

# The “W” Network and the Dynamic Control of Unreliable Flexible Servers

Soroush Saghafian<sup>1</sup>, Mark P. Van Oyen<sup>1</sup>, Bora Kolfal<sup>2</sup>

<sup>1</sup> *Dept. of Indust. & Oper. Eng., Univ. of Michigan, Ann Arbor, MI 48109*

<sup>2</sup> *School of Business, University of Alberta, Edmonton, AB T6G2R6*

## Abstract

We address the problem of effectively assigning partially flexible resources to various jobs in Markovian parallel queueing systems with heterogenous and unreliable servers. We emphasize a structure forming a “W” and find that this design is highly efficient; it requires only a small amount of cross-training, but often performs almost as well as a fully cross-trained system. We show that (even allowing disruptions) a version of the  $c\mu$  rule, which prioritizes serving the “fixed task before the shared,” is optimal under some conditions. Since the optimal policy is complex in general, we develop a powerful and yet simple control policy. This policy, termed “LEWC,” (implementable in any parallel queueing system) defines a simple measure of workload costs and assigns each server to the queue with the Largest Expected Workload Cost (LEWC). Thus it effectively combines the intuition underlying two widely used policies: (1) the load balancing objective in serving the Longest Queue (LQ), and (2) the greedy cost minimization emphasis of the  $c\mu$  rule. Our extensive numerical tests show that LEWC performs well in comparison with four key policies: optimal, LQ,  $c\mu$ , and generalized  $c\mu$  ( $Gc\mu$ ). We also prove the stability of the LEWC, LQ, and  $Gc\mu$  policies.

[Supplementary materials are available for this article. Go to the publisher’s online edition of *IIE Transactions* for additional appendices (detailed proofs, additional analyses, data sets, etc.).]

**Keywords:** Flexible servers, Markov Decision Process, control of queues, unreliable servers, stochastic resource allocation.

---

## 1 Introduction

The use of cross-trained workers (or flexible machines) in manufacturing or service sectors provides flexibility by dynamically shifting workers (workloads) to respond to volatile demands, machine/worker availabilities, congestion, etc. Typically, agents/workers are *partially flexible*, in that they are trained to serve a limited number of different requests (task types) so as to achieve a cost-effective level of flexibility.

The literature on the modeling and analysis of flexibility includes the following three themes: (1) *System Design* of specific paradigms for creating flexibility to maximize an objective, (2) *Server Scheduling and Control* policies to reap the benefits of flexibility, and (3) *Performance Analysis* of specific systems and policies. Our work contributes to all three themes, especially the second theme.

System design, the first theme, motivates the development of methodology to determine which capabilities a server should be endowed with (see, for instance, Jordan and Graves (1995), Aksin and Karaesmen (2002), Hopp et al. (2004), Hopp and Van Oyen (2004), Iravani et al. (2005), Iravani

et al. (2007), Bassamboo et al. (2009), Chou et al. (2010), and Andradottir et al. (2010)). We first analyze parallel (Markovian) queueing systems with general structures (i.e., arbitrary number of queues and servers in parallel with general skill/capability sets for the servers) to prove properties such as stability. Then we focus on the “W” paradigm/structure for parallel operations that are “make-to-order.”

To motivate the “W” paradigm, consider the small customer support center illustrated in Fig. 1. In this example, both agents can handle phone calls. However, only one of them is responsible for supporting customers through the internet using chat and email. The other agent is provided the resources to handle postal mail and faxes. We refer to the queueing structure of Fig. 1 as a “W” queueing network (since it forms a “W” with respect to the server skills and workflow). For the manager of the system illustrated in Fig. 1, different request types have different response time urgencies. For instance, a quick reply to a chat or an email request is often more important than a fast response to a postal mail or to a fax. Phone calls on hold (waiting in queue) are also usually more urgent than a mail or a fax request.

