

Energy-Aware Scheduling on Heterogeneous Processors

Osman T. Akgun, Douglas G. Down, *Senior Member, IEEE*, and Rhonda Righter

Abstract—We study a multiple-server system where servers are heterogeneous in terms of both their speeds and their usage costs. We show, for a clearing system, that the optimal control policy is threshold type, and then we extend this result to a system with arrivals. We consider both the case with reassignment, where jobs can be removed from a server and assigned to a different server or placed in the queue at any time, and the case where reassignment is not possible, so jobs once assigned to a server cannot be preempted. Determining which server is preferred (i.e., the one that is used even if there are very few jobs waiting) is surprisingly difficult, and depends on the arrival rate.

Index Terms—Controlled queueing system, heterogeneous servers, processor scheduling, energy-aware systems.

I. INTRODUCTION

ENERGY consumption has been growing exponentially in computers and computer centers [1], embedded systems, portable devices, etc., and it is also a critical concern in battery-operated devices such as sensors. Indeed, embedded systems are being designed to have heterogeneous processing elements to reduce power consumption [2]. Energy cost has become the critical cost factor for server farms, and there is also increasing concern over the heat and carbon emissions generated [3]. No longer is faster necessarily better; both speed and energy consumption must be considered in the operation of these systems, and often faster servers require more energy. Hence the emphasis is on providing good service while maintaining energy efficiency. We study a multiple-server system where servers are heterogeneous in terms of both their speeds and their usage costs. Including usage costs is also a means of improving fairness in systems where the servers are humans, such as call centers [4]. Applying a higher charge to a faster call center agent can help balance workload in a fairer manner across agents with heterogeneous speeds.

The heterogeneous server problem has been studied in the literature in terms of both admission control and scheduling control. Admission control has received a lot of attention in the operations management community (see, e.g., Naoar [5], Yechiali [6], Stidham [7], and Kulkarni and Gautam [8]).

We consider optimal scheduling of homogeneous jobs on heterogeneous servers, i.e., deciding whether to assign jobs to an available server in order to minimize mean total holding

costs and server usage costs. The jobs waiting in the system (including those receiving service) incur a holding cost and the jobs using the servers incur a usage cost (different for each server) per unit time. Usage costs are not incurred when servers are idle. Service times are exponentially distributed, and customers arrive according to a Poisson process. The case with only holding costs and different server speeds (the slow server problem) has been extensively studied with different approaches. Weiss and Pinedo [9] considered the slow server problem where reassignment is possible, that is, the jobs can be reassigned to other servers at any given time even when they are in service. They showed that the optimal policy is to use all the servers if there are more jobs than servers, and to otherwise serve all the jobs on the fastest servers. More work has been done on the harder problem in which jobs cannot be reassigned. Agrawala et al. [10] showed that with no arrivals, the optimal policy is threshold type (with a threshold on the number of jobs determining whether a server is used), and the threshold for a given server is independent of the state of the slower servers. With arrivals and only two servers, the optimal policy is again of threshold type. Lin and Kumar [11] and Koole [12] showed this result using dynamic programming approaches. Walrand [13] showed the same result with sample path arguments. Viniotis and Ephremides [14] extended the results to fairly general arrival processes and service-time distributions. Stockbridge [15] used a martingale approach and linear programming to show the same result and also showed that with a finite buffer, the optimal policy might not be a threshold policy. When there are more than two servers and no reassignment, the problem turns out to be much more difficult. Rosberg and Makowski [16] showed that for sufficiently small arrival rates, the optimal policy is the same as the case with no arrivals. For more general arrival rates Weber [17] showed that, in contrast to the no arrival case, if there exists an optimal threshold policy, the optimal threshold for using a server depends on the state of the slower servers. Kim, Ahn and Righter [18] showed that a threshold policy for primary jobs is optimal when there is an infinite number of secondary jobs that are assigned to servers that are not used for primary jobs.

In the literature the problem with server usage costs also usually includes server setup costs (costs incurred for turning a server on or off). For two homogeneous servers, Bell [19] proved that the optimal policy is a hysteresis policy, which has different upper and lower threshold values, one for turning on a server and one for turning it off. For more than two servers or for heterogeneous servers, characterizing the optimal policy

Manuscript received August 28, 2011.

O. T. Akgun and R. Righter are with the Department of Industrial Engineering and Operations Research, University of California, Berkeley, CA, 94720 USA (e-mail: akguno@ieor.berkeley.edu).

D. G. Down is with the Department of Computing and Software, McMaster University, Hamilton, ON, L8S 4L7, Canada.

is much more difficult. Instead, the optimality of a hysteresis policy is assumed. Ibe and Kleison [20] derived closed form solutions for more than two homogeneous servers and for two heterogeneous servers using the Green's function method. Liu and Golubchik [21] derived the steady state distribution for the number of jobs for homogeneous and heterogeneous servers. Zhang and Phillis [22] used a fuzzy logic controller to calculate threshold values. There has also been some work in the area of dynamically changing the speeds (speed scaling) of individual servers in the presence of speed-dependent usage costs [23], [24]. We assume that particular servers have fixed speeds and usage costs, and there are no setup costs.

Without setup costs the problem is still difficult. Weiss and Pinedo [9] studied minimizing the server usage costs in a system without arrivals and where reassignment is possible, but the policies they consider exclude threshold-type policies. Rykov and Efrosinin [25] studied the system without reassignment and with both server usage costs and holding costs, but these costs are agreeable in the sense that faster servers have smaller expected usage costs. This assumption simplifies the problem considerably, because the server preference index becomes trivial.

Xu and Shanthikumar [26] used the relation between social optimality and individual optimality to show that, for admission control, the optimal policy is a threshold type. Using this approach, Xu [27] showed the earlier known threshold result for two-server scheduling control. We use the same approach in this paper. The idea is to find the individually optimal policy for a dual system, in which the individually and socially optimal policies are the same, and then relate these results to the original system. Equivalence of the individually and the socially optimal policies also gives us the tools to compute the optimal threshold.

Showing that the individually optimal policy is threshold type is similar to the problem with only holding costs in most of the cases we consider. The difficulty caused by the addition of the server usage costs emerges when we try to calculate the values for optimal thresholds and related expected costs, and even in determining the server preference relation. Without usage costs, it is simple to show that faster servers are always preferred. So, for example, when it is optimal to only use one server, it will be the fastest one. In our case, however, the preference relation among servers is much more complicated. Even when we assume there are no arrivals and we allow reassignment, the preference relations are complicated expressions that depend on all the server rates and usage costs. When we allow arrivals and consider only two servers, we show, surprisingly, that the preference relation between the two servers depends on the arrival rate. Indeed, for a system with a finite number of arrivals, the preferred server can change with the state of the system.

The organization of the paper is as follows. In Section II, we study the clearing system (no arrivals) and in Section III we look at the system with arrivals. In both sections, we analyze two different problems, one in which jobs can be reassigned from one server to another or removed from a server and put in the queue at no cost, and one in which after being assigned to a server a job cannot be removed

from that server. For all cases, we show the equivalence of the socially and the individually optimal (threshold) policies, except for the problem with arrivals, without reassignment, and more than two servers. In that case we give a counterexample showing that the individually and socially optimal policies are not the same. For more than two servers with arrivals and with reassignment, we are able to show the equivalence of the socially and individually optimal policies, but in this case, the actual server thresholds are very difficult to derive. For the other cases we use the individually optimal policies to calculate the optimal thresholds.

II. THE CLEARING SYSTEM

We first consider a multi-server queueing system with s parallel servers, where at time 0 there exists a certain number of homogeneous jobs that need to be processed. In particular, there are no new arrivals (i.e., a clearing system). The service times are independently and exponentially distributed with rate μ_j on the j th server, $j = 1, \dots, s$. Whenever a job is processed on server j , it incurs a cost with rate β_j . The jobs waiting in the system incur a holding cost with rate h . We are going to find the scheduling policy (assigning jobs to servers) that minimizes the expected total cost incurred until all jobs are processed (or equivalently minimizes the per job average cost). We will consider two problems, one in which jobs can be reassigned from one server to another or removed from a server and put in the queue at no cost, and one in which after being assigned to a server a job cannot be removed from that server (no reassignment). We first consider the individually optimal policy for the two problems.

A. The Individually Optimal Policy

In this section, we consider the individually optimal policy in which the jobs are given arbitrary priorities, and each attempts to minimize its own holding cost and server usage cost. A job pays holding cost per unit time, h , until it leaves the system and it also pays server usage cost per unit time, β_j , while it is being served by server j .

1) *Problem without reassignment:* We first consider the problem without reassignment, which, unlike the slow server problem, is surprisingly easier and will of course lead to higher cost than the problem with reassignment. We assume that at time 0 all the jobs that are waiting in the queue are arbitrarily assigned priorities. Servers which are not busy are offered to the unassigned job with the highest priority; if that job chooses to wait for one of the busy servers, then the job with the second highest priority will be offered the server. The process continues until all servers are busy or all the jobs have been considered for assignment. When there is a departure, the assignment process will be repeated for the unassigned jobs. If a job chooses to use a server, it must stay with that server until service completion.

We show that the individually optimal policy is a threshold policy. A threshold policy means that there is a preference order for the servers for all possible states and the most preferred available server is used if the number waiting in the queue is greater than a threshold value for that server.

In particular, the threshold-valued job will use the server. Therefore, the threshold value for each server, t_j , will be increasing in the order of server preference. Throughout we use increasing, etc., in the nonstrict sense. Let us label the servers according to the following index:

$$\frac{h + \beta_j}{\mu_j} \leq \frac{h + \beta_{j+1}}{\mu_{j+1}}, j = 1, \dots, s - 1.$$

Note that in the classic slow server problem the labeling is in decreasing order of server speed.

The next theorem shows that the above ordering is the optimal preference order for the servers and that the individually optimal policy is a threshold-type policy. Let $M_j = \sum_{k=1}^j \mu_k$, $B_j = \sum_{k=1}^j \beta_k$, and let t_j denote the threshold value for using server j (if all servers $1, \dots, j - 1$ are busy and the j th server is available at the time of assignment then the t_j th job in the queue will be assigned to the j th server, and $t_j \leq t_{j+1}, j = 1, \dots, s - 1$). Let $\delta_j \in \{0, 1\}$ denote whether the j th server is available at the time of assignment or not and let $\delta = (\delta_1, \dots, \delta_s)$ be the overall server state. Let $g : \{0, 1\}^s \rightarrow \{1, \dots, s\}$ be a map such that $g(\delta) = \min\{j : \delta_j = 0\}$, so g returns the first zero (idle) component in δ , and let e_k denote the k th unit vector in \mathbf{R}^s . Let $v_i(\delta)$ be the individually optimal expected cost for the job that has i th priority among the jobs waiting to be assigned to a server, given the server state.

