiments illustrate the significant savings that good maintenance policies can provide, under a variety of scheduling policies.

5.1 Assumptions

Our primary objectives in performing the simulation experiments are to investigate, with the criterion of minimizing the average number of jobs in the system,

- the performance of various scheduling policies and
- the savings that good preventive maintenance can provide,

relative to an optimal policy that is computed via dynamic programming. To this end, we consider the following version of the scheduling and maintenance model described in Section 2 that satisfies Assumptions M and S. Jobs of class 1 and 2 arrive according to independent Poisson processes with rates λ_1 and λ_2. When it is able to process jobs, the single server can be in the "deteriorated" or "like-new" states, denoted respectively by server states 1 and 2. Hence the set of possible server states is $\{0,1,2\}$. The (exponential) deterioration rates are denoted by α_1 and α_2, and the maintenance times are assumed to be iid exponential with rate α_0. As before, the respective (exponential) service rates will be denoted by μ_1^1, μ_1^2 and μ_2^1, μ_2^2. Finally, since we focus on the criterion of minimizing the average number of jobs in the system, we assume that the holding cost rates are $c_1 = c_2 = 1$, and that there are no fixed maintenance costs. The latter means that the cost of performing maintenance is equal to the holding costs incurred during a maintenance interval.

Two utilization levels under no (preventive) maintenance are considered: 75% (which is a typical target equipment utilization level in semiconductor wafer fabs [15, p. 62]) and 90%. For a given utilization level

$$\rho = \frac{\lambda_1}{\bar{\mu}_1} + \frac{\lambda_2}{\bar{\mu}_2}, \tag{14}$$

where

$$\bar{\mu}_k = \sum_{s=0}^{2} \frac{\mu_k^s/\alpha_s}{\sum_{s=0}^{2}(1/\alpha_s)}$$

is the average service rate for class k jobs. The model parameters were generated in the following way. First, the service rates $\mu_1^1, \mu_1^2, \mu_2^1, \mu_2^2$, the maintenance/deterioration rates $\alpha_0, \alpha_1, \alpha_2$, and the ratios λ_1/λ_2 were selected using a Plackett-Burman experimental design. The arrival rates were generated using (14). The parameters for each of the 12 systems that were simulated are presented in Table 1 below. The rates are given in jobs per hour. Observe that systems 3, 4, 5, 6, 9, and 12 do not satisfy Assumption CR.

System	Utilization	(λ_1, λ_2)	$(\mu_1^1, \mu_2^1), (\mu_1^2, \mu_2^2)$	α_0	α_1	α_2
1	0.75	(0.82, 0.82)	(2,2), (4,4)	0.33	0.083	0.067
2	0.75	(0.87, 0.87)	(4.8,3.2), (6,4)	0.5	0.083	0.1
3	0.75	(0.98,1.5)	(2, 4.8), (4, 6)	0.5	0.13	0.067
4	0.75	(1.03, 1.03)	(3,3.2), (6,4)	0.33	0.13	0.1
5	0.75	(1.19,0.79)	(3.2,3), (4, 6)	0.5	0.13	0.1
6	0.75	(1.3, 1.3)	(4.8,3), (6,6)	0.33	0.083	0.067
7	0.9	(0.82, 0.82)	(2,2), (4,4)	0.5	0.083	0.1
8	0.9	(0.89,1.3)	(3.2,4.8), (4,6)	0.33	0.083	0.1
9	0.9	(1.1,1.6)	(4.8,2), (6,4)	0.5	0.13	0.067
10	0.9	(1.4,0.96)	(3.2, 3.2), (4,4)	0.33	0.13	0.067
11	0.9	(1.5,1.5)	(3,3), (6,6)	0.33	0.13	0.1
12	0.9	(1.8,1.2)	(3,4.8), (6,6)	0.5	0.083	0.067

Table 1: Parameters of the Simulated Models

5.2 Policies

The following scheduling policies were evaluated for the systems described in Section 5.1.

MDP-Based Scheduling (DP): This is the optimal scheduling policy for the discrete-time MDP with finite state and action sets obtained by truncating and uniformizing the original continuous-time MDP.

$c\mu$**-Rule (CMU):** This is the (possibly dynamic) priority rule described in Section 3, where class 1 (resp. 2) is prioritized, when the server state is s, if $c_1\mu_1^s \geqslant$ (resp. $\leqslant$) $c_2\mu_2^s$.

Average $c\mu$-Rule (ACMU): This is the (static) priority rule where class 1 (resp. 2) is prioritized if $c_1\bar{\mu}_1 \geqslant$ (resp. $\leqslant$) $c_2\bar{\mu}_2$.

Longest-Queue-First (LQF): Under this scheduling policy, the longest queue is always served first.

Each of these scheduling policies was considered under both "optimal" maintenance (denoted by OM) and no maintenance. We consider the DP-OM policy where preventive maintenance is allowed, and the DP policy where no preventive maintenance is performed. In addition, letting $\mathcal{S}$ stand for CMU, ACMU, or LQF, we consider both the $\mathcal{S}$-OM policy obtained by fixing the scheduling policy $\mathcal{S}$ and solving the corresponding MDP where only maintenance decisions need to be made, and the $\mathcal{S}$ policy under which there is no preventive maintenance.

5.3 Results

For each of the parameter settings in Table 1, an average over 50 replications of a 5-year simulation time horizon was used to estimate the expected average number of queued jobs under each of the policies described in Section 5.2. These estimates are presented in Tables 2 and 3 below, with 95% confidence intervals.

5.3.1 Scheduling Without Preventive Maintenance

We first consider the case of scheduling without preventive maintenance. The systems where Assumption CR does not hold are indicated with boldface System IDs.