Generally, in such systems, a per unit of time cost (or relative weight) of  $h'_i$  can be assigned to holding a request of type  $i$ . Additionally, the servers are usually *heterogenous*: they have different skill levels (service rates) in serving different job types. In general, one can model the service of a request of type  $i$  by server  $j$  as occurring with a rate of  $\mu'_{ji} \geq 0$  (where zero indicates that server  $j$  lacks skill  $i$ ). Moreover, servers might be subject to *stochastic disruptions* occurring with a rate of  $\theta'_j \geq 0$  for server  $j$ , which represents the time lost due to an IT disruption, unplanned absences (e.g. unexpected meetings), etc. When disrupted, server  $j$  returns to a working state after an expected  $r'_j$  units of time, which represents its average “repair” time. Assuming a type  $i$  request comes to the system at rate  $\lambda'_i$ , the manager of such a system needs to know how to assign the agents to different requests in *real-time* to obtain good performance and extract the most benefit from the partial flexibility of the servers.

Conceptually, the “W” structure can be observed in many systems in practice. One of the situations where a “W” structure may naturally arise is where tasks performed by the servers have a wide variety and can be classified as tasks that are server specific and tasks that are shared between servers. Consider, for example, a small clinic with a physician and a nurse working together. There is

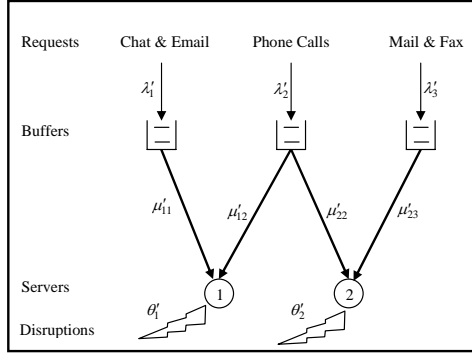
a set of tasks that would be performed by the nurse (e.g. taking blood pressure and other diagnostic tests, administering medications, and basic treatments) and there is also a separate set of tasks that would be performed by the physician (e.g. diagnosis of diseases and injuries, prescribing medications and treatments, and performing higher skill medical procedures). Additionally, there is a set of tasks that could be performed by either the nurse or the physician, depending on the workloads of the nurse and the physician (e.g., diagnostic tests, bandaging, and giving home self-care or follow-up instructions).

The “W” structure may also arise from considerations of demand workload and service capacity. For example, the shared demand type may represent the demand class for which a server cannot provide enough capacity, and thus, capacity can be shifted via cross-training. Moreover, there are often tasks that are not cost effective to cross-train. This may be caused by the training/certification expense of the skill, the difficulty in obtaining workers competent at that skill, or the infrastructure and layout that makes the cross-functionality ineffective.

We contribute to the first theme of flexibility research, system design, by showing that the “W” structure achieves most of the potential performance with two servers and three job types, supporting the notion that *a little flexibility goes a long way* (which has been a theme of several papers such as Jordan and Graves (1995) and Bassamboo et al. (2009) for some different structures). Considering the expense of cross-training servers and, more importantly, application-specific obstacles to cross-train certain task types, the frugality of the “W” design makes it widely useful in application.

We contribute to the second theme of flexibility research, control, by generating insights into effective mechanisms for the control of servers in the “W” design as well as systems with any general structure. Specifically, for the “W” structure, we rigorously establish a partial characterization of the  $c\mu$  rule (i.e., the weighted shortest processing time policy) as an optimal policy under certain operating conditions. We also develop a high-performance heuristic index policy, termed as “*Largest Expected Waiting Cost*” (LEWC), and benchmark it relative to the optimal policy for a large test suite. The proposed LEWC index policy, however, is not specific to the “W” design and can be implemented in any parallel queueing system.