Theorem 1: Without reassignment, the individually optimal policy is threshold type, so job t_j will use the j th server if it is the available server with the lowest index ($g(\delta) = j$). The threshold values and the individually optimal expected costs are given by

$$t_1 = 1,$$

$$t_j = \max \left\{ 1, \left\lceil C_j + \frac{M_{j-1}}{\mu_j} - (j - 1) \right\rceil \right\}, j = 2, \dots, s, \quad (1)$$

$$v_i(\delta) = \begin{cases} \frac{(i+j)h + B_j}{M_j} & \text{for } t_j \leq i < t_{j+1}, \\ & j < g(\delta) \\ \frac{h + \beta_{g(\delta)}}{\mu_{g(\delta)}} & \text{for } i = t_{g(\delta)} \\ v_{i-1}(\delta + e_{g(\delta)}) & \text{for } i > t_{g(\delta)}, \end{cases} \quad (2)$$

where $C_j = \frac{\beta_j M_{j-1} - \mu_j B_{j-1}}{h \mu_j}$.

Proof: It is clear that if a job is offered more than one server it will only consider the one with the smallest index, because the index at the j th server, $\frac{h + \beta_j}{\mu_j}$, is the cost of using the server assuming no reassignment. This also gives us a threshold policy for the individually optimal policy, because of the way servers are offered to jobs. We also have that $v_i(\delta)$ is increasing in i , so if a server is rejected by a job, it will never be used by that job in the future. We first show (2). The second and third cases are true trivially by definition of the threshold. We prove the first case by induction on j . For $j = 1$, the i th job, where $t_1 \leq i < t_2$, will wait for the first server, paying an expected holding cost of $\frac{ih}{\mu_1}$ and then will pay an expected cost of $\frac{h + \beta_1}{\mu_1}$ while in service, so the first

case is true for $j = 1$. Suppose that the first case of (2) is true for $j - 1$. Consider the i th job, where $i = t_j$ and $j < g(\delta)$.

$$\begin{aligned} v_i(\delta) &= \frac{h + \sum_{k=1}^j \mu_k v_i(\delta - e_k) + \sum_{k=j+1}^n \mu_k v_i(\delta)}{M_n} \\ &= \frac{h + \sum_{k=1}^j \mu_k v_i(\delta - e_k)}{M_j} \\ &= \frac{h + \mu_j \frac{h + \beta_j}{\mu_j} + \sum_{k=1}^{j-1} \mu_k v_{i-1}(\delta - e_k + e_k)}{M_j} \\ &= \frac{2h + \beta_j + M_{j-1} v_{i-1}(\delta)}{M_j} \\ &= \frac{2h + \beta_j + (i - 1 + (j - 1))h + B_{j-1}}{M_j} \\ &= \frac{(i + j)h + B_j}{M_j} \end{aligned}$$

where the third equality follows from the second and third cases of (2) and the fifth equality follows by induction for the first case of (2). Finally the i th job, where $t_j < i < t_{j+1}$, will wait for one of the servers $1, \dots, j$ to be available, paying an expected holding cost of $\frac{(i-t_j)h}{M_j}$ until becoming the t_j th job waiting to be assigned. Therefore

$$\begin{aligned} v_i(\delta) &= \frac{(i - t_j)h}{M_j} + v_{t_j}(\delta) = \frac{(i - t_j)h}{M_j} + \frac{(t_j + j)h + B_j}{M_j} \\ &= \frac{(i + j)h + B_j}{M_j} \end{aligned}$$

and the proof of (2) is complete.

Next we prove (1), again by induction on j . First note that idling is not optimal so $t_1 = 1$. Suppose (1) is true for $j - 1$. Consider the i th job waiting to be assigned, where $i > t_{j-1}$ and $g(\delta) = j$. If the i th job waits for one of the servers $1, \dots, j - 1$ to be available, then its expected cost will be $\frac{(i+j-1)h + B_{j-1}}{M_{j-1}}$ by (2). If it uses the j th server instead, then its expected cost will be $\frac{h + \mu_j}{\beta_j}$. Some algebra yields that the i th job will use the j th server if and only if (except in the case of ties, where we assume the j th server is used),

$$i \geq \frac{(h + \beta_j)M_{j-1} - \mu_j[(j - 1)h + B_{j-1}]}{h \mu_j}.$$

Hence t_j satisfies (1) and the proof is complete. ■

Note that Theorem 1 gives us an interesting interpretation of the expected costs for individual jobs. In particular, for a job i that would accept any of the first j servers, but none of the others ($t_j \leq i < t_{j+1}$), if the j acceptable servers are all busy, the expected cost for that job is the same as if it were the $i + j$ th job in a single-server system where the usage cost for the server is the sum of the costs for the first j servers, and the speed is the sum of the speeds of the first j servers.

We have the following corollary, which gives the classical slow server result [10] for $\beta_j = 0$.

Corollary 2: When $\beta_j = 0$ for all $1 \leq j \leq s$, the individually optimal policy is threshold type. The threshold values and the individually optimal expected costs are given by

$$t_j = \max \left\{ 1, \left\lceil \frac{M_{j-1}}{\mu_j} - (j - 1) \right\rceil \right\}, j = 1, \dots, s,$$

$$v_i(\delta) = \begin{cases} \frac{(i+j)h}{M_{j-1}} & \text{for } t_j \leq i < t_{j+1}, \\ & j < g(\delta) \\ \frac{h}{\mu_{g(\delta)}} & \text{for } i = t_{g(\delta)} \\ v_{i-1}(\delta + e_{g(\delta)}) & \text{for } i > t_{g(\delta)}. \end{cases}$$

2) *Problem with reassignment*: In this section we consider the case where reassignment is possible, that is, the jobs that are already using servers can be reassigned before completing service. Now we assume a priority ordering on *all* jobs, including those being served. Let v_i be the individually optimal expected cost for the job that has i th priority. Here we do not need the state of the servers because we are allowing reassignment. That is, each time we make a decision we can assume all jobs are in the queue. It is intuitive to think that the servers' preference index will still be $\frac{h+\beta_j}{\mu_j}$, because this is obviously the value of the server for the first job. However, other jobs' preferences for the other servers depend on their priority position. We give an example that illustrates the added complexity.

Example 3: Suppose there are two jobs and three servers with parameters $(h, \beta_1, \beta_2, \beta_3, \mu_1, \mu_2, \mu_3) = (2, 2, 5, 2, 2, 2, 1)$. Therefore $\frac{h+\beta_1}{\mu_1} < \frac{h+\beta_2}{\mu_2} < \frac{h+\beta_3}{\mu_3}$ and

$$v_1 = \min_i \frac{h + \beta_i}{\mu_i} = \frac{h + \beta_1}{\mu_1} = 2.$$

Now consider the second job. If it waits for the first server, its individual expected cost will be $\frac{2h+\beta_1}{\mu_1}$. If it starts service on the second server, then its individual expected cost will be

$$\frac{h + \beta_2}{\mu_1 + \mu_2} + \frac{\mu_1}{\mu_1 + \mu_2} v_1.$$

The first term is the cost incurred until a service completion occurs at one of servers 1 or 2, and the second term is the additional cost incurred if the service completion happens at the first server. (The job can be reassigned, i.e., leave the second server, so if the first server finishes before the second, the job will now be the first job waiting and will use the first server). Plugging in v_1 yields $\frac{2h+\beta_1+\beta_2}{\mu_1+\mu_2}$ for the cost to the second job if it uses the second server until it completes or can move to the first server. Similarly if the second job starts service on the third server, then its individual expected cost will be $\frac{2h+\beta_1+\beta_3}{\mu_1+\mu_3}$. Hence,

$$\begin{aligned} v_2 &= \min \left\{ \frac{2h + \beta_1}{\mu_1}, \frac{2h + \beta_1 + \beta_2}{\mu_1 + \mu_2}, \frac{2h + \beta_1 + \beta_3}{\mu_1 + \mu_3} \right\} \\ &= \min \left\{ 3, \frac{11}{4}, \frac{8}{3} \right\} = \frac{8}{3} \\ &= \frac{2h + \beta_1 + \beta_3}{\mu_1 + \mu_3}, \end{aligned}$$

and the second job will prefer to use the third server rather than the second. Therefore, the second job's preference is not consistent with the $\frac{h+\beta_i}{\mu_j}$ preference index.

Let Π denote an arbitrary permutation of the servers, that is, Π is a one-to-one map from $\{1, \dots, s\}$ to $\{1, \dots, s\}$. Let $M_j(\Pi) = \sum_{k=1}^j \mu_{\Pi(k)}$, $B_j(\Pi) = \sum_{k=1}^j \beta_{\Pi(k)}$ and

$C_j^k(\Pi) = \frac{\beta_k M_{j-1}(\Pi) - \mu_k B_{j-1}(\Pi)}{h \mu_k}$. We recursively define a preference ordering, Π^* , as follows.

$$\Pi^*(1) = \arg \min_{k \in \{1, \dots, s\}} \frac{h + \beta_k}{\mu_k},$$

and

$$\Pi^*(j) = \arg \min_{k \in \{j, \dots, s\}} \frac{\max \{j, [C_j^k(\Pi^*)]\} h + B_{j-1}(\Pi^*) + \beta_k}{M_{j-1}(\Pi^*) + \mu_k}$$

for $j = 2, \dots, s$. The next theorem shows that Π^* is the optimal ordering and that the individually optimal policy is a threshold-type policy.

Theorem 4: With reassignment the individually optimal policy is threshold type and the optimal preference relation for the servers is defined by Π^* . The threshold values and the individually optimal expected costs are given by

$$\hat{t}_{\Pi^*(j)} = \max \left\{ j, [C_j^{\Pi^*(j)}(\Pi^*)] \right\}, \quad j = 1, \dots, s \quad (3)$$

$$v_i(\Pi^*) = \frac{ih + B_j(\Pi^*)}{M_j(\Pi^*)} \quad \text{for } \hat{t}_{\Pi^*(j)} \leq i < \hat{t}_{\Pi^*(j+1)}, \quad j = 1, \dots, s, \quad (4)$$

where $\hat{t}_{s+1} = \infty$.

Proof: Our proof is by induction on j . First we show it for $j = 1$. For the first job waiting for service (the highest priority job) the cost of using server j is $\frac{h+\beta_j}{\mu_j}$, hence Π^* is the optimal ordering when $j = 1$. Note that it is not optimal for the first job to idle, so, as in the proof of Theorem 1, (3) and (4) are both true for $j = 1$.