Recall that in this setting, it was proved in Section 3 that, if Assumption CR holds, then the state-dependent and average $c\mu$-rules become the same static priority policy, and that this policy is optimal. The results for the systems where Assump-

System ID	DP	CMU	ACMU	LQF
1	4154 ± 108	4326 ± 204	4235 ± 156	4107 ± 119
2	1099 ± 35	1121 ± 31	1092 ± 31	1144 ± 26
3	2440 ± 78	2342 ± 60	2424 ± 57	2800 ± 77
4	3251 ± 109	3274 ± 78	3263 ± 79	3549 ± 90
5	1982 ± 62	1966 ± 56	2033 ± 51	2185 ± 83
6	2331 ± 38	2323 ± 71	2394 ± 95	2533 ± 119
7	4092 ± 139	4174 ± 103	4022 ± 60	4043 ± 172
8	2388 ± 72	2376 ± 81	2389 ± 40	2601 ± 32
9	5501 ± 196	5633 ± 182	5534 ± 182	6898 ± 249
10	5801 ± 102	5832 ± 187	5904 ± 162	5887 ± 226
11	8618 ± 219	8258 ± 269	8374 ± 200	8636 ± 293
12	2796 ± 65	2947 ± 65	3030 ± 62	3246 ± 100

Table 2: Average Queue Lengths with 95% Confidence Intervals: Scheduling Without Preventive Maintenance

tion CR holds (namely, those with a System ID of 1, 2, 7, 8, 10, or 11) are consistent with this theoretical result. Moreover, with the exception of system 12, scheduling according to the cμ-rule (CMU) or the average cμ-rule (ACMU) is not significantly worse than the scheduling policy DP obtained by solving an MDP.

In addition, while LQF performs comparably to DP and CMU under some system parameters (namely, for systems 1, 2, 5, 7, 10, and 11), it is significantly worse in the remaining cases (namely, for systems 3, 4, 6, 8, 9, and 12). Moreover, the performance of LQF tends to exhibit more variability than DP and the policies based on the cμ-rule. A natural reason for this is that LQF does not take the service rates for the different job types and server states into account. This means that an accumulation of jobs with a low service rate can cause LQF to become "stuck" serving those jobs, while other jobs with higher service rates accumulate. This lends support to the idea that service-rate information is important to developing good scheduling policies.

5.3.2 Scheduling With Preventive Maintenance

Next, we consider jointly making scheduling and preventive maintenance decisions. The simulation results are presented in Table 3 below. As was done in Table 2, 95% confidence intervals are provided and the systems that do not satisfy Assumption CR are indicated by boldface System IDs.

As was the case with the CMU policy in Section 5.3.1, the CMU-OM and ACMU-OM policies track the performance of the MDP-based policy DP-OM in all cases except System 12, which does not satisfy Assumption CR. Moreover, we again see that LQF-based scheduling generally leads to worse and more variable performance. Hence, in both the cases of no preventive maintenance and optimal preventive maintenance, the simulation results suggest that using $c\mu$-based scheduling often results in nearly-optimal performance, when minimizing the queue lengths is the primary objective. A caveat, of course, is that this is only a heuristic, and system 12 provides an example where it may be comparable to using the more naïve longest-queue-first scheduling rule.

Finally observe that, with the addition of optimal preventive maintenance, significant gains in performance were realized in systems 1, 3, 5, 7, 9, 11, and 12, where systems 3, 5, 9, and 12 do not satisfy Assumption CR. For systems 2, 4, 6, 8, and 10, where systems 4 and 6 do not satisfy Assumption CR, the results indicate that good scheduling is more important, as optimal preventive maintenance does not

System ID	DP-OM	CMU-OM	ACMU-OM	LQF-OM
1	2630 ± 118	2590 ± 63	2731 ± 51	4230 ± 108
2	1092 ± 21	1093 ± 22	1104 ± 25	1143 ± 34
3	1744 ± 41	1753 ± 49	1760 ± 65	2813 ± 80
4	3236 ± 121	3210 ± 117	3233 ± 118	3523 ± 74
5	1718 ± 64	1770 ± 38	1755 ± 50	1950 ± 37
6	2268 ± 114	2375 ± 79	2428 ± 110	2427 ± 101
7	2009 ± 63	2042 ± 58	2051 ± 39	2025 ± 54
8	2346 ± 55	2365 ± 63	2484 ± 50	2597 ± 51
9	2903 ± 45	2944 ± 93	2878 ± 75	3295 ± 106
10	5786 ± 247	5755 ± 134	5926 ± 137	5899 ± 221
11	5378 ± 191	5497 ± 171	5483 ± 166	5430 ± 247
12	1790 ± 61	1903 ± 53	1911 ± 54	1916 ± 72

Table 3: Average Queue Lengths with 95% Confidence Intervals: Scheduling with Optimal Preventive Maintenance

reduce the average queue lengths by a significant amount. The reason for this is not apparent to us, as the parameter settings for both groups vary across the range of values that were considered. While we have provided conditions in Sections 3 and 4 under which scheduling based on the $c\mu$-rule is optimal, identifying conditions under which most of the performance gains will be due to scheduling or preventive maintenance remains a worthwhile future research direction.

6 Conclusion

In this work, we used a queueing control model to study the problem of how to jointly allocate work and perform preventive maintenance for a flexible server. We identified a condition (Assumption CR) under which it is optimal to schedule according to a state-dependent $c\mu$-rule, as well as an average $c\mu$-rule where the mean service rates are used, when preventive maintenance is not possible (Theorem 2). When Assumption CR does not hold, using these $c\mu$-based scheduling rules may result in an unstable system (Example 1), but our numerical results indicate that it is still possible for such scheduling rules to perform well without Assumption CR (Section 5.3.1).