Even after ignoring possible disruptions, the control problem that we consider in this paper is



**Figure 1:** An example of a small customer support center (the “W” structure).

a difficult and still an open area of research. For instance, Bell and Williams (2001) considered the “N” structure (with reliable servers), a special case of the “W” with the third demand stream removed, and noted that even for the “N” *“the problem of finding a control policy that minimizes a cost associated with holding jobs in the system is notoriously difficult.”* The “W” model is a significant departure from the “N,” because it has two partially flexible servers, whereas the “N” has two extremes: one inflexible and one fully flexible server. Server disruptions further complicate the problem. In addition to identifying sufficient conditions under which the well-known greedy  $c\mu$  policy is optimal, our numerical analysis provides further insights for situations where those conditions do not hold. Particularly, the optimal policy is a *state-dependent threshold type policy* characterized by four switching surfaces in the cases studied.

Addressing the system design agenda of the first theme, Iravani et al. (2005), Iravani et al. (2011), and Iravani et al. (2007) have developed methodologies such as Structural Flexibility and Capability Flexibility for estimating the better of alternative cross-training architectures with respect to mean waiting time. To test these methods, the above papers primarily used the Longest Queue (LQ) as the control policy. In this paper, we propose LEWC as a more effective policy. It should be noted that even for a particular structure such as “W,” performance analysis (the third theme) under the optimal policy is difficult. Thus, we provide a careful MDP-based numerical benchmarking study that gives insights into the optimal policy as well as LEWC, LQ,  $c\mu$ , and generalized  $c\mu$  ( $Gc\mu$ ) with quadratic holding cost (also referred as Max-Weight). We find that not only does the LEWC heuristic clearly outperforms LQ,  $c\mu$ , and  $Gc\mu$ , but it is also a near optimal policy with a relatively small optimality gap. Moreover, we establish its stability. Since LEWC can be used for the control of servers in

systems with any flexibility structure, the obtained results introduce LEWC as a promising policy for future research into the design of flexible structures with arbitrary topologies. This is particularly useful because the comparison of alternative flexibility/queueing designs under their optimal control policies is computationally intractable for large systems.

The rest of this paper is organized as follows. Section 2 briefly reviews some related studies. Section 3 formulates the problem using an MDP framework and identifies some attributes applicable to a parallel queueing system with a general flexibility structure. Section 4 presents the results on the “W” structure, describes the proposed LEWC heuristic, and extensively tests its performance.

## 2 Literature Survey

When there is a single server in the system that is fully flexible and has memoryless service times, Buyukkoc et al. (1985) and Walrand (1988) show that the well-known  $c\mu$  policy is optimal. The  $c\mu$  rule is a very intuitive and easy control policy to implement; however, it may perform poorly when partial flexibility is introduced, as is the case with the “W.” It remains, however, optimal under some conditions. For instance, Down and Lewis (2010) prove the optimality of the  $c\mu$  rule for an “N” structure under some special conditions. Veatch (2010) shows the optimality of  $c\mu$  for systems without disruptions where servers collaborate on jobs and special conditions are satisfied.

In parallel systems, which is our focus, the literature mainly considers the control problem in the heavy-traffic regime (see, for instance, Van Mieghem (1995), Harrison (1998), Harrison and López (1999), Bell and Williams (2001), Bramson and Williams (2000), Meyn (2003), Mandelbaum and Stolyar (2004), and Bell and Williams (2005)). The literature, however, lacks policies that are effective for a *wide range of utilizations*. Our target in this paper is on systems in the utilization range of 70% to 90%.

The problem of dynamically assigning servers to jobs has also been studied under the throughput maximization objective (see, for instance, Andradóttir et al. (2001), Andradóttir et al. (2003), Armony and Bambos (2003), Dai and Lin (2005), and Andradóttir et al. (2007)). Among these papers, Andradóttir et al. (2007) is most related to our work since it also allows for disruptions. However, throughput maximization is appropriate only for systems in which delay is not a major concern, and

in most cases it is an easier problem to analyze.