Next suppose (3) and (4) are true for $j-1$ and the first $j-1$ preferred servers are determined according to Π^* . Suppose jobs $\hat{t}_{\Pi^*(j-1)} + 1, \dots, i-1$ wait for the first $j-1$ servers instead of using one of the other servers. Now consider the individually optimal decision for the i th job. It will use the k th server rather than waiting if and only if

$$\begin{aligned} &\frac{h}{M_{j-1}(\Pi^*)} + v_{i-1}(\Pi^*) \\ &\geq \frac{h + \beta_k}{M_{j-1}(\Pi^*) + \mu_k} + \frac{M_{j-1}(\Pi^*)}{M_{j-1}(\Pi^*) + \mu_k} v_{i-1}(\Pi^*). \end{aligned} \quad (5)$$

The term on the left hand side is the cost incurred if the job waits and after a departure becomes the $(i-1)$ th job. The right hand side is the cost incurred if the job prefers to use the k th server (it can become the $(i-1)$ th job after a departure from one of the $j-1$ servers before it finishes service in the k th server). By the induction assumption for (4), $v_{i-1}(\Pi^*) = \frac{(i-1)h + B_{j-1}(\Pi^*)}{M_{j-1}(\Pi^*)}$. Plugging $v_{i-1}(\Pi^*)$ into (5) and some algebra yields that the i th job will use the k th server instead of waiting (i.e., $i = \hat{t}_{\Pi^*(j)}$) if and only if

$$i \geq \max \left\{ j, \frac{\beta_k M_{j-1}(\Pi^*) - \mu_k B_{j-1}(\Pi^*)}{h \mu_k} \right\}, \quad (6)$$

and if $i = \hat{t}_{\Pi^*(j)}$ the i th job's individually optimal cost will be

$$v_i(\Pi^*) = \min_{k \in \{j, \dots, s\}} \frac{ih + B_{j-1}(\Pi^*) + \beta_k}{M_{j-1}(\Pi^*) + \mu_k}. \quad (7)$$

Therefore (6) and (7) together imply that if the i th job prefers not to wait, then it will prefer server $\Pi^*(j)$ and the threshold

value satisfies (3) since i will be the threshold job, and (4) holds for $i = \hat{t}_{\Pi^*(j)}$.

If job m prefers to wait for one of the first j servers rather than using server $j + 1$, i.e., $\hat{t}_{\Pi^*(j)} < m < \hat{t}_{\Pi^*(j+1)}$, the individually optimal expected cost will be, using (4) for $i = \hat{t}_{\Pi^*(j)}$,

$$\begin{aligned} v_m(\Pi^*) &= \frac{(m - \hat{t}_{\Pi^*(j)})h}{M_j(\Pi^*)} + v_{\hat{t}_{\Pi^*(j)}}(\Pi^*) \\ &= \frac{(m - \hat{t}_{\Pi^*(j)})h}{M_j(\Pi^*)} + \frac{\hat{t}_{\Pi^*(j)}h + B_j(\Pi^*)}{M_j(\Pi^*)} \\ &= \frac{mh + B_j(\Pi^*)}{M_j(\Pi^*)}. \end{aligned}$$

Therefore (4) is also satisfied for $\hat{t}_{\Pi^*(j)} < m < \hat{t}_{\Pi^*(j+1)}$, and the proof is complete. ■

Note that if $\beta_j \leq \beta_{j+1}$ and $\mu_j \geq \mu_{j+1}$, $j = 1, \dots, n - 1$, so faster servers are also cheaper, then the optimal ordering is $\Pi^*(j) = j$, $j = 1, \dots, n$, which is consistent with the $\frac{h+\beta_j}{\mu_j}$ index. In this case, with reassignment, from (3)

$$\hat{t}_j = \max \{j, \lceil C_j \rceil\},$$

where $C_j = C_j^{\Pi^*(j)}(\Pi^*)$ for ease of notation.

If $\beta_j = 0$ for all j , then $\hat{t}_j = j$, so the fastest available servers are all used by any jobs present, in agreement with Weiss and Pinedo [9].

Now let us compare the thresholds with and without reassignment. From (1), the threshold without reassignment is $t_j = \max \left\{ 1, \left\lceil C_j + \frac{M_{j-1}}{\mu_j} - (j-1) \right\rceil \right\} = \max \left\{ 1, \left\lceil C_j + \frac{M_{j-1}}{\mu_j} \right\rceil \right\} - (j-1)$ and from Corollary 2, the threshold for the slow server problem ($\beta_j = 0$ for all $1 \leq j \leq s$) is $t_j^0 = \max \left\{ 1, \left\lceil \frac{M_{j-1}}{\mu_j} - (j-1) \right\rceil \right\} = \max \left\{ j, \left\lceil \frac{M_{j-1}}{\mu_j} \right\rceil \right\} - (j-1)$. Recall that without reassignment, we only considered those jobs in queue in our priority labeling, whereas with reassignment, *all* jobs present have priority labeling. Suppose that with reassignment the threshold for using the j th server is such that the server will not automatically be used if there are enough jobs, that is, $\hat{t}_j = \lceil C_j \rceil$. Then $t_j = \lceil M_{j-1}/\mu_j + C_j \rceil - (j-1)$, and we have the interesting fact that the threshold value without reassignment, t_j , is the sum of the threshold value with reassignment, \hat{t}_j , plus the threshold value for the slow server problem without reassignment, t_j^0 (within one, due to the ceiling function). For instance, suppose we have two servers and $(h, \beta_1, \beta_2, \mu_1, \mu_2) = (1, 6, 3, 3, 1)$. Then $t_j^0 = 3$, $\hat{t}_j = C_j = 3$ and $t_j = 6 = t_j^0 + \hat{t}_j$.

B. The Socially Optimal Policy

Recall that we are primarily interested in the socially optimal policy for the two problems, with and without reassignment. The socially optimal policy is the policy minimizing the total incurred expected cost summed over all jobs. Kumar and Walrand [28] showed that in the socially optimal policy

which can be implemented by each job implementing some individual policy, if a job never utilizes a previously declined server, then the individually and socially optimal policies are the same. The following theorem is a special case but instead of showing that their condition is satisfied, we give a direct proof.

Theorem 5: For both systems with and without reassignment, an individually optimal policy is also socially optimal.

Proof: The proofs for the systems with and without reassignment are very similar, so we give the proof assuming no reassignment. Our proof is by policy iteration, and is similar to the proof of Theorem 3 of Xu [27]. Let π^* denote the individually optimal policy. Let f be another policy which makes a different decision from the individually optimal decision at the initial decision epoch and then agrees with π^* after the initial decision. We will show that f cannot be the socially optimal policy.

Suppose that at the initial decision epoch there are r jobs waiting in queue and π^* assigns the job with k th priority to server j , while server j is idle under f . Then the k th job will be better off under π^* because it follows its individually optimal policy. Jobs $1, \dots, k-1$ will have the same individual expected cost for both policies since both policies follow the individually optimal policy after the initial decision and those jobs will not use server j . For $k+1 \leq i \leq r$, job i under f may correspond to either job $i-1$ or i under π^* after the initial decision, depending on whether the next event is a service completion on server j . Since v_i is decreasing in i these jobs will also be better off under π^* . So, π^* will be better than f for all jobs, and hence it will be better in terms of the total expected cost, i.e., f cannot be the socially optimal policy.

Next, suppose that at the initial decision epoch, f assigns the job with r th priority (the lowest priority job) to server j , while server j is idle under π^* . Note that the lowest priority job assignment is done without loss of generality under the socially optimal policy as all the jobs are identical. Jobs $1, \dots, r-1$ will have the same individual expected cost for both policies since they both follow the individually optimal policy after the initial decision and they will not use server j , and the r th job will be better off under π^* since it follows its individually optimal decision for that first decision, which is waiting. Therefore, again π^* will be better than f in terms of the total expected cost and f cannot be the socially optimal policy. ■

III. THE SYSTEM WITH ARRIVALS

We suppose that new jobs arrive according to a Poisson process with arrival rate $\lambda < \sum_{i=1}^s \mu_i$. Now the job priorities are determined by the last-come-first-priority-discipline, with preemption (LCFP-P), which gives higher priority to jobs that arrived later, and these jobs can preempt lower priority jobs using the servers. This is the same as the discipline described in Xu [27]. Now for both the cases with and without reassignment, all jobs, even those being served, have priority labels.

We first show that with reassignment the individually and socially optimal policies coincide and are threshold policies.

Without reassignment this result only holds for two servers. Later, we will derive the thresholds for two servers with reassignment by considering the individually optimal policy.

A. The Socially Optimal Policy

1) Problem with reassignment:

Theorem 6: Under LCFP-P with reassignment, the individually optimal policy is a threshold type and is identical to the socially optimal policy.

Proof: First we consider the individually optimal policy. Suppose a job tries to minimize its individual expected cost. At every decision epoch, because reassignment is possible, we order all the jobs according to their priorities and offer the servers to the jobs in priority order. Each job is going to have a preference order for the servers (this preference might depend on the priority of the job) and it will either choose to use the preferred server among the available ones or will wait for one of the busy servers. Let v_i be the individually optimal cost of the i th priority order job. It is clear that idling is not optimal for the highest priority job, so it will use a server, and that server will by definition be the preferred server, call it the first server. It is also clear that the individual expected cost of a job, w_i , is increasing in the priority order, i . Therefore eventually (if there are enough jobs present) there will be a job choosing to use some server besides the first server, call it the second server, and so on for the remaining servers. Hence the individually optimal policy will be a threshold-type policy. We now argue that the thresholds will all be finite. If not, there will be a set of servers that will never be used, say \mathbf{J} . For i large enough, $v_i = \sum_{k \notin \mathbf{J}} \frac{h}{\mu_k - \lambda} + v_{i-1}$, where $\frac{1}{\sum_{k \notin \mathbf{J}} \mu_k - \lambda}$ is the length of a busy cycle in an $M/M/1$ queue with service rate $\sum_{k \notin \mathbf{J}} \mu_k$. So v_i is strictly increasing in i . Therefore for i large enough the cost for using the k th server to completion, for some k in \mathbf{J} , is $\frac{h + \beta_k}{\mu_k} < v_i$. Therefore, the threshold for the k th server is finite and the k th server will be used if the queue length is sufficiently large. On the other hand, we also have that v_i is finite, because it could accept any server that becomes available, in which case its cost would be bounded below by the holding cost for waiting for i busy periods assuming all servers are used (none is offered to the considered customer) plus $\max_j \beta_j / \mu_j$.

Next we consider the socially optimal policy. The proof is by policy iteration and similar to the proof of Theorem 5 with π^* and f similarly defined. The only difference is that there are future arrivals after the initial decision epoch. These arrivals will have the same individual expected cost in π^* and f , because they have higher preemptive priority. Therefore, arguing as in the proof of Theorem 5, π^* will still be better than f for all jobs, and hence it will be better in terms of the total expected cost. ■

2) Problem without reassignment:

a) *Two servers:* The proof that the socially optimal policy is the same as the individually optimal policy for two servers, when reassignment is not permitted, is by policy iteration and is similar to the proofs of Theorems 5 and 6, so is omitted.

Theorem 7: Under LCFP-P without reassignment and for two servers, the individually optimal policy is the same as the socially optimal policy and is threshold type.

b) *More than two servers:* Theorem 7 will not be true in general for more than two servers. Consider the case where at the initial decision epoch f assigns the job with r th priority (the lowest priority job) to server j , while server j is idle under π^* . Then we cannot say all the jobs except the r th job have the same individual expected cost in both policies, because if there are jobs with lower priority than the r th job using servers $j + 1, \dots, s$, they are going to be affected by job r 's decision. For instance under π^* , the r th job can become the $(r + 1)$ th job due to an arrival and might choose to preempt one of the jobs using servers $j + 1, \dots, s$. However, under f , this will not be the case, so that the jobs using servers $j + 1, \dots, s$ can be better off under f .