We then used Theorem 2 to show that, when the preventive maintenance policies are restricted to be age-based, calendar-based, or more generally independent of the queue lengths, it is without loss of optimality to use the aforementioned $c\mu$-based scheduling rules (Theorem 5). In the context of semiconductor manufacturing, the implementation of condition-based maintenance is still very much on the cutting-edge of current research (see e.g., Djurdjanovic [13]), and that age/job-based preventive maintenance policies remain very relevant to practice (see e.g., Yao et al. [28]). Regarding the structure of preventive maintenance policies, we were able to prove that under assumptions analogous to those considered in Kaufman and Lewis [17] for one job class, the monotonicity property of optimal maintenance policies identified in [17, Theorem 3.2] is preserved when there are two job types. In particular, there exists an optimal joint scheduling and maintenance policy where, for each fixed number of class-1 and class-2 jobs, the maintenance decisions are based on a threshold on the server state.

Finally, we presented the results of numerical experiments that compared the performance of $c\mu$-based scheduling with a more naïve scheduling rule (longest-queue-first) and with scheduling based on solving an MDP (Section 5). We observed that, regardless of whether preventive maintenance was performed, the $c\mu$-based scheduling rules are competitive with MDP-based scheduling. Moreover, the worse and more variable performance of longest-queue-first scheduling illustrated the value of incorporating service-rate information. On the other hand, the numerical experiments did not suggest an easy distinction between situations where scheduling has more of a performance impact than preventive maintenance, or vice versa. A better understanding of this distinction, as well as other research directions described in Section 6.1 below, is left for future work.

6.1 Future Work

Our work suggests a number of promising research directions.

Optimality Conditions for the $c\mu$-Rule: Assumption CR, which only involves the service rates, does not depend on the deterioration dynamics of the server. For situations where Assumption CR is too strong, it would be worthwhile to identify conditions on the deterioration process under which $c\mu$-based scheduling remains optimal. Moreover, it may be possible to relax the queue-obliviousness of maintenance policies in Theorem 4. Finally, it would be interesting to determine whether there are any guarantees on the optimality of $c\mu$-based scheduling as a function of some measure of the degree to which Assumption CR is violated.

Good and Implementable Maintenance Policies: The focus of this paper has been on identifying conditions under which it suffices to follow a simple policy for the scheduling decisions. This of course leaves open the question of how maintenance policies should be derived. As was pointed out in Kaufman and Lewis [17], the optimal MDP-based policies can be very complicated. It would therefore be worthwhile to develop maintenance heuristics that both perform well across system parameters of interest, and that are easy to implement.

Relaxing Modeling Assumptions: In many applications, including some in semiconductor manufacturing [5, 7], the assumption that the server deteriorates independently of the work it performs is too strong. It would also be of interest to consider multiple servers and/or stations, or to assume that the server state is only partially observable.

Acknowledgments. The authors thank Chiao-Ju Sun for her help with the numerical experiments. The work of the second author is supported by the Natural Sciences and Engineering Research Council of Canada.

References

[1] S. Andradóttir, H. Ayhan, and D. G. Down. Compensating for failures with flexible servers. *Operations Research*, 55(4):753–768, 2007.

[2] K. Backer, R. J. Huang, M. Lertchaitawee, M. Mancini, and C. Tan. Taking the next leap forward in semiconductor yield improvement. *McKinsey & Company*, May 2018.

[3] R. E. Bohn and C. Terwiesch. The economics of yield-driven processes. *Journal of Operations Management*, 18(1):41–59, 1999.

[4] P. Brémaud. *Point Processes and Queues: Martingale Dynamics*. Springer-Verlag New York, 1981.

[5] Y. Cai, J. J. Hasenbein, E. Kutanoglu, and M. Liao. Single-machine multiple-recipe predictive maintenance. *Probability in the Engineering and Informational Sciences*, 27(2):209–235, 2013.

[6] Y. Cai, E. Kutanoglu, J. Hasenbein, and J. Qin. Single-machine scheduling with advanced process control constraints. *Journal of Scheduling*, 15(2):165–179, 2012.

[7] M. Celen and D. Djurdjanovic. Integrated maintenance decision-making and product sequencing in flexible manufacturing systems. *Journal of Manufacturing Science and Engineering*, 137(4):041006–041006–15, 2015.

[8] M. E. Cholette, M. Celen, D. Djurdjanovic, and J. D. Rasberry. Condition monitoring and operational decision making in semiconductor manufacturing. *IEEE Transactions on Semiconductor Manufacturing*, 26(4):454–464, 2013.

[9] Integrated Circuit Engineering Corporation. Yield and Yield Management. In *Cost Effective IC Manufacturing*. Scottsdale, AZ, 1997.

[10] J. G. Dai. On positive Harris recurrence of multiclass queueing networks: A unified approach via fluid limit models. *The Annals of Applied Probability*, 5(1):49–77, 1995.

[11] J. G. Dai. A fluid limit model criterion for instability of multiclass queueing networks. *The Annals of Applied Probability*, 6(3):751–757, 1996.

[12] J. G. Dai and S. P. Meyn. Stability and convergence of moments for multiclass queueing networks via fluid limit models. *IEEE Transactions on Automatic Control*, 40(11):1889–1904, November 1995.

[13] D. Djurdjanovic. Condition monitoring and operational decision-making in modern semiconductor manufacturing systems. In *Proceedings of the Pacific Rim Statistical Conference for Production Engineering*, ICSA Book Series in Statistics, pages 41–66. Springer, Singapore, 2018.

[14] L. Grigoriu and D. Briskorn. Scheduling jobs and maintenance activities subject to job-dependent machine deteriorations. *Journal of Scheduling*, 20(2):183–197, 2017.

[15] W. J. Hopp. Single Server Queueing Models. In *Building Intuition*, International Series in Operations Research & Management Science, pages 51–79. Springer, Boston, MA, 2008.

[16] S. M. R. Iravani and I. Duenyas. Integrated maintenance and production control of a deteriorating production system. *IIE Transactions*, 34(5):423–435, 2002.