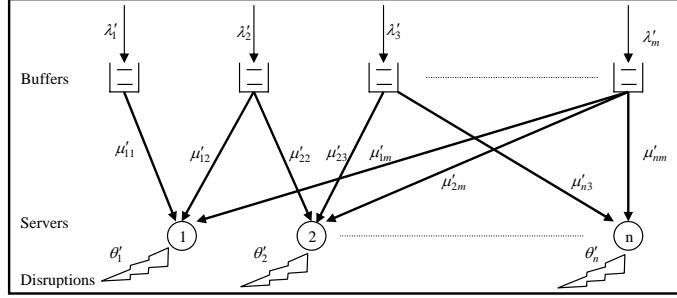
Work on the benefit of flexibility to compensate for the risk of disruptions is also related to our work. For this stream of research, we refer interested readers to Andradottir et al. (2007), Saghafian and Van Oyen (2011), and the references therein.

### 3 General Characteristics

This section addresses general Markovian parallel queueing structures with partially flexible and possibly unreliable servers. That is, we consider Markovian parallel queueing systems with an arbitrary number of servers, arbitrary number of customer classes, and arbitrary flexibility structures. We allow even more generality by allowing for a stochastic disruption/repair process unique to each server. We first describe our model, and then formulate it using an MDP framework.

#### 3.1 The Model

Consider a queueing system represented by a bipartite graph  $G = (\mathcal{N}, \mathcal{E})$  where  $\mathcal{N}$  is partitioned to two finite sets:  $\mathcal{N}_c = \{1, \dots, m\}$  for customer/jobs classes, and  $\mathcal{N}_s = \{m + 1, \dots, m + n\}$  for servers/machines (see Fig. 2). (A labeling  $\{1, \dots, n\}$  might be used for servers when it does not generate a confusion.) Arrivals of customers of class  $i \in \mathcal{N}_c$  follow a Poisson process with rate  $\lambda'_i \in \mathbb{R}^+$ , and server  $j \in \mathcal{N}_s$  can serve a customer of class  $i \in \mathcal{N}_c$  with an exponentially distributed amount of time with rate  $\mu'_{ji} \in \mathbb{R}^+$ . In the graph  $G$ ,  $(i, j) \in \mathcal{E} \subseteq \mathcal{N}_c \times \mathcal{N}_s$  if, and only if,  $\mu'_{ji} > 0$ . We let  $\mathcal{S}_j = \{i : (i, j) \in \mathcal{E}\}$  denote the skill set or “capabilities” of server  $j$  and  $\mathcal{S}_i^{-1} = \{j : (i, j) \in \mathcal{E}\}$  denote the servers capable of serving class  $i$ . To allow for server unreliability, disruptions to server  $j \in \mathcal{N}_s$  occur according to a Poisson process with rate  $\theta'_j \geq 0$  (equality holds if server  $j$  is completely reliable). Note that we focus on systems for which disruptions occur at the same rate whether or not the server is in use. For example, unplanned employee absence, a power outage, or an economic disruption may happen independently of server idleness. Once a server is disrupted, it immediately undergoes a repair process that takes an exponentially distributed amount of time with rate  $r'_j > \theta'_j$  for server  $j$ . All above-mentioned stochastic processes are considered to be independent of each other.



**Figure 2:** A general parallel queuing system with server disruptions and arbitrary flexibility structure.

Let  $\mathbf{h}' = (h'_1, \dots, h'_m)$ , where  $h'_i$  denotes the per unit time (inventory) holding cost associated with holding a customer of class  $i$ . The objective is to find an optimal resource allocation (or server assignment) policy to minimize the average holding cost of the system assuming that the servers cannot collaborate on the same job (unless otherwise mentioned), but service preemption is permitted. To achieve this goal, let  $\mathbf{X}^\pi(t) = (X_1^\pi(t), \dots, X_m^\pi(t))$  where  $X_i^\pi(t)$  denotes the number of class  $i$  customers in the system at time  $t$  under policy  $\pi$ . A policy is then optimal if it achieves the following optimal cost:

$$Z^* = \inf_{\pi \in \Pi} Z^\pi = \inf_{\pi \in \Pi} \left\{ \sum_{i \in \mathcal{N}_c} h'_i L_i^\pi \right\}, \quad (1)$$

where  $\Pi$  is the set of all admissible policies, and  $L_i^\pi$  denotes the long-run average number of class  $i$  customers in the system under policy  $\pi$ . This latter measure can be computed as:

$$L_i^\pi = \limsup_{T \rightarrow \infty} \frac{1}{T} \int_0^T E[X_i^\pi(s)] ds. \quad (2)$$