We show, with an example similar to Weber's example for the slow-server problem, that for more than two servers, if a threshold policy is optimal, the threshold for using a server depends in general, on the states of less desirable servers [17]. In our example the only source of heterogeneity is in usage costs.

Example 8: Suppose we have three servers where $\mu_j \equiv \mu, j = 1, 2, 3$ and $\beta_1 \leq \beta_2 \leq \beta_3, \beta = (\beta_1, \beta_2, \beta_3)$. Without loss of generality we suppose $\lambda + 3\mu = 1$. We use the same dynamic programming approach as in Weber's example. Let $V_n(k, \alpha)$ denote the total cost accrued by the n th decision epoch starting from the state (k, α) where $k \in \mathbf{Z}$ denotes the number waiting in queue and $\alpha \in \{0, 1\}^3$ denotes the state of the servers (0 for idle and 1 for busy). Note that the decision epochs are arrivals or service completions. Let $A_j \alpha$ denote the vector obtained from α by assigning jobs to the j cheapest available servers and let $D_j \alpha$ denote the vector obtained from α by setting α_j to 0. We can write the dynamic programming equations as

$$V_{n+1}(k, \alpha) = \min_{j \leq k \wedge (3 - |\alpha|)} \bar{V}_{n+1}(k - j, A_j \alpha)$$

where $|\cdot|$ denotes the L^1 norm and \wedge denotes minimum, and

$$\begin{aligned} \bar{V}_{n+1}(k, \alpha) &= h(k + |\alpha|) + \beta \alpha^T + \lambda V_n(k + 1, \alpha) \\ &+ \mu \sum_{i=1}^3 V_n(k, D_i \alpha). \end{aligned}$$

For instance,

$$V_{n+1}(2, 100) = \min\{\bar{V}_{n+1}(2, 100), \bar{V}_{n+1}(1, 110), \bar{V}_{n+1}(0, 111)\}$$

and

$$\begin{aligned} \bar{V}_{n+1}(2, 100) &= 3h + \beta_1 + \lambda V_n(3, 100) \\ &+ \mu[V_n(2, 000) + V_n(2, 100) + V_n(2, 100)] \end{aligned}$$

$$\begin{aligned} \bar{V}_{n+1}(1, 110) &= 3h + \beta_1 + \beta_2 + \lambda V_n(2, 110) \\ &+ \mu[V_n(1, 010) + V_n(1, 100) + V_n(1, 110)] \end{aligned}$$

$$\begin{aligned} \bar{V}_{n+1}(0, 111) &= 3h + \beta_1 + \beta_2 + \beta_3 + \lambda V_n(1, 111) \\ &+ \mu[V_n(0, 011) + V_n(0, 101) + V_n(0, 110)] \end{aligned}$$

Suppose $(\lambda, \mu, h, \beta_1, \beta_2, \beta_3) = (0.25, 0.25, 3.01, 1.50, 7.50, 9.50)$. Then for $n = 24$, we get the following values, with 4 decimal places of accuracy:

$$\begin{array}{l} (k, \alpha) \quad \bar{V}_{24} \quad V_{24} \\ 1, 100 \quad 191.9873 \quad 191.9873 \\ 0, 110 \quad 191.9899 \\ 1, 101 \quad 242.0680 \quad 242.0582 \\ 0, 111 \quad 242.0582 \end{array} \quad (8)$$

Thus when the most expensive (least preferred) server is idle and there is only one job waiting, we would prefer not to assign that waiting job to the second server, whereas when the most expensive server is busy we would prefer to assign the waiting job to the second server. We found, as did Weber, that it is hard to come up with an example where the optimal threshold depends on the server states. For example the threshold for the second server does not depend on the state of the third server when the parameter values are $(\lambda, \mu, h, \beta_1, \beta_2, \beta_3) = (0.25, 0.25, 3.00, 1.50, 7.50, 9.50)$.

Because the socially optimal policy might depend on the state of the slower servers we have the following result. The proof is the same as the proof of Theorem 12 in [18].

Theorem 9: For the problem without reassignment and more than two servers, the individually optimal policy under LCFP-P is not necessarily the same as the socially optimal policy.

B. The Individually Optimal Policy with Two Servers

1) *Problem with reassignment:* Now we more completely characterize the individually optimal policy with reassignment and two servers. In Section II we showed that without arrivals, if $\frac{h+\beta_1}{\mu_1} < \frac{h+\beta_2}{\mu_2}$ then the first server is preferred to the second server for all states. However, with arrivals the preference depends on λ . For example, for $\mu_1 = 0.25$, $\mu_2 = 0.75$, $\beta_1 = 2$, $\beta_2 = 26$, $h = 8$, if $\lambda = 0$ the first server is preferred, but if $\lambda = 2$, the second server is preferred. To illustrate this example further we consider $\lambda = 2$, but there will only be two arrivals; after that arrivals stop. From now on we assume $\frac{h+\beta_1}{\mu_1} \leq \frac{h+\beta_2}{\mu_2}$. This example shows the somewhat surprising result that it can be optimal to use server 1 but not server 2 in some states, and to use server 2 but not server 1 in others.

Let v_i^n denote the individually optimal cost for the job in the i th priority order when there are n future arrivals and assuming both servers are available.

Example 10: Suppose there are only two future arrivals and the system parameters are as follows:

$$\mu_1 = 0.25, \mu_2 = 0.75, \beta_1 = 2, \beta_2 = 26, h = 8, \lambda = 2,$$

so $\frac{h+\beta_1}{\mu_1} \leq \frac{h+\beta_2}{\mu_2}$. We know from Theorem 4 that, without arrivals, v_i^0 can be calculated as

$$v_1^0 = \min\left\{\frac{h+\beta_1}{\mu_1}, \frac{h+\beta_2}{\mu_2}\right\} = \min\{40, 45.33\} = 40$$

$$v_2^0 = \min\left\{\frac{2h+\beta_1}{\mu_1}, \frac{2h+\beta_1+\beta_2}{\mu_1+\mu_2}\right\} = \min\{72, 44\} = 44$$

$$v_3^0 = \frac{3h+\beta_1+\beta_2}{\mu_1+\mu_2} = 52.$$

Thus, with no future arrivals the first server will be preferred and the second server will be used if there are at least two jobs. When we have only one future arrival,

$$\begin{aligned} v_1^1 &= \min\left\{\frac{h+\beta_1+\lambda v_2^0}{\mu_1+\lambda}, \frac{h+\beta_2+\lambda v_2^0}{\mu_2+\lambda}\right\} \\ &= \min\{43.56, 44.36\} = 43.56 \end{aligned}$$

so the first server will be preferred by the first job, and

$$\begin{aligned} v_2^1 &= \min\left\{\frac{h+\mu_1 v_1^1+\lambda v_3^0}{\mu_1+\lambda}, \frac{h+\beta_2+\mu_1 v_1^1+\lambda v_3^0}{\mu_1+\mu_2+\lambda}\right\} \\ &= \min\{54.62, 49.63\} = 49.63, \end{aligned}$$

so, the second server will be used if there are at least two jobs and one future arrival. Finally,

$$\begin{aligned} v_1^2 &= \min\left\{\frac{h+\beta_1+\lambda v_2^1}{\mu_1+\lambda}, \frac{h+\beta_2+\lambda v_2^1}{\mu_2+\lambda}\right\} \\ &= \min\{48.56, 48.46\} = 48.46. \end{aligned}$$

Thus, with a single job in the system and two future arrivals, the job will use the second server, but with a single job and no new arrivals, the job will use the first server. We have also found computationally that for a large number of arrivals, the second server is still preferred to the first server.

The above example shows that we need further conditions for the first server to be preferred to the second server when we have future arrivals. The following lemma is trivial to show, because we consider costs under the individually optimal policy.

Lemma 11: v_i^n is increasing in i, n .

The next theorem provides a sufficient condition for the first server to be preferred to the second server for all states and all arrival rates. An intuitively obvious condition is when the first server is both cheaper and faster, but in fact, we need a less restrictive condition. Note that by Theorem 6, Theorem 12 also holds for the socially optimal policy.

Theorem 12: Under the individually optimal policy and LCFP-P with reassignment and two servers, if $\frac{h+\beta_1}{\mu_1} \leq \frac{h+\beta_2}{\mu_2}$ and if $\mu_1 \geq \mu_2$, then the first server is preferred to the second server for all states.

Proof: To show that the first server will be preferred, we prove that, for all n ,

$$\begin{aligned} v_1^n &= \min\left\{\frac{h+\beta_1+\lambda v_2^{n-1}}{\mu_1+\lambda}, \frac{h+\beta_2+\lambda v_2^{n-1}}{\mu_2+\lambda}\right\} \\ &= \frac{h+\beta_1+\lambda v_2^{n-1}}{\mu_1+\lambda}. \end{aligned}$$

This is equivalent to showing

$$h\mu_2+\beta_1\mu_2+\beta_1\lambda+\mu_2\lambda v_2^{n-1} \leq h\mu_1+\beta_2\mu_1+\beta_2\lambda+\mu_1\lambda v_2^{n-1}.$$

Because $\frac{h+\beta_1}{\mu_1} \leq \frac{h+\beta_2}{\mu_2}$, it is sufficient to show that

$$(\mu_1-\mu_2)v_2^{n-1} \geq \beta_1-\beta_2. \quad (9)$$

Since v_2^{n-1} is increasing in n , and $\mu_1 \geq \mu_2$, it is enough to show the above for $n = 1$.

To calculate v_2^0 first suppose $2 \geq \frac{\beta_2\mu_1 - \beta_1\mu_2}{h\mu_2}$. Then by Theorem 4 the second job is above the threshold and will use the second server, so we have $v_2^0 = \frac{2h + \beta_1 + \beta_2}{\mu_1 + \mu_2}$. Therefore,

$$\begin{aligned} (\mu_1 - \mu_2)v_2^0 &= \frac{2(h\mu_1 - h\mu_2) + (\mu_1 - \mu_2)(\beta_1 + \beta_2)}{\mu_1 + \mu_2} \\ &\geq \frac{2(\beta_1\mu_2 - \beta_2\mu_1) + (\mu_1 - \mu_2)(\beta_1 + \beta_2)}{\mu_1 + \mu_2} \\ &= \frac{\beta_1\mu_2 - \beta_2\mu_1 + \mu_1\beta_1 - \mu_2\beta_2}{\mu_1 + \mu_2} \\ &= \beta_1 - \beta_2, \end{aligned}$$

where the second inequality follows from $\frac{h + \beta_1}{\mu_1} \leq \frac{h + \beta_2}{\mu_2}$. Hence, (9) is true.