[17] D. L. Kaufman and M. E. Lewis. Machine maintenance with workload considerations. *Naval Research Logistics*, 54(7):750–766, 2007.

[18] J. J. McCall. Maintenance policies for stochastically failing equipment: A survey. *Management Science*, 11(5):493–524, 1965.

[19] P. Nain. Interchange arguments for classical scheduling problems in queues. *Systems & Control Letters*, 12(2):177–184, 1989.

[20] W. P. Pierskalla and J. A. Voelker. A survey of maintenance models: The control and surveillance of deteriorating systems. *Naval Research Logistics*, 23(3):353–388, 1976.

[21] L. I. Sennott. Average cost semi-Markov decision processes and the control of queueing systems. *Probability in the Engineering and Informational Sciences*, 3(2):247–272, 1989.

[22] Y. S. Sherif and M. L. Smith. Optimal maintenance models for systems subject to failure–A review. *Naval Research Logistics*, 28(1):47–74, 1981.

[23] T. W. Sloan and J. G. Shanthikumar. Combined production and maintenance scheduling for a multiple-product, single-machine production system. *Production and Operations Management*, 9(4):379–399, 2000.

[24] C. J. Spanos. Statistical process control in semiconductor manufacturing. *Microelectronic Engineering*, 10(3):271–276, 1991.

[25] C. Valdez-Flores and R. M. Feldman. A survey of preventive maintenance models for stochastically deteriorating single-unit systems. *Naval Research Logistics*, 36(4):419–446, 1989.

[26] C. Wu, D. G. Down, and M. E. Lewis. Heuristics for allocation of reconfigurable resources in a serial line with reliability considerations. *IIE Transactions*, 40(6):595–611, 2008.

[27] C. Wu, M. E. Lewis, and M. Veatch. Dynamic allocation of reconfigurable resources in a two-stage tandem queueing system with reliability considerations. *IEEE Transactions on Automatic Control*, 51(2):309–314, 2006.

[28] X. Yao, E. Fernández-Gaucherand, M. C. Fu, and S. I. Marcus. Optimal preventive maintenance scheduling in semiconductor manufacturing. *IEEE Transactions on Semiconductor Manufacturing*, 17(3):345–356, 2004.

A Appendix

A.1 Instability of Statically Prioritizing Class 1 in Example 1

Consider the (non-idling) policy that always prioritizes class 1 when the server is online. To show that this policy is *unstable*, in the sense that it incurs an infinite long-run expected average cost regardless of the initial state, we consider its associated fluid model.

Let $T_{k,s}(t)$ denote the total amount of time during $[0,t]$ that the server has spent serving class k jobs while it is in state s, and suppose $Q_1(0) = Q_2(0) = 0$. Arguments analogous to those in [11, p. 753] (replace k with k,s) imply that for every sequence $\{q_n, n \geqslant 0\}$ such that $q_n \to \infty$ there exists a subsequence $\{q_m, m \geqslant 0\}$ such that $\lim_{m\to\infty} T_{k,s}(q_m t)/q_m =: \overline{T}_{k,s}(t)$ exists for $k = 1,2$, $s = 1,2$, and $t \geqslant 0$. According to [11, Proposition 3.1], the associated scaled queue lengths $\overline{Q}_k(t) := \lim_{m\to\infty} Q_k(q_m t)/q_m$, $k = 1,2$, satisfy

$$\overline{Q}_k(t) = \lambda_k t - \mu_k^1 \overline{T}_{k,1}(t) - \mu_k^2 \overline{T}_{k,2}(t), \quad k = 1,2,\ t \geqslant 0, \tag{15}$$

where $\lambda_1 = 5$, $\lambda_2 = 0.8$, $\mu_1^1 = \mu_1^2 = 10$, $\mu_2^1 = 1$, and $\mu_2^2 = 2$. In what follows, we will require derivatives of $\overline{T}_{k,s}(t)$ and $\overline{Q}_k(t)$, for $k = 1,2$ and $s = 1,2$. For $t \geqslant s \geqslant 0$, $\overline{T}_{k,s}(t) - \overline{T}_{k,s}(s) \leqslant t - s$, so $\overline{T}_{k,s}(t)$ is Lipschitz continuous. Hence, by (15), $\overline{Q}_k(t)$ is also Lipschitz continuous. As a result, the required derivatives exist almost everywhere.

Since class 1 is prioritized in states $s = 1,2$, and the server is always online (corrective maintenance occurs instantaneously), the server is always busy at class 1 whenever class 1 jobs are present. As a result,

$$\overline{Q}_1(t) > 0 \implies \frac{d}{dt}\overline{T}_{1,1}(t) + \frac{d}{dt}\overline{T}_{1,2}(t) = 1 \implies \frac{d}{dt}\overline{Q}_1(t) = -5 < 0. \tag{16}$$

Note that for a nonnegative continuous function $f(t)$, if $\frac{d}{dt}f(t) < 0$ whenever $f(t) > 0$, then if $f(t_0) = 0$ for some $t_0 \geqslant 0$, $f(t) = 0$ for all $t \geqslant t_0$. As $\overline{Q}_1(0) = 0$, it then follows from (16) and the continuity of $\overline{Q}_1(t)$ that $\overline{Q}_1(t) = 0$ for all $t \geqslant 0$ which, according to (15), implies that $\frac{d}{dt}\overline{T}_{1,1}(t) + \frac{d}{dt}\overline{T}_{1,2}(t) = \frac{1}{2}$.As the service rates for server 1 and the server deterioration rates in states $s = 1, 2$ are identical,

$$\frac{d}{dt}\overline{T}_{1,1}(t) = \frac{d}{dt}\overline{T}_{1,2}(t) = \frac{1}{4}. \tag{17}$$