### 3.2 Formulation of the Markov Decision Process

For  $j \in \mathcal{N}_s$ , let  $a_j(t) = 1$  denote that server  $j$  is available (i.e., not disrupted) at time  $t$  and let  $a_j(t) = 0$  otherwise. The state of the system is then a vector  $\tilde{\mathbf{X}}(t) = (\mathbf{X}(t), \mathbf{a}(t))$  with state space  $\mathcal{S} = \mathbb{Z}^+ \times \{0, 1\}^n$ , where  $\tilde{\mathbf{X}}_i(t) = X_i(t) \in \mathbb{Z}^+$  for  $i \in \mathcal{N}_c$  and  $\tilde{\mathbf{X}}_j(t) = a_j(t) \in \{0, 1\}$  for  $j \in \mathcal{N}_s$ . We use uniformization (see Lippman (1975)) to formulate the discrete time equivalent of the problem. Since  $\theta'_j < r'_j$ , we use the uniformization factor  $\psi = \sum_{i \in \mathcal{N}_c} \lambda'_i + \sum_{j \in \mathcal{N}_s} r'_j + \sum_{j \in \mathcal{N}_s} \max_{i \in \mathcal{N}_c} \{\mu'_{ji}\}$  (where  $0 < \psi < \infty$ ). Let  $\lambda_i = \lambda'_i/\psi$ ,  $\mu_{ji} = \mu'_{ji}/\psi$ ,  $\theta_j = \theta'_j/\psi$ , and  $r_j = r'_j/\psi$  denote the parameters after uniformization corresponding to the transition probabilities in the underlying discrete Markov chain. Also, let  $\alpha$  be a continuous time discount rate and  $\xi$  be an exponential random variable with

rate  $\psi$  denoting the length of one unit of time in the corresponding discrete Markov chain. The equivalent discount factor in discrete time is then:

$$\beta = E[e^{-\alpha\xi}] = \int_0^\infty (e^{-\alpha t}) (\psi e^{-\psi t}) dt = \frac{\psi}{\alpha + \psi} \quad (3)$$

Also, since the state of the system does not change in one period of the discrete time version, the equivalent instantaneous one period cost is:

$$\mathbf{h}\mathbf{X}^T = E\left[\int_0^\xi \mathbf{h}'\mathbf{X}^T e^{-\alpha t} dt\right] = \frac{1-\beta}{\alpha} \mathbf{h}'\mathbf{X}^T = \frac{\mathbf{h}'}{\psi + \alpha} \mathbf{X}^T, \quad (4)$$

and so  $\mathbf{h} = \mathbf{h}'/(\psi + \alpha)$ . The finite-horizon optimal expected discounted cost can then be computed using the following optimality equation defined for every  $\tilde{\mathbf{X}} \in \mathcal{S}$  and  $n \in \mathbb{Z}^+$ :

$$\begin{aligned} V_{n+1,\beta}(\tilde{\mathbf{X}}) = & \mathbf{h}\mathbf{X}^T + \beta \left[ \sum_{i \in \mathcal{N}_c} \lambda_i V_{n,\beta}(A_i \tilde{\mathbf{X}}) + \sum_{j \in \mathcal{N}_s} [\theta_j a_j V_{n,\beta}(B_j \tilde{\mathbf{X}}) + r_j (1 - a_j) V_{n,\beta}(R_j \tilde{\mathbf{X}})] \right. \\ & + \min_{\mathbf{u} \in \mathcal{U}(\tilde{\mathbf{X}})} \left\{ \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \mu_{ji} V_{n,\beta}(D_i \tilde{\mathbf{X}}) \right. \\ & \left. \left. + \left( 1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \mu_{ji} - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] \right) V_{n,\beta}(\tilde{\mathbf{X}}) \right\} \right], \end{aligned} \quad (5)$$