Next, suppose

$$2 < \frac{\beta_2\mu_1 - \beta_1\mu_2}{h\mu_2}, \quad (10)$$

so the threshold is greater than 2, and $v_2^0 = \frac{2h + \beta_1}{\mu_1}$. Therefore, using (10),

$$\begin{aligned} (\mu_1 - \mu_2)v_2^0 &= \frac{2(h\mu_1 - h\mu_2) + \mu_1\beta_1 - \mu_2\beta_1}{\mu_1} \\ &\geq \frac{2h\mu_1 - \beta_2\mu_1 + \beta_1\mu_2 + \mu_1\beta_1 - \mu_2\beta_1}{\mu_1} \\ &= \frac{2h\mu_1 + \mu_1(\beta_1 - \beta_2)}{\mu_1} \\ &= h + (\beta_1 - \beta_2) \\ &\geq \beta_1 - \beta_2. \end{aligned}$$

Hence, (9) is true. \blacksquare

We now consider the case where $\frac{h + \beta_1}{\mu_1} \leq \frac{h + \beta_2}{\mu_2}$ but we may have $\mu_1 < \mu_2$, so, as we have seen, the server preference may depend on λ . First we need the following lemma which says costs are increasing in λ . We will show that for small arrival rates, the first server is preferred, but for large arrival rates, the second (faster) server is preferred. Recall that for stability we have $\lambda < \mu_1 + \mu_2$. Let $v_i(\lambda) = \lim_{n \rightarrow \infty} v_i^n(\lambda)$ be the individually optimal expected cost when the arrivals are not stopped (regular Poisson arrivals with rate λ). Here, we make the dependence of v_i on the arrival rate explicit.

Lemma 13: $v_i(\lambda)$ is increasing in λ .

Proof: Choose $\lambda_1 < \lambda_2 < \mu_1 + \mu_2$; we will show that $v_i(\lambda_1) \leq v_i(\lambda_2)$. We use uniformization with rate $\lambda_2 + \mu_1 + \mu_2 = \sigma$ for the decision epochs, so a ‘‘dummy transition’’ occurs with probability $\frac{\lambda_2 - \lambda_1}{\sigma}$ if the arrival rate is λ_1 , and with probability $\frac{\mu_i}{\sigma}$ if server i is idle. Without loss of generality suppose $\sigma = 1$. Suppose a job tries to minimize its individual expected cost for a finite horizon. Let $w_i^m(\lambda)$ be the individually optimal cost of the i th job for m decision epochs, and $w_i^0(\lambda) = 0$. Because a job will leave the system in a finite number of decision epochs with probability 1, we have $\lim_{m \rightarrow \infty} w_i^m(\lambda) = v_i(\lambda)$ for all $i < \infty$. Similar to Lemma 11, it is immediate that

$$w_i^m \text{ is increasing in } m, i. \quad (11)$$

We will show that $w_i^m(\lambda_1) \leq w_i^m(\lambda_2)$ by induction on m . For $m = 0$, the result trivially follows because $w_i^0(\lambda) = 0$ for

all i . Suppose $w_i^{m-1}(\lambda_1) \leq w_i^{m-1}(\lambda_2)$ for all i . We will then show $w_i^m(\lambda_1) \leq w_i^m(\lambda_2)$ by using induction on i . First we consider $i = 1$. In both systems (with arrival rates λ_1 and λ_2), the first job will be able to use either server. Suppose server A is chosen in system 2; call the other server server B . We let the highest priority job use server A in system 1 too. Then

$$\begin{aligned} w_1^m(\lambda_2) &= h + \beta_A + \lambda_2 w_2^{m-1}(\lambda_2) + \mu_B w_1^{m-1}(\lambda_2) \\ w_1^m(\lambda_1) &\leq h + \beta_A + \lambda_1 w_2^{m-1}(\lambda_1) \\ &\quad + (\lambda_2 - \lambda_1 + \mu_B) w_1^{m-1}(\lambda_1), \end{aligned}$$

and the result follows from the induction hypothesis on m and (11). If no server is chosen in system 2, then we assume the same in system 1, and we have

$$\begin{aligned} w_1^m(\lambda_2) &= h + \lambda_2 w_2^{m-1}(\lambda_2) + (\mu_A + \mu_B) w_1^{m-1}(\lambda_2) \\ w_1^m(\lambda_1) &\leq h + \lambda_1 w_2^{m-1}(\lambda_1) \\ &\quad + (\lambda_2 - \lambda_1 + \mu_A + \mu_B) w_1^{m-1}(\lambda_1), \end{aligned}$$

and again the result follows from the induction hypothesis on m and (11).

For $i > 1$, we look at the following possible cases.

- Server B is offered to job i in system 2, and job i uses it. Let job i also use it in system 1. Note that this is possible, because i will also be offered server B in system 1, from the induction hypothesis on m : $w_{i-1}^{m-1}(\lambda_1) \leq w_{i-1}^{m-1}(\lambda_2)$, and job $i-1$ will reject server B if $\beta_B/\mu_B \leq w_{i-1}^{m-1}(\lambda)$. Then

$$\begin{aligned} w_i^m(\lambda_2) &= h + \beta_B + \mu_A w_{i-1}^{m-1}(\lambda_2) + \lambda_1 w_{i+1}^{m-1}(\lambda_2) \\ &\quad + (\lambda_2 - \lambda_1) w_{i+1}^{m-1}(\lambda_2) \\ w_i^m(\lambda_1) &\leq h + \beta_B + \mu_A w_{i-1}^{m-1}(\lambda_1) + \lambda_1 w_{i+1}^{m-1}(\lambda_1) \\ &\quad + (\lambda_2 - \lambda_1) w_i^{m-1}(\lambda_1), \end{aligned}$$

and the result follows from the induction hypothesis on m and (11).

- Server B is offered to job i in system 2, and job i does not use it. Again, job i will also be offered server B in system 1. Let i not use it in system 1. Then,

$$\begin{aligned} w_i^m(\lambda_2) &= h + \mu_A w_{i-1}^{m-1}(\lambda_2) + \lambda_2 w_{i+1}^{m-1}(\lambda_2) \\ &\quad + \mu_B w_i^{m-1}(\lambda_2) \\ w_i^m(\lambda_1) &\leq h + \mu_A w_{i-1}^{m-1}(\lambda_1) + \lambda_1 w_{i+1}^{m-1}(\lambda_1) \\ &\quad + (\lambda_2 - \lambda_1 + \mu_B) w_i^{m-1}(\lambda_1), \end{aligned}$$

and the result follows from the induction hypothesis on m and (11).

- Server B is not offered to job i in system 2, so both servers are busy in system 2. If job i is not offered server B in system 1, then the result follows as in the previous cases. If job i is offered server B in system 1, we let job i use it until the next event. Then, because job i would have preferred to use server B in system 2 (because some higher priority job preferred it over waiting),

$$\begin{aligned} w_i^m(\lambda_2) &\geq h + \beta_B + \mu_A w_{i-1}^{m-1}(\lambda_2) + \lambda_2 w_{i+1}^{m+1}(\lambda_2) \\ w_i^m(\lambda_1) &\leq h + \beta_B + \mu_A w_{i-1}^{m-1}(\lambda_1) + \lambda_1 w_{i+1}^{m+1}(\lambda_1) \\ &\quad + (\lambda_2 - \lambda_1) w_i^{m-1}(\lambda_1), \end{aligned}$$

and the result follows from the induction hypothesis on m and (11).

We have shown for all m and i that $w_i^m(\lambda_1) \leq w_i^m(\lambda_2)$. Hence for all m , $w_i^m(\lambda_1) \leq \lim_{m \rightarrow \infty} w_i^m(\lambda_2) = v_i(\lambda_2) < \infty$. Taking the limit once again yields $v_i(\lambda_1) = \lim_{m \rightarrow \infty} w_i^m(\lambda_1) \leq v_i(\lambda_2)$ and the result follows. ■

Note that the proof above also gives us the following corollary

Corollary 14: Let $\mathcal{S} \subseteq (0, \infty)$ be the set of arrival rates such that the highest priority job prefers server A and let $T(\lambda)$ denote the threshold for using server B when the arrival rate is $\lambda \in \mathcal{S}$. Then $T(\lambda)$ is increasing in λ on \mathcal{S} .

Theorem 15: Under LCFP-P with reassignment and two servers, either the first server is preferred for all arrival rates or there exists a $\lambda_0 < \infty$ such that for all $\lambda < \lambda_0$, the first server is preferred, and if $\lambda > \lambda_0$, the second server is preferred.

Proof: As in the proof of Theorem 12, the first server will be preferred if and only if

$$\lambda((\mu_2 - \mu_1)v_2(\lambda) + (\beta_1 - \beta_2)) \leq h\mu_1 + \beta_2\mu_1 - h\mu_2 - \beta_1\mu_2.$$

Because $\frac{h+\beta_1}{\mu_1} \leq \frac{h+\beta_2}{\mu_2}$, the right hand side is nonnegative. Therefore, if $(\mu_2 - \mu_1)v_2(\lambda) + (\beta_1 - \beta_2) \leq 0$, the inequality above will be true for all $\lambda \geq 0$. If $(\mu_2 - \mu_1)v_2(\lambda) + (\beta_1 - \beta_2) > 0$, then we will consider $\mu_2 > \mu_1$, since for $\mu_2 \leq \mu_1$, we showed that the above is true for all $\lambda \geq 0$ in Theorem 12. So, if $\mu_2 > \mu_1$, then the left hand side is strictly increasing in λ , because $v_2(\lambda)$ is increasing in λ by Lemma 13. Therefore there exists a $\lambda_0 < \infty$ such that the right and left hand sides are equal, and for all $\lambda < \lambda_0$, the first server is preferred, and if $\lambda > \lambda_0$, the second server is preferred. ■

Let server A (B) be the preferred (not preferred) server, given arrival rate λ . That is, if $\mu_1 \geq \mu_2$ or if $\lambda < \lambda_0$, $A = 1$ and $B = 2$, else $A = 2$ and $B = 1$. Suppose the job in the i th priority order, $i \geq 2$ is offered server B. Then it will use server B only if

$$\begin{aligned} & \frac{h + \mu_A v_{i-1}(\lambda) + \mu_B v_i(\lambda) + \lambda v_{i+1}(\lambda)}{\mu_A + \mu_B + \lambda} \\ & \geq \frac{h + \beta_B + \mu_A v_{i-1}(\lambda) + \lambda v_{i+1}(\lambda)}{\mu_A + \mu_B + \lambda} \end{aligned}$$

which is equivalent to

$$v_i(\lambda) \geq \frac{\beta_B}{\mu_B}. \quad (12)$$

Since $v_i(\lambda)$ is increasing in i by Lemma 11, the above will be true for some finite $i = T$.

2) *Problem without reassignment:* We now consider the problem without reassignment, that is, a job will remain in service unless preempted by an arrival. Similar to the problem with reassignment, the preference might depend on the arrival rate. In this case a job's individually optimal expected cost will not only depend on its priority order but also will depend on whether it is in service or not and the states of the other servers. The conclusion of Example 10 (with the same numbers) will still be true, so that, the second server will be preferred to the first server in some states but the reverse will be true in others. Therefore, we need conditions in addition to $\frac{h+\beta_1}{\mu_1} \leq \frac{h+\beta_2}{\mu_2}$ for the first server to be preferred to the second

server for all states and arrival rates. The following sufficient condition is the same as the problem with reassignment and the proof is similar.

Theorem 16: For the two server case, under the individually optimal policy and LCFP-P without reassignment, if $\frac{h+\beta_1}{\mu_1} \leq \frac{h+\beta_2}{\mu_2}$ and if $\mu_1 \geq \mu_2$, then the first server is preferred to the second server for all arrival rates.