Letting $\mathbf{1}\{\cdot\}$ denote the indicator function, we have that

$$T_{1,s}(v) + T_{2,s}(v) \leqslant \int_0^v \mathbf{1}\{S(u) = s\}du$$

and taking the fluid limit of both sides yields

$$\overline{T}_{1,s}(t) + \overline{T}_{2,s}(t) \leqslant \frac{t}{2}.$$

Thus,

$$\frac{d}{dt}\overline{T}_{1,s}(t) + \frac{d}{dt}\overline{T}_{2,s}(t) \leqslant \frac{1}{2}, \qquad s = 1, 2. \tag{18}$$

Combining (15), (17), and (18), we conclude that

$$\begin{aligned}\frac{d}{dt}\overline{Q}_2(t) &= 0.8 - (1)\frac{d}{dt}\overline{T}_{2,1}(t) - (2)\frac{d}{dt}\overline{T}_{2,2}(t) \\ &\geqslant 0.8 - (1)\left(\frac{1}{2} - \frac{d}{dt}\overline{T}_{1,1}(t)\right) - (2)\left(\frac{1}{2} - \frac{d}{dt}\overline{T}_{1,2}(t)\right) = 0.05 > 0.\end{aligned}$$

According to [11, Theorem 3.2], this implies that statically prioritizing class 1 is unstable.

A.2 Existence of a Stable Policy in Example 1

Consider the policy that prioritizes class s when the server state is s, for $s = 1, 2$. To show that this policy incurs a finite long-run expected average cost regardless of the initial state, by [12, Theorem 4.1] it suffices to show that its associated fluid model is stable in the sense that it drains and remains empty after a finite amount of time [10].

To define the fluid model, again consider the function $T_{k,s}(t)$ defined in Appendix A.1, and let $q = Q_1(0) + Q_2(0)$. Any limit point as $q \to \infty$ of the scaled process

$$\left(\frac{Q_1(qt)}{q}, \frac{Q_2(qt)}{q}, \frac{T_{1,1}(qt)}{q}, \frac{T_{2,1}(qt)}{q}, \frac{T_{1,2}(qt)}{q}, \frac{T_{2,2}(qt)}{q}\right)$$

is called a *fluid limit* of the original system. Every fluid limit

$$\left(\overline{Q}_1(t), \overline{Q}_2(t), \overline{T}_{1,1}(t), \overline{T}_{1,2}(t), \overline{T}_{2,1}(t), \overline{T}_{2,2}(t)\right)$$

satisfies a set of differential equations known as the *fluid model*. For the system in Example 1 under the proposed policy, the fluid model is:

$$\frac{d}{dt}\overline{Q}_1(t) = \lambda_1 - \mu_1^1 \frac{d}{dt}\overline{T}_{1,1}(t) - \mu_1^2 \frac{d}{dt}\overline{T}_{1,2}(t), \tag{19}$$

$$\frac{d}{dt}\overline{Q}_2(t) = \lambda_2 - \mu_2^1 \frac{d}{dt}\overline{T}_{2,1}(t) - \mu_2^2 \frac{d}{dt}\overline{T}_{2,2}(t), \tag{20}$$

where $\lambda_1 = 5$, $\lambda_2 = 0.8$, $\mu_1^1 = \mu_1^2 = 10$, $\mu_2^1 = 1$, and $\mu_2^2 = 2$.

We now show that, under the proposed policy, every fluid limit is stable. In other words, for every fluid limit there exists a finite time $t_e \geqslant 0$ such that $\overline{Q}_1(t) = \overline{Q}_2(t) = 0$ for all $t \geqslant t_e$. First, recall that the deterioration rates are equal to 1, and that CM occurs instantaneously. An argument similar to that used to derive (18) yields

$$\overline{Q}_1(t) + \overline{Q}_2(t) > 0 \implies \frac{d}{dt}\left[\overline{T}_{1,s}(t) + \overline{T}_{2,s}(t)\right] = \frac{1}{2} \quad \forall s \in \{1,2\}. \tag{21}$$

Recall that class 2 is prioritized when $s = 2$. Hence, according to (21),

$$\overline{Q}_2(t) > 0 \implies \frac{d}{dt}\overline{T}_{2,2}(t) = \frac{1}{2}. \tag{22}$$

Combining (20) with (22), and recalling that $\frac{d}{dt}\mathsf{T}_{2,1}(t) \geqslant 0$ for all t, we conclude that

$$\overline{Q}_2(t) > 0 \implies \frac{d}{dt}\overline{Q}_2(t) \leqslant \lambda_2 - 1 < 0, \tag{23}$$

since $\lambda_2 = 0.8 < 1$. So, as $\overline{Q}_2(t_e) = 0$ for $t_e = \overline{Q}_2(0)/(1-\lambda_2)$, this with (23) yields

$$\overline{Q}_2(t) = 0, \quad t \geqslant t_e. \tag{24}$$

Next, we consider what happens to the fluid in queue 1 after queue 2 has drained. In general, since class 1 is prioritized when the server state $s = 1$, we know from (21) that $\frac{d}{dt}\overline{T}_{1,1}(t) = \frac{1}{2}$ whenever $\overline{Q}_1(t) > 0$. According to (19), this means

$$\overline{Q}_1(t) > 0 \implies \frac{d}{dt}\overline{Q}_1(t) = 5 - 10 \cdot \frac{d}{dt}\overline{T}_{1,2}(t). \tag{25}$$

On the other hand, suppose $t \geqslant t_e$. From (24), we know that $\frac{d}{dt}\overline{Q}_2(t) = 0$. Moreover, since class 1 is prioritized when $s = 1$, we also know that $\frac{d}{dt}\overline{T}_{2,1}(t) = 0$. In light of (20), these two observations imply that $\frac{d}{dt}\overline{T}_{2,2}(t) = \frac{\lambda_2}{2}$. According to (21) and the fact that $\lambda_2 < 1$, this means

$$\frac{d}{dt}\overline{T}_{1,2}(t) = \frac{1-\lambda_2}{2} > 0. \tag{26}$$

We therefore conclude from (25) that

$$t \geqslant t_0 \text{ and } \overline{Q}_1(t) > 0 \implies \frac{d}{dt}\overline{Q}_1(t) < 0. \tag{27}$$

In summary, (24) and (27) imply that both queues drain and remain empty after a finite amount of time, i.e., that the fluid model is stable.