where  $V_{n,\beta}(\tilde{\mathbf{X}})$  represents the optimal cost of an n-period problem starting at state  $\tilde{\mathbf{X}}$ ,  $\mathbf{1}\{\cdot\}$  is the indicator function, and the initial condition is  $V_{0,\beta}(\tilde{\mathbf{X}}) = 0$  for every  $\tilde{\mathbf{X}} \in \mathcal{S}$ . In this optimality equation, the arrival, departure, repair, and breakdown state transition operators for  $i \in \mathcal{N}_c$  and  $j \in \mathcal{N}_s$  are denoted by  $A_i \tilde{\mathbf{X}} = \tilde{\mathbf{X}} + \mathbf{e}_i$ ,  $D_i \tilde{\mathbf{X}} = \tilde{\mathbf{X}} - \mathbf{e}_i$ ,  $R_j \tilde{\mathbf{X}} = \tilde{\mathbf{X}} + \mathbf{e}_j$ , and  $B_j \tilde{\mathbf{X}} = \tilde{\mathbf{X}} - \mathbf{e}_j$ , respectively, where  $\mathbf{e}_i$  ( $\mathbf{e}_j$ ) is a vector with the same dimension as  $\mathcal{S}$  with a one in  $i$ th ( $j$ th) position and zeros elsewhere. Moreover, the control action is the vector  $\mathbf{u} = (u_j \in \mathcal{N}_c \cup \{0\}, \forall j \in \mathcal{N}_s)$  where  $u_j = i \in \mathcal{N}_c$  if server  $j$  is assigned to serve class  $i$ , and  $u_j = 0$  if it is not assigned to any class. The set of admissible control actions at state  $\tilde{\mathbf{X}}$  is denoted by a set of vectors  $\mathcal{U}(\tilde{\mathbf{X}})$ , where:

$$\mathcal{U}(\tilde{\mathbf{X}}) = \left\{ \mathbf{u} = (u_j \in \mathcal{N}_c \cup \{0\} \text{ s.t. } \forall i \in \mathcal{N}_c : \mathbf{1}\{u_j = i\} \leq a_j \mathbf{1}\{i \in \mathcal{S}_j\}, \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \leq X_i) \right\}. \quad (6)$$

That is, server  $j$  cannot be assigned to class  $i$  if it is disrupted, if it lacks skill  $i$ , or if the number of class  $i$  jobs is insufficient.

Similar to (5), the optimal average inventory holding cost can be computed using an MDP with the following average-cost optimality equation:

$$\begin{aligned}
J(\tilde{\mathbf{X}}) + Z_U^* &= \frac{\mathbf{h}'}{\psi} \mathbf{X}^T + \sum_{i \in \mathcal{N}_c} \lambda_i J(A_i \tilde{\mathbf{X}}) + \sum_{j \in \mathcal{N}_s} [\theta_j a_j J(B_j \tilde{\mathbf{X}}) + r_j (1 - a_j) J(R_j \tilde{\mathbf{X}})] \\
&+ \min_{\mathbf{u} \in \mathcal{U}(\tilde{\mathbf{X}})} \left\{ \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \mu_{ji} J(D_i \tilde{\mathbf{X}}) \right. \\
&\quad \left. + \left( 1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \mu_{ji} - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] \right) J(\tilde{\mathbf{X}}) \right\}, \tag{7}
\end{aligned}$$

where  $J(\tilde{\mathbf{X}})$  is a relative cost function,  $Z_U^*$  denotes the optimal per period average cost in the uniformized problem, and  $Z^* = \psi Z_U^*$  is the optimal per period average cost of the original problem. The next section first analyzes the stability conditions of the the general queueing system under consideration. Then it provides another method to compute the optimal average cost and the relative function  $J(\tilde{\mathbf{X}})$  using the finite-horizon version of the problem (i.e., value iteration).