We would like to show that Theorem 15 also holds without reassignment, but we were only able to show a partial version (Theorem 21 below). To show it we need some more definitions and preliminary lemmas.

Suppose the highest priority job prefers to use server A when the arrival rate is λ . Call the other server B. Assuming uniformization with rate $\lambda + \mu_1 + \mu_2 = 1$ and a finite horizon as we did in the proof of Lemma 13, let $w_i^m(\lambda)$ denote the individually optimal cost for m decision epochs for the i th priority job in the system when there is a lower priority job using server B (note that this cost is the same as if server B were idle). Let $x_i^m(\lambda)$ be similarly defined assuming the i th job is using server B. Notice that for i large enough, say $i > T^m(\lambda)$, a job will preempt the low priority job in server B for $w_i^m(\lambda)$ and the same job will preempt the i th priority job using server B for $x_i^m(\lambda)$, and both systems will be identical, so for $i > T^m(\lambda)$, we have $w_i^m = x_i^m = y_i^m$, where the latter is the individually optimal cost for the i th job when a higher priority job is using server B. For smaller i values $w_i^m(\lambda) \leq x_i^m(\lambda)$, because the i th job could preempt the lower priority job using server B in $w_i^m(\lambda)$, and then both systems would be identical. We have the following trivial lemma (so we omit the proof).

Lemma 17: $w_i^m(\lambda)$ and $x_i^m(\lambda)$ are both increasing in m and i .

The following shows the difference is decreasing in i .

Lemma 18: $x_i^m(\lambda) - w_i^m(\lambda)$ is decreasing in i .

Proof: We use uniformization with rate $\lambda + \mu_1 + \mu_2 = 1$ for the decision epochs. The proof is by induction on m . For $m = 1$, $w_i^1(\lambda) = h$, $x_i^1(\lambda) = h + \beta_B$ for all i and the result follows. Suppose the lemma is true for $m - 1$. As noted before, we have $x_i^{m-1}(\lambda) - w_i^{m-1}(\lambda) = 0$ for all $i \geq T^{m-1}(\lambda)$. For $1 < i < T^m(\lambda)$, we have

$$\begin{aligned} x_i^m(\lambda) &= h + \beta_B + \lambda x_{i+1}^{m-1}(\lambda) + \mu_A x_{i-1}^{m-1}(\lambda) \\ w_i^m(\lambda) &= h + \lambda w_{i+1}^{m-1}(\lambda) + \mu_A w_{i-1}^{m-1}(\lambda) + \mu_B w_i^{m-1}(\lambda). \end{aligned}$$

Hence,

$$\begin{aligned} x_i^m(\lambda) - w_i^m(\lambda) &= \beta_B + \lambda[x_{i+1}^{m-1}(\lambda) - w_{i+1}^{m-1}(\lambda)] \\ &\quad + \mu_A[x_{i-1}^{m-1}(\lambda) - w_{i-1}^{m-1}(\lambda)] \\ &\quad - \mu_B w_i^{m-1}(\lambda) \\ x_{i+1}^m(\lambda) - w_{i+1}^m(\lambda) &= \beta_B + \lambda[x_{i+2}^{m-1}(\lambda) - w_{i+2}^{m-1}(\lambda)] \\ &\quad + \mu_A[x_i^{m-1}(\lambda) - w_i^{m-1}(\lambda)] \\ &\quad - \mu_B w_{i+1}^{m-1}(\lambda), \end{aligned}$$

so the result follows from induction on m and Lemma 17. Finally for $i = 1$,

$$\begin{aligned} x_1^m(\lambda) &= h + \beta_B + \lambda x_2^{m-1}(\lambda) + \mu_A x_1^{m-1}(\lambda) \\ w_1^m(\lambda) &\leq h + \lambda w_2^{m-1}(\lambda) + (\mu_A + \mu_B)w_1^{m-1}(\lambda), \end{aligned}$$

where the right hand side for the inequality is the cost incurred until the next decision epoch when the highest priority job does not use server A . Hence,

$$\begin{aligned} x_1^m(\lambda) - w_1^m(\lambda) &\geq \beta_B + \lambda[x_2^{m-1}(\lambda) - w_2^{m-1}(\lambda)] \\ &\quad + \mu_A[x_1^{m-1}(\lambda) - w_1^{m-1}(\lambda)] \\ &\quad - \mu_B w_1^{m-1}(\lambda) \\ x_2^m(\lambda) - w_2^m(\lambda) &= \beta_B + \lambda[x_3^{m-1}(\lambda) - w_3^{m-1}(\lambda)] \\ &\quad + \mu_A[x_1^{m-1}(\lambda) - w_1^{m-1}(\lambda)] \\ &\quad - \mu_B w_2^{m-1}(\lambda), \end{aligned}$$

and the result follows from induction on m and Lemma 17. ■

We develop further properties of $w_i^m(\lambda)$ and $x_i^m(\lambda)$.

Lemma 19: Let $\mathcal{S} \subseteq (0, \infty)$ be the set of arrival rates where the highest priority job prefers server A . If $\mu_A \leq \mu_B$ then

- (i) $x_i^m(\lambda) - w_i^m(\lambda)$ is decreasing in λ on \mathcal{S} ,
- (ii) $w_i^m(\lambda)$ and $x_i^m(\lambda)$ are both increasing in λ on \mathcal{S} .

Proof: Choose $\lambda_1, \lambda_2 \in \mathcal{S}$ such that $\lambda_1 < \lambda_2 < \mu_1 + \mu_2$. We again use uniformization, now with rate $\lambda_2 + \mu_1 + \mu_2 = 1$ for the decision epochs. We will show (i) and (ii) by induction on m . For $m = 1$ and for all λ , $w_i^1(\lambda) = h$, and $x_i^1(\lambda) = h + \beta_B$ for all i , hence (i) and (ii) are true. Suppose $w_i^{m-1}(\lambda_1) \leq w_i^{m-1}(\lambda_2)$, $x_i^{m-1}(\lambda_1) \leq x_i^{m-1}(\lambda_2)$ and $x_i^{m-1}(\lambda_1) - w_i^{m-1}(\lambda_1) \geq x_i^{m-1}(\lambda_2) - w_i^{m-1}(\lambda_2)$ for all i . Now we show $x_i^m(\lambda_1) - w_i^m(\lambda_1) \geq x_i^m(\lambda_2) - w_i^m(\lambda_2)$ for all i . Suppose the highest priority job in system 2 (with rate λ_2), prefers to use server A rather than idling. We let it do the same in system 1 (with rate λ_1). Then,

$$\begin{aligned} x_1^m(\lambda_1) - w_1^m(\lambda_1) &\geq \lambda_1[x_2^{m-1}(\lambda_1) - w_{i+1}^{m-1}(\lambda_1)] \\ &\quad + \mu_A[x_1^{m-1}(\lambda_1) - w_1^{m-1}(\lambda_1)] \\ &\quad + (\lambda_2 - \lambda_1)[x_1^{m-1}(\lambda_1) - w_1^{m-1}(\lambda_1)] \\ &\quad - (\mu_B - \mu_A)w_1^{m-1}(\lambda_1) + \beta_B - \beta_A \\ x_1^m(\lambda_2) - w_1^m(\lambda_2) &= \lambda_1[x_2^{m-1}(\lambda_2) - w_2^{m-1}(\lambda_2)] \\ &\quad + \mu_A[x_1^{m-1}(\lambda_2) - w_1^{m-1}(\lambda_2)] \\ &\quad + (\lambda_2 - \lambda_1)[x_2^{m-1}(\lambda_2) - w_2^{m-1}(\lambda_2)] \\ &\quad - (\mu_B - \mu_A)w_1^{m-1}(\lambda_2) + \beta_B - \beta_A, \end{aligned}$$

and the result follows from induction on m and Lemma 18. On the other hand if the highest priority job in system 2 (with rate λ_2), prefers to idle server A , we let it do the same in system 1 (with rate λ_1). Then,

$$\begin{aligned} x_1^m(\lambda_1) - w_1^m(\lambda_1) &\geq \beta_B + \lambda_1[x_2^{m-1}(\lambda_1) - w_{i+1}^{m-1}(\lambda_1)] \\ &\quad + \mu_A[x_1^{m-1}(\lambda_1) - w_1^{m-1}(\lambda_1)] \\ &\quad + (\lambda_2 - \lambda_1)[x_1^{m-1}(\lambda_1) - w_1^{m-1}(\lambda_1)] \\ &\quad - \mu_B w_1^{m-1}(\lambda_1) \\ x_1^m(\lambda_2) - w_1^m(\lambda_2) &= \beta_B + \lambda_1[x_2^{m-1}(\lambda_2) - w_2^{m-1}(\lambda_2)] \\ &\quad + \mu_A[x_1^{m-1}(\lambda_2) - w_1^{m-1}(\lambda_2)] \\ &\quad + (\lambda_2 - \lambda_1)[x_2^{m-1}(\lambda_2) - w_2^{m-1}(\lambda_2)] \\ &\quad - \mu_B w_1^{m-1}(\lambda_2), \end{aligned}$$

and the result follows from induction on m and Lemma 18. Thus we have shown $x_i^m(\lambda_1) - w_i^m(\lambda_1) \geq x_i^m(\lambda_2) - w_i^m(\lambda_2)$

for $i = 1$. Next, for $1 < i < \min\{T^m(\lambda_1), T^m(\lambda_2)\}$, we have

$$\begin{aligned} x_i^m(\lambda_1) - w_i^m(\lambda_1) &= \beta_B + \lambda_1[x_{i+1}^{m-1}(\lambda_1) - w_{i+1}^{m-1}(\lambda_1)] \\ &\quad + \mu_A[x_{i-1}^{m-1}(\lambda_1) - w_{i-1}^{m-1}(\lambda_1)] \\ &\quad + (\lambda_2 - \lambda_1)[x_i^{m-1}(\lambda_1) - w_i^{m-1}(\lambda_1)] \\ &\quad - \mu_B w_i^{m-1}(\lambda_1) \\ x_i^m(\lambda_2) - w_i^m(\lambda_2) &= \beta_B + \lambda_1[x_{i+1}^{m-1}(\lambda_2) - w_{i+1}^{m-1}(\lambda_2)] \\ &\quad + \mu_A[x_{i-1}^{m-1}(\lambda_2) - w_{i-1}^{m-1}(\lambda_2)] \\ &\quad + (\lambda_2 - \lambda_1)[x_{i+1}^{m-1}(\lambda_2) - w_{i+1}^{m-1}(\lambda_2)] \\ &\quad - \mu_B w_i^{m-1}(\lambda_2), \end{aligned}$$

and the result follows from induction on m and Lemma 18. On the other hand, for $t = T^m(\lambda_1)$, we have,

$$\begin{aligned} h + \beta_B + \lambda_1 x_{t+1}^{m-1}(\lambda_1) + \mu_A x_{t-1}^{m-1}(\lambda_1) \\ \quad + (\lambda_2 - \lambda_1) x_t^{m-1}(\lambda_1) \\ \leq h + \lambda_1 w_{t+1}^{m-1}(\lambda_1) + \mu_A w_{t-1}^{m-1}(\lambda_1) \\ \quad + (\lambda_2 - \lambda_1 + \mu_B) w_t^{m-1}(\lambda_1), \end{aligned}$$

which is equivalent to

$$\begin{aligned} \lambda_1[w_{t+1}^{m-1}(\lambda_1) - x_{t+1}^{m-1}(\lambda_1)] + \mu_A[w_{t-1}^{m-1}(\lambda_1) - x_{t-1}^{m-1}(\lambda_1)] \\ + (\lambda_2 - \lambda_1)[w_t^{m-1}(\lambda_1) - x_t^{m-1}(\lambda_1)] \\ + \mu_B w_t^{m-1}(\lambda_1) - \beta_B \geq 0. \end{aligned}$$