A.3 Proof of Theorem 6

In this section, we assume that Assumptions M and S hold. Under Assumption M, the joint scheduling and maintenance model described in Section 2 is a *semi-Markov* *decision process (SMDP)*; for background on SMDPs, see e.g., Sennott [21] and the references therein.

An SMDP is defined by the following objects:

1. the state set $\mathbb{X}$,
2. sets of available actions $A(x)$ for each $x \in \mathbb{X}$,
3. transition probabilities $p(y|x, a)$ for each $x, y \in \mathbb{X}$ and $a \in A(x)$,
4. distributions $F(\cdot|x, a, y)$ for the time spent in each state $x \in \mathbb{X}$ given that action $a \in A(x)$ is taken and the next state of the process is $y \in \mathbb{X}$,
5. immediate costs $D(x, a)$ and cost rates $d(x, a)$ for each $x \in \mathbb{X}$ and $a \in A(x)$.

Recalling that we are only considering nonidling policies, for the joint scheduling and maintenance problem the above objects are defined as follows.

1. $\mathbb{X} = \{0, 1, \dots\}^2 \times \{0, 1, \dots, S\}$;
2. letting $k = 0, 1, 2$ respectively denote idling, serving class 1, and serving class 2, and letting PM and CM respectively denote initiating preventive and corrective maintenance, for $(i_1, i_2, s) \in \mathbb{X}$ let

$$A(i_1, i_2, s) = \begin{cases} \{CM\}, & \text{if } s = 0; \\ \{0, PM\}, & \text{if } i_1, i_2 = 0,\ s \geqslant 1; \\ \{1, PM\}, & \text{if } i_1 \geqslant 1,\ i_2 = 0,\ s \geqslant 1; \\ \{2, PM\}, & \text{if } i_1 = 0,\ i_2 \geqslant 1,\ s \geqslant 1; \\ \{1, 2, PM\}, & \text{if } i_1, i_2 \geqslant 1,\ s \geqslant 1; \end{cases} \tag{28}$$

3. for $(i_1, i_2, s), y \in \mathbb{X}$ and $a \in A(i_1, i_2, s)$, letting $\mathbf{e}_k$ be the vector in $\mathbb{R}^3$ where the k^{th} entry is a 1 and all other are zero and $\mu_0^s \equiv 0$, and recalling that by Assumption M(iii) the maintenance times are iid with distribution $G(\cdot)$,

$$p(y|(i_1,i_2,s),a) = \begin{cases} \int_0^\infty \frac{e^{-\lambda_1 t}(\lambda_1 t)^{n_1}}{n_1!} \frac{e^{-\lambda_2 t}(\lambda_2 t)^{n_2}}{n_2!}\, dG(t) \\ \qquad \text{if } s = 0, a = CM, y = (i_1+n_1, i_2+n_2, S) \\ \qquad \text{or } s \geqslant 1, a = PM, y = (i_1+n_1, i_2+n_2, S); \\ \frac{\lambda_1}{\lambda_1+\lambda_2+\mu_k^s+\alpha_s} & \text{if } s \geqslant 1, a = k, y = (i_1+1, i_2, s); \\ \frac{\lambda_2}{\lambda_1+\lambda_2+\mu_k^s+\alpha_s} & \text{if } s \geqslant 1, a = k, y = (i_1, i_2+1, s); \\ \frac{\alpha_s}{\lambda_1+\lambda_2+\mu_k^s+\alpha_s} & \text{if } s \geqslant 1, a = k, y = (i_1, i_2, s-1); \\ \frac{\mu_k^s}{\lambda_1+\lambda_2+\mu_k^s+\alpha_s} & \text{if } s \geqslant 1, a = k, y = (i_1, i_2, s) - \mathbf{e}_k; \end{cases}$$

4. for $(i_1, i_2, s), y := (j_1, j_2, u) \in \mathbb{X}$ and $a \in A(x)$,

$$F(t \mid (i_1,i_2,s),a,y) = \begin{cases} G(t) & \text{if } s = 0, a = CM, j_1 \geqslant i_1, j_2 \geqslant i_2, u = S \\ & \text{or } s \geqslant 1, a = PM, j_1 \geqslant i_1, j_2 \geqslant i_2, u = S; \\ 1 - e^{-\lambda_1 t} & \text{if } s \geqslant 1, a \in \{0,1,2\}, (j_1,j_2,u) = (i_1+1, i_2, s); \\ 1 - e^{-\lambda_2 t} & \text{if } s \geqslant 1, a \in \{0,1,2\}, (j_1,j_2,u) = (i_1, i_2+1, s); \\ 1 - e^{-\alpha_s t} & \text{if } s \geqslant 1, a \in \{0,1,2\}, (j_1,j_2,u) = (i_1, i_2, s-1); \\ 1 - e^{-\mu_1^s t} & \text{if } s \geqslant 1, a = 1, (j_1,j_2,u) = (i_1-1, i_2, s); \\ 1 - e^{-\mu_2^s t} & \text{if } s \geqslant 1, a = 2, (j_1,j_2,u) = (i_1, i_2-1, s); \end{cases}$$