### 3.3 Stability

It is important first to identify the stability region of the system for several reasons. In addition to several interesting theoretical considerations, it provides an important practical design guideline. We define the general queueing network under consideration to be stabilizable if, and only if, there exists a policy  $\pi \in \Pi$  such that  $Z^\pi = \sum_{i \in \mathcal{N}_c} h'_i L_i^\pi < \infty$ . This is equivalent to the existence of a *finite* mean equilibrium distribution of the underlying stochastic process  $\{\mathbf{X}(t), t \geq 0\}$ . To check the stability of the underlying system with partially flexible and unreliable servers, we develop and implement the following Linear Program (LP) (in the spirit of Harrison and López (1999) and Andradottir et al. (2007)). Our LP maximizes the minimum “*excess service capacity*,”  $\tau$ , that can be provided for *all* customer classes.

#### LP 1:

$$\text{Max } \tau \tag{8}$$

Subject to:

$$\sum_{j \in \mathcal{S}_i^{-1}} y_{ji} \left( \frac{r_j}{\theta_j + r_j} \right) \mu_{ji} \geq \lambda_i + \tau \quad \forall i \in \mathcal{N}_c, \quad (9)$$

$$\sum_{i \in \mathcal{S}_j} y_{ji} \leq 1 \quad \forall j \in \mathcal{N}_s, \quad (10)$$

$$y_{ji} \geq 0 \quad \forall j \in \mathcal{N}_s, \forall i \in \mathcal{S}_j. \quad (11)$$

In this LP, we introduce the decision variable  $y_{ji}$  ( $j \in \mathcal{N}_s, i \in \mathcal{S}_j$ ) to denote the long-run proportion of time that server  $j$  is “assigned” to work on class  $i$  (including the times during which server  $j$  is disrupted) when the arrival rate of class  $i$  is  $\lambda_i + \tau$ . Notice that (using either Renewal Theory or a two-state Markov chain model of disruption and repair process) server  $j$  in steady state is available  $\frac{r_j}{\theta_j + r_j}$  percent of the time. Thus,  $y_{ji} \left( \frac{r_j}{\theta_j + r_j} \right)$  represents the long-run proportion of the time that server  $j$  is available and working on class  $i$  (when the arrival rate of class  $i$  is  $\lambda_i + \tau$ ); and  $y_{ji} \left( \frac{r_j}{\theta_j + r_j} \right) \mu_{ji}$  is the corresponding long-run average capacity offered to class  $i$  by server  $j$  given  $y_{ji}$ . Hence, from constraint (9) we see that objective function (8) maximizes the minimum excess capacity among all classes. Constraint (10) (together with (11)) sets an upper bound for the total fraction of time that a server can be assigned to a specific class. The following theorem, based on fluid model analysis (see, for instance, Dai (1999)) and similar to some results presented in the literature (see for instance, Andradottir et al. (2007)), relates the above LP to the stabilizability of the system. This theorem provides a tool to ensure that the class of finite cost policies is not empty, and hence the optimization in (1) is of interest. See Online Appendix A for all of the proofs.

**Theorem 1** (Stability). *Let  $\tau^*$  be the optimal objective value of LP 1. Then:*

- (i) *The system is stabilizable (i.e.,  $\exists \pi \in \Pi$  s.t.  $Z^\pi < \infty$ ) if  $\tau^* > 0$ .*
- (ii) *The system is not stabilizable (i.e.,  $\forall \pi \in \Pi : Z^\pi = \infty$ ) if  $\tau^* < 0$ .*

Now that we have a tool to check stabilizability, we can take one step further and (1) guarantee the existence of an optimal *stationary* policy, and (2) establish the convergence of the finite-horizon problem to the average-cost case (both in the cost and in the policy). Indeed, we can establish a convenient alternative approach to find an optimal average-cost policy by stating that (1) it is sufficient