Then, using the induction hypothesis on m and Lemma 18, we get

$$\begin{aligned} h + \beta_B + \lambda_2 x_{t+1}^{m-1}(\lambda_2) + \mu_A x_{t-1}^{m-1}(\lambda_2) \\ \leq h + \lambda_2 w_{t+1}^{m-1}(\lambda_2) + \mu_A w_{t-1}^{m-1}(\lambda_2) + \mu_B w_t^{m-1}(\lambda_2). \end{aligned}$$

Therefore we have $T^m(\lambda_2) \leq t = T^m(\lambda_1)$. Then, for $T^m(\lambda_2) \leq i < T^m(\lambda_1)$, we have $x_i^m(\lambda_1) - w_i^m(\lambda_1) \geq 0 = x_i^m(\lambda_2) - w_i^m(\lambda_2)$, and for $i \geq T^m(\lambda_1)$ we have $x_i^m(\lambda_1) - w_i^m(\lambda_1) = 0 = x_i^m(\lambda_2) - w_i^m(\lambda_2)$. Thus, we have shown $x_i^m(\lambda_1) - w_i^m(\lambda_1) \geq x_i^m(\lambda_2) - w_i^m(\lambda_2)$ for all i .

Next we show $w_i^m(\lambda_1) \leq w_i^m(\lambda_2)$ for all i . First we consider $i = 1$. If the job prefers not to idle in system 2, we let it do the same in system 1, and we have

$$\begin{aligned} w_1^m(\lambda_2) &= h + \beta_A + \lambda_2 w_2^{m-1}(\lambda_2) + \mu_B w_1^{m-1}(\lambda_2) \\ w_1^m(\lambda_1) &\leq h + \beta_A + \lambda_1 w_2^{m-1}(\lambda_1) \\ &\quad + (\lambda_2 - \lambda_1 + \mu_B) w_1^{m-1}(\lambda_1), \end{aligned}$$

and the result follows from the induction hypothesis on m and Lemma (17). If the job prefers not to use server A in system 2, then we let it do the same in system 1, and the result follows similarly. Next we consider $i > 1$. We look at the following possible cases.

- $i = T^m(\lambda_2) \leq T^m(\lambda_1)$: Let job i also use server B in system 1, by preempting the lower priority job. Then

$$\begin{aligned} w_i^m(\lambda_2, l) &= h + \beta_B + \mu_A x_{i-1}^{m-1}(\lambda_2) + \lambda_1 x_{i+1}^{m-1}(\lambda_2, l) \\ &\quad + (\lambda_2 - \lambda_1) x_{i+1}^{m-1}(\lambda_2, l) \\ w_i^m(\lambda_1, l) &\leq h + \beta_B + \mu_A x_{i-1}^{m-1}(\lambda_1) + \lambda_1 x_{i+1}^{m-1}(\lambda_1, l) \\ &\quad + (\lambda_2 - \lambda_1) x_i^{m-1}(\lambda_1), \end{aligned}$$

and the result follows from the induction hypothesis on m and Lemma 17.

- $i < T^m(\lambda_2) \leq T^m(\lambda_1)$: We have

$$\begin{aligned} w_i^m(\lambda_2, l) &= h + \mu_A w_{i-1}^{m-1}(\lambda_2, l) + \lambda_2 w_{i+1}^{m-1}(\lambda_2, l) \\ &\quad + \mu_B w_i^{m-1}(\lambda_2, l) \\ w_i^m(\lambda_1, l) &= h + \mu_A w_{i-1}^{m-1}(\lambda_1, l) + \lambda_1 w_{i+1}^{m-1}(\lambda_1, l) \\ &\quad + (\lambda_2 - \lambda_1 + \mu_B) w_i^{m-1}(\lambda_1, l), \end{aligned}$$

and the result follows from the induction hypothesis on m and Lemma 17.

- $T^m(\lambda_2) \leq T^m(\lambda_1) < i$: We have

$$\begin{aligned} w_i^m(\lambda_2, l) &= h + (\mu_A + \mu_B) w_{i-1}^{m-1}(\lambda_2, l) \\ &\quad + \lambda_2 w_{i+1}^{m-1}(\lambda_2, l) \\ w_i^m(\lambda_1, l) &= h + (\mu_A + \mu_B) w_{i-1}^{m-1}(\lambda_1, l) \\ &\quad + \lambda_1 w_{i+1}^{m-1}(\lambda_1, l) \\ &\quad + (\lambda_2 - \lambda_1) w_i^{m-1}(\lambda_1, l), \end{aligned}$$

and the result follows from the induction hypothesis on m and Lemma 17.

- $T^m(\lambda_2) < i \leq T^m(\lambda_1)$: We let job i use server B in system 1, by preempting the lower priority job in system 1. Then, because job i would have preferred to use server B in system 2 (because some higher priority job preferred it over waiting),

$$\begin{aligned} w_i^m(\lambda_2) &\geq h + \beta_B + \mu_A x_{i-1}^{m-1}(\lambda_2) + \lambda_2 x_{i+1}^{m+1}(\lambda_2) \\ w_i^m(\lambda_1) &\leq h + \beta_B + \mu_A x_{i-1}^{m-1}(\lambda_1) + \lambda_1 x_{i+1}^{m+1}(\lambda_1) \\ &\quad + (\lambda_2 - \lambda_1) x_i^{m-1}(\lambda_1), \end{aligned}$$

and the result follows from the induction hypothesis on m and Lemma 17.

Finally, similar arguments yield $x_i^m(\lambda_1) \leq x_i^m(\lambda_2)$ and the proof is complete. ■

Corollary 20: Let $\mathcal{S} \subseteq (0, \infty)$ be the set of arrival rates where the highest priority job prefers server A and let $T(\lambda)$ denote the threshold for using server B when the arrival rate is $\lambda \in \mathcal{S}$. If $\mu_A \leq \mu_B$, then $T(\lambda)$ is increasing in λ on \mathcal{S} .

Note that $\lim_{m \rightarrow \infty} w_i^m(\lambda) = v_i(\lambda)$ where $v_i(\lambda)$ denotes the infinite horizon individually optimal cost for the i th job in queue when both servers are idle.

We have the following theorem stating the dependency of the server preference on the arrival rate. The proof is similar to the proof of Theorem 15, using the lemmas above, so we omit it.

Theorem 21: Under LCFP-P without reassignment and two servers, either the first server is preferred for all arrival rates or there exists a $\lambda_0 < \infty$ such that for all $\lambda < \lambda_0$, the first server is preferred.

C. Computing the threshold - two servers with reassignment

In this section we give an algorithm to compute the optimal threshold for the problem with reassignment and two servers. We also calculate the optimal expected cost for the job in the i th priority order.

Lemma 22: Suppose a threshold policy with threshold T is used, so the first job will be assigned to the first server and the T th job will be assigned to the second server if there are

at least T jobs. Then the expected cost for each job under the LCFP-P individually optimal policy can be calculated via the following system of equations.

$$v_i = \begin{cases} \frac{\lambda(H_2 + P_2 v_T)}{\mu_1 + \lambda P_2} & \text{if } i = 1 \\ H_i + P_i v_T \\ \quad + (1 - P_i) \frac{\lambda(H_2 + P_2 v_T)}{\mu_1 + \lambda P_2} & \text{if } 1 < i < T \\ \frac{(h + \beta_2 + \mu_1 H_{T-1} + G)}{\frac{\mu_1^2(1 - P_{T-1})}{\mu_1 + \lambda P_2} + \mu_2} & \text{if } i = T \\ \frac{(i - T)h}{\mu_1 + \mu_2 - \lambda} + v_T & \text{if } i > T \end{cases} \quad (13)$$

where

$$P_i = \frac{\left(\frac{\mu_1}{\lambda}\right)^{i-1} - 1}{\left(\frac{\mu_1}{\lambda}\right)^{T-1} - 1},$$

$$H_i = h \left[\frac{i-1}{\mu_1 - \lambda} - \frac{T-1}{\mu_1 - \lambda} P_i \right],$$

and

$$G = \frac{\lambda h}{\mu_1 + \mu_2 + \lambda} + \frac{\mu_1 \lambda H_2 (1 - P_{T-1})}{\mu_1 + \lambda P_2}.$$

Proof: Job $i = 1$ will start using the first server, and if there is an arrival before it finishes service, it will become the second priority job. Hence,

$$v_1 = \frac{h + \beta_1 + \lambda v_2}{\mu_1 + \lambda}. \quad (14)$$

For $i < T$, the job at the i th position will be the $(i+1)$ th ($(i-1)$ th) priority job if the next transition is an arrival (departure). Therefore, this process is a gambler's ruin process, where the gambler starts with $i-1$ coins, wins each game with probability $p = \frac{\lambda}{\lambda + \mu_1}$, and eventually either loses everything (becomes the first priority job in our setting) or ends up with $T-1$ coins (becomes the T th priority job in our setting). The probability of the latter is equal to

$$P_i = \frac{\left(\frac{\mu_1}{\lambda}\right)^{i-1} - 1}{\left(\frac{\mu_1}{\lambda}\right)^{T-1} - 1}.$$

The i th job will pay the holding cost until either it starts at the first server or it becomes the T th priority job. This cost will be equal to $H_i = h E[N_{i-1}^{T-1}] \frac{1}{\mu_1 + \lambda}$ where

$$E[N_{i-1}^{T-1}] = \frac{i-1}{1-2p} - \frac{T-1}{1-2p} P_i$$

is the expected number of games played in the gambler's ruin problem starting with $i-1$ coins when the total number of coins is $T-1$. Hence,

$$H_i = h \left[\frac{i-1}{\mu_1 - \lambda} - \frac{T-1}{\mu_1 - \lambda} P_i \right].$$

By combining these, we get

$$v_i = H_i + P_i v_T + (1 - P_i) v_1. \quad (15)$$

Job $i = T$ will start using the second server, and if there is an arrival (departure) before it finishes service, then it will become the $(T + 1)$ th ($(T - 1)$ th) priority job. Hence

$$v_T = \frac{h + \beta_2 + \mu_1 v_{T-1} + \lambda v_{T+1}}{\mu_1 + \mu_2 + \lambda}. \quad (16)$$

For $i > T$, the i th job will be the T th priority job after some finite amount of time with probability 1 (given $\mu_1 + \mu_2 > \lambda$). Indeed, the time until it becomes the T th priority job will be stochastically identical to the sum of $i - T$ busy cycles in an $M/M/1$ queue with service rate $\mu_1 + \mu_2$ and arrival rate λ . So,

$$v_i = \frac{(i - T)h}{\mu_1 + \mu_2 - \lambda} + v_T. \quad (17)$$

Solving (14), (15), (16) and (17) together gives (13). ■

We conclude this section by giving an iterative algorithm to calculate the optimal threshold value.

Algorithm 23: Computing the optimal threshold

$T \leftarrow 2$

while not terminated **do**

 Calculate v_T using Lemma 13

 By (12),

if $v_T \geq \frac{\beta_B}{\mu_B}$ **then**

T is the optimal threshold value, so terminate,

else

$T \leftarrow T + 1$

end if

end while

Note that the above algorithm eventually terminates by finiteness of the threshold and the optimal threshold can also be calculated more efficiently by a binary search between 2 and $T(0)$ due to Corollary 14.

IV. REMARKS

Remark 24: If the server usage costs are linearly dependent on the service rates, i.e., $\beta_i = C + \mu_i$ for some constant C , then the $\frac{h+\beta_i}{\mu_i}$ index will be strictly decreasing in μ_i . In this case faster servers will be preferred for all of our models, and for all arrival rates. For the clearing systems, for the models without and with reassignment respectively,

$$t_j = \max \left\{ 1, \left\lceil \frac{C+h}{h} \left(\frac{M_{j-1}}{\mu_j} - (j-1) \right) \right\rceil \right\},$$

$$\hat{t}_j = \max \left\{ j, \left\lceil \frac{C}{h} \left(\frac{M_{j-1}}{\mu_j} - (j-1) \right) \right\rceil \right\}.$$

Specifically for $C = 0$, we have $t_j = t_j^0$, so the thresholds for the model without reassignment is the same as the thresholds for the slow server problem ($\beta_j = 0$ for all $1 \leq j \leq s$), and we also have $\hat{t}_j = j$, hence all available servers are used for the model with reassignment.

Remark 25: If the server usage costs are quadratic in the service rates, i.e., $\beta_i = C + \mu_i^2$ for some constant C , then the $\frac{h+\beta}{\mu}$ index will have its global minimum at $\mu = \sqrt{h+C}$. Therefore if $\mu_j \leq \sqrt{h+C}$ for all j , then the $\frac{h+\beta_i}{\mu_i}$ index will be strictly decreasing in μ_i . Let $M_j^{(2)} = \sum_{i=1}^j \mu_i^2$. Then, for clearing systems, we have

$$t_j = \max \left\{ 1, \left\lceil \frac{C+h}{h} \left(\frac{M_{j-1}}{\mu_j} - (j-1) \right) - \frac{1}{h} \left(M_{j-1}^{(2)} - \mu_j M_{j-1} \right) \right\rceil \right\}$$

and if the optimal preference ordering is the μ index, then

$$\hat{t}_j = \max \left\{ j, \left\lceil \frac{C}{h} \left(\frac{M_{j-1}}{\mu_j} - (j-1) \right) - \frac{1}{h} \left(M_{j-1}^{(2)} - \mu_j M_{j-1} \right) \right\rceil \right\}.$$

Note that the thresholds are smaller when operating costs are quadratic in the speeds than when they are linear.

In practical systems, $P(\mu) = \mu^\alpha$ or $P(\mu) = C + \mu^\alpha$ where $P(\mu)$ is the power (energy consumed per unit time) required to run the servers at speed μ . Hence $\beta = k\mu^\alpha$. In many applications $\alpha \in (1, 3)$, for instance for CMOS chips, $\alpha \approx 1.8$ is a good approximation [23].

V. CONCLUSION

In this paper we studied optimal scheduling in a system with holding and server usage costs. We analyzed optimal preference relations and calculated optimal threshold values for the system without arrivals. When there are arrivals, we also have shown that the two sources of heterogeneity surprisingly make the preference relation depend on the arrival rate.

We first studied the clearing system and derived the optimal preference index for multiple servers and for two different problems, one in which jobs can be reassigned from one server to another or removed from a server and put in the queue at no cost, and one in which after being assigned to a server a job cannot be removed from that server. For the same two problems, we then studied systems with arrivals. We showed the equivalence of the socially and the individually optimal (threshold) policies, except for the problem without reassignment, and more than two servers. Then using the individually optimal policies, we showed that the preference relation depends on the arrival rate for both problems. Finally we gave an algorithm to calculate the optimal threshold for the problem with reassignment.

REFERENCES

- [1] U. E. P. A. E. S. Program. (2007, Aug.) Report to congress on server and data center energy efficiency public law 109-431. [Online]. Available: http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_Report_Congress_Final1.pdf
- [2] C. Y. Yang, J. J. Chen, T. W. Kuo, and L. Thiele, "An approximation scheme for energy-efficient scheduling of real-time tasks in heterogeneous multiprocessor systems," in *Proc. Conference on Design, Automation and Test*, 2009.
- [3] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos, "Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, pp. 1458–1472, Nov. 2008.
- [4] M. Armony and A. Ward, "Fair dynamic routing in large-scale heterogeneous-server systems," *Operat. Res.*, vol. 58, pp. 624–637, 2010.
- [5] P. Naor, "On the regulation of queue size by levying tolls," *Econometrica*, vol. 37, pp. 15–24, 1969.
- [6] U. Yechiali, "Customers' optimal joining rules for the GI/M/s queue," *Management Science*, vol. 18, pp. 434–443, 1972.
- [7] S. Stidham, "Optimal control of admission to a queueing system," *IEEE Trans. Autom. Control*, vol. AC-30, pp. 705–713, Aug. 1985.

- [8] V. G. Kulkarni and N. Gautam, "Admission control of multi-class traffic with service priorities in high-speed networks," *Queueing Systems*, vol. 27, pp. 1–16, 1997.
- [9] G. Weiss and M. Pinedo, "Scheduling tasks with exponential service times on non-identical processors to minimize various cost functions," *Journal of Applied Probability*, vol. 17, pp. 187–202, 1980.
- [10] A. K. Agrawala, E. G. J. Coffman, M. R. Garey, and S. K. Tripathi, "A stochastic optimization algorithm minimizing expected flow times on uniform processors," *IEEE Trans. Comput.*, vol. C-33, pp. 351–356, Apr. 1984.
- [11] P. Lin and P. R. Kumar, "Optimal control of a queueing system with two heterogeneous servers," *IEEE Trans. Autom. Control*, vol. AC-29, pp. 696–703, Aug. 1984.
- [12] G. Koole, "A simple proof of the optimality of a threshold policy in a two-server queueing system," *Systems and Control Letters*, vol. 26, pp. 301–303, 1995.
- [13] J. Walrand, "A note on "optimal control of a queueing system with two heterogeneous servers"," *Systems and Control Letters*, vol. 4, pp. 131–134, 1984.
- [14] I. Viniotis and A. Ephremides, "Extension of optimality of the threshold policy in heterogeneous multiserver queueing systems," *IEEE Trans. Autom. Control*, vol. 33, pp. 104–109, Sep. 1988.
- [15] R. H. Stockbridge, "A martingale approach to the slow server problem," *Journal of Applied Probability*, vol. 28, pp. 480–486, 1991.
- [16] Z. Rosberg and A. M. Makowski, "Optimal routing to parallel heterogeneous servers-small arrival rates," *IEEE Trans. Autom. Control*, vol. 35, pp. 789–796, Jul. 1990.
- [17] R. R. Weber, "On a conjecture about assigning jobs to processors of different speeds," *IEEE Trans. Autom. Control*, vol. 38, pp. 166–170, Jan. 1993.
- [18] J. H. Kim, H. S. Ahn, and R. Righter, "Managing queues with heterogeneous servers," *Journal of Applied Probability*, vol. 48, pp. 435–452, 1999.
- [19] C. E. Bell, "Optimal operation of an M/M/2 queue with removable servers," *Operat. Res.*, vol. 28, pp. 1189–1204, 1980.
- [20] O. C. Ibe and J. Kleison, "Multi-server threshold queues with hysteresis," *Performance Evaluation*, vol. 21, pp. 185–213, 1995.
- [21] J. C. S. Liu and L. Golubchik, "Stochastic complement analysis of multi-server threshold queues with hysteresis," *Performance Evaluation*, vol. 35, pp. 19–48, 1999.
- [22] R. Zhang and Y. A. Phillis, "Fuzzy control of queueing systems with heterogeneous servers," *IEEE Trans. Fuzzy Syst.*, vol. 7, pp. 17–26, Feb. 1999.
- [23] L. L. H. Andrew, M. Lin, and A. Wierman, "Optimality, fairness, and robustness in speed scaling designs," in *Proc. ACM Sigmetrics*, 2010.
- [24] A. Wierman, L. L. H. Andrew, and A. Tang, "Power-aware speed scaling in processor sharing systems," in *Proc. IEEE INFOCOM*, Apr. 2009.
- [25] V. V. Rykov and D. V. Efrosinin, "On the slow server problem," *Automation and Remote Control*, vol. 70, pp. 2013–2023, 2009.
- [26] S. H. Xu and J. G. Shantikumar, "Optimal expulsion control - a dual approach to admission control of an ordered-entry system," *Operat. Res.*, vol. 41, pp. 1137–1152, 1993.
- [27] S. H. Xu, "A duality approach to admission and scheduling control of queues," *Queueing Systems*, vol. 18, pp. 273–300, 1994.
- [28] P. R. Kumar and J. Walrand, "Individually optimal routing in parallel systems," *Journal of Applied Probability*, vol. 22, pp. 989–995, 1986.

Douglas G. Down Douglas G. Down received the B.A.Sc. and M.A.Sc. degrees in electrical engineering from the University of Toronto and the PhD degree in electrical engineering from the University of Illinois at Urbana-Champaign.

He is a Professor in the Department of Computing and Software at McMaster University. His interests lie in scheduling and performance evaluation for distributed, flexible systems. He is a Senior Member of IEEE.

Rhonda Righter Rhonda Righter received the BS degree in applied math and business from Carnegie Mellon and the PhD degree in industrial engineering and operations research from University of California, Berkeley.

She is a Professor and past Chair of the Department of Industrial Engineering and Operations Research at the UC Berkeley. Before coming to Berkeley she taught at the Leavey School of Business at Santa Clara University. Her primary research and teaching interests are in the general area of stochastic modeling and optimization, especially as applied to service, manufacturing, telecommunications, and grid computing systems.

Prof. Righter is an associate editor for the Journal of Scheduling. She formerly served on the editorial boards of Management Science, Operations Research, Operations Research Letters, and Queueing Systems. She is the past (founding) Chair of the Applied Probability Society of INFORMS.

Osman T. Akgun received the B.S. degree in industrial engineering from Bosphorus University, Turkey, in 2007, and the M.S. degree in industrial engineering and operations research from University of California, Berkeley, in 2009.

He is a Ph.D. student in the Industrial Engineering and Operations Department at University of California, Berkeley. His research interests include stochastic modeling and optimization with applications to manufacturing, service operations and computer communications.