5. for $(i_1, i_2, s) \in \mathbb{X}$ and $a \in A(i_1, i_2, s)$,

$$D((i_1,i_2,s),a) = \begin{cases} K_c & \text{if } a = CM; \\ K_p & \text{if } a = PM; \\ 0 & \text{otherwise}; \end{cases}$$

and

$$d((i_1,i_2,s),a) = c_1 i_1 + c_2 i_2.$$

It is useful to consider discounting the expected total cost incurred over an infinite horizon. In particular, given a *discount rate* $\beta > 0$, the expected β-*discounted cost* incurred from the initial state $(i_1, i_2, s) \in \mathbb{X}$ under the policy $\pi \in \Pi$ is

$$v_\beta^\pi(i_1, i_2, s) := \mathbb{E}\left[\sum_{n: t_n^\pi \leqslant t} e^{-\beta t_n^\pi}\left[K_c M_c^\pi(t_n^\pi) + K_p M_p^\pi(t_n^\pi)\right] + \int_0^\infty e^{-\beta t} \sum_{k=1}^{2} c_k Q_k^\pi(t)\, dt \,\middle|\, Q_1^\pi(0) = i_1,\ Q_2^\pi(0) = i_2,\ S^\pi(0) = s\right].$$

Moreover, a policy π_* is β-*optimal* if $v_\beta^{\pi_*}(x) = \inf_{\pi \in \Pi} v_\beta^\pi(x) =: v_\beta(x)$ for all $x \in \mathbb{X}$.

Definition 9. *A function* $v : \mathbb{X} \to \mathbb{R}$ *is* monotone in the system state *if*

$$i_1 \leqslant i_1',\ i_2 \leqslant i_2', s \geqslant s' \implies v(i_1, i_2, s) \leqslant v(i_1', i_2', s').$$

A straightforward adaptation of the sample-path argument in [17, Proof of Proposition 3.3] can be used to prove the following useful monotonicity property of v_β.

Proposition 10. *The value function v_β is monotone in the system state.*

Lemma 11. *Assumptions M and S imply that the hypotheses of [21, Theorem 2, Proposition 4] hold.*

Proof. The hypotheses of [21, Theorem 2] consist of [21, Assumptions 1-5].

1. For $t \geqslant 0$, $x, y \in \mathbb{X}$, and $a \in A(x)$, let

$$H(t|x, a) := \sum_{y \in \mathbb{X}} p(y|x, a) F(t|x, a, y).$$

The first assumption states that there exist $\epsilon, \delta > 0$ such that

$$1 - H(\delta|x, a) \geqslant \epsilon \qquad \forall x \in \mathbb{X},\ a \in A(x). \tag{29}$$

First, recall that according to Assumption M(iii), $1/\alpha_0 = \int_0^\infty t dG(t) > 0$. This implies that there exists a $\delta^* > 0$ such that $1 - G(\delta^*) > 0$. Moreover, letting

$$\overline{\gamma} := \max\{\lambda_1, \lambda_2, \alpha_1, \ldots, \alpha_B, \mu_1^1, \ldots, \mu_1^S, \mu_2^1, \ldots, \mu_2^S\} > 0$$

and

$$\epsilon^* := \min\{1 - G(\delta^*), e^{-\overline{\gamma}\delta^*}\} > 0,$$

it follows that (29) holds with $\epsilon = \epsilon^*$ and $\delta = \delta^*$.

2. For $x \in \mathbb{X}$ and $a \in A(x)$, let

$$\tau(x, a) := \sum_{y \in \mathbb{X}} p(y|x, a) \int_0^\infty t dF(t|x, a, y). \tag{30}$$

The second assumption states that there exists a constant $B < \infty$ such that

$$\tau(x, a) \leqslant B \qquad \forall x \in \mathbb{X},\ a \in A(x). \tag{31}$$

Letting

$$\underline{\gamma} := \min\{\lambda_1, \lambda_2, \alpha_1, \ldots, \alpha_B, \mu_1^1, \ldots, \mu_1^S, \mu_2^1, \ldots, \mu_2^S\} > 0$$

and

$$B^* := \max\{1/\alpha_0, 1/\underline{\gamma}\} < \infty,$$

it follows that (31) holds with $B = B^*$.

3. The third assumption states that

$$v_\beta(x) < \infty \qquad \forall \beta > 0,\ x \in \mathbb{X}. \tag{32}$$

According to [21, Remark 1], a sufficient condition for (32) to hold is the existence of a policy π such that

$$w^\pi(x) < \infty \qquad \forall x \in \mathbb{X}. \tag{33}$$

Let s^* be a state that satisfies Assumption S(i). By analyzing a fluid model analogous to the one in [17, Proof of Proposition 3.1], it can be shown that (33) is satisfied by any policy that initiates PM whenever the server state is less than s^*, and otherwise does not idle an online server if the system is nonempty. Hence (32) holds.

4. Let $\mathbf{0} := (0, 0, S)$, and

$$h_\beta(x) := v_\beta(x) - v_\beta(\mathbf{0}), \qquad x \in \mathbb{X}.$$

The fourth assumption states that there exists a $\beta_0 > 0$ and $M : \mathbb{X} \to [0, \infty)$ such that

$$h_\beta(x) \leqslant M(x) \qquad \forall \beta \in (0, \beta_0),\ x \in \mathbb{X} \tag{34}$$

and

$$\exists a(x) \in A(x) \text{ such that } \sum_{y \in \mathbb{X}} p(y|x, a(x)) M(y) < \infty \qquad \forall x \in \mathbb{X}. \tag{35}$$

Let $X^\pi(t) := (Q_1^\pi(t), Q_2^\pi(t), S^\pi(t))$ denote the state of the system at time t under the policy π. For $z \in \mathbb{X}$, let $\tau_z^\pi := \inf\{t > 0 \mid X^\pi(t) = z\}$ and, for $x, y \in \mathbb{X}$, let

$$C^\pi(x, y) := \mathbb{E}\left[\sum_{n : t_n^\pi \leqslant \tau_y^\pi} \left[K_c M_c^\pi(t_n^\pi) + K_p M_p^\pi(t_n^\pi) + \int_0^{\tau_y^\pi} \sum_{k=1}^{2} c_k Q_k^\pi(t)\, dt \,\right| X^\pi(0) = x\right].$$

denote the expected total cost incurred up to a first passage from x to y under the policy π. According to [21, Remark 1], a sufficient condition for (34) and (35) to hold for some $\beta_0 > 0$, $M : \mathbb{X} \to [0, \infty)$ is the existence of a stationary policy φ such that

$$C^\varphi(x, \mathbf{0}) < \infty \qquad \forall x \in \mathbb{X}. \tag{36}$$

Let s^* be a state that satisfies Assumption S(i), and let φ_{s^*} be any stationary policy that initiates PM whenever the server state is less than s^*, and otherwise does not idle an online server if the system is nonempty. By analyzing a fluid model analogous to the one in [17, Proof of Proposition 3.1], it can be shown that φ_{s^*} satisfies (33), and that the embedded state process under φ_{s^*} is a unichain Markov chain where the set of states $\{(i, j, s) \mid s^* \leqslant s \leqslant S\}$ is the ergodic class and the remaining states are transient. By [21, Lemma 2], it follows that (36) holds with $\varphi = \varphi_{s^*}$. Hence there exist $\beta_0 > 0$ and $M : \mathbb{X} \to [0, \infty)$ such that (34) and (35) hold.

5. The fifth assumption states that there exist a $\beta_0 > 0$ and $N \geqslant 0$ such that

$$-N \leqslant h_\beta(x) \qquad \forall \beta \in (0, \beta_0),\ x \in \mathbb{X}. \tag{37}$$

Since $h_\beta(x) = \nu_\beta(x) - \nu_\beta(\mathbf{0})$ for $x \in \mathbb{X}$, and $\mathbf{0} = (0, 0, S)$, it follows from Proposition 10 that (37) holds with $N = 0$ and any $\beta_0 > 0$.

Next, the hypotheses of [21, Proposition 4] consist of [21, Assumptions 1-5] and the following assumption: there exist $\epsilon > 0$ and a finite set $G \subset \mathbb{X}$ such that

$$\min_{a \in A(x)} d(x, a) \geqslant \frac{B(g + \epsilon)}{\inf_{x,a} \tau(x, a)} \qquad \forall x \in \mathbb{X} \setminus G \tag{38}$$

where g is a constant from [21, Theorem 2], and $\inf_{x,a} \tau(x, a) > 0$ by [21, Lemma 1]. Recalling that

$$d((i_1, i_2, s), a) = c_1 i_1 + c_2 i_2, \qquad (i_1, i_2, s) \in \mathbb{X},\ a \in A(i_1, i_2, s),$$

consider any $\epsilon^* > 0$ and let

$$G^* := \left\{ (i_1, 0, s) \in \mathbb{X} \;\middle|\; i_1 < \left\lfloor \frac{B(g + \epsilon^*)}{c_1 \inf_{x,a} \tau(x, a)} \right\rfloor \right\}.$$

Then $|G^*| < \infty$, and (38) holds with $\epsilon^* = \epsilon$ and $G^* = G$. $\square$

For $f : \mathbb{X} \to \mathbb{R}$, $x \in \mathbb{X}$, and $a \in A(x)$, let

$$\begin{aligned} T^{a}_{\beta} f(x) := D(x,a) + d(x,a) \sum_{y \in \mathbb{X}} p(y|x,a) \int_0^{\infty} \int_0^{t} e^{-\beta u}\, du\, dF(t|x,a,y) \\ + \sum_{y \in \mathbb{X}} p(y|x,a) \int_0^{\infty} e^{-\beta t}\, dF(t|x,a,y)\, f(y) \end{aligned}$$

Theorem 12. *Under Assumptions M and S, the following statements hold.*

(i) The value function v_β satisfies the discounted-cost optimality equation (DCOE)

$$v_\beta(x) = \min_{a \in A(x)} T^{a}_{\beta} v_\beta(x) \qquad \forall x \in \mathbb{X}. \tag{39}$$

(ii) For every $\beta > 0$ there exists a β-optimal deterministic stationary policy π_β.

(iii) A deterministic stationary policy π is β-optimal if and only if

$$\pi(x) \in \arg\min_{a \in A(x)} T^{a}_{\beta} v_\beta(x) \qquad \forall x \in \mathbb{X}.$$

(iv) Every β-optimal deterministic stationary policy is monotone in the server state.

Proof. According to Lemma 11, [21, Assumptions 1,3] hold, which implies that statements (i)-(iii) hold by [21, Theorem 1].

Next, suppose that it is not β-optimal to perform PM in state (i_1, i_2, s). Then by statement (iii),

$$T^{PM}_{\beta} v_\beta(i_1, i_2, s) > v_\beta(i_1, i_2, s).$$

Since PM incurs the same fixed cost whenever it is initiated, and the subsequent maintenance times are iid, it follows from Proposition 10 that

$$T^{PM}_{\beta} v_\beta(i_1, i_2, s+1) = T^{PM}_{\beta} v_\beta(i_1, i_2, s) > v_\beta(i_1, i_2, s) \geqslant v_\beta(i_1, i_2, s+1).$$

By statement (iii), this implies that it is also not β-optimal to perform PM in state $(i_1, i_2, s+1)$. Hence statement (iv) holds. □

Proof of Theorem 6. Lemma 11 implies that [21, Theorem 2, Proposition 4] hold for the SMDP formulated in this section. In particular, [21, Theorem 2, Proposition 4] state that there exists a deterministic stationary optimal policy π_* that is a limit point of a sequence of β-optimal deterministic stationary policies. Since the action sets are finite, it follows that π_* is actually β-optimal for some $\beta > 0$. According to Theorem 12(iv), π_* is monotone in the server state. $\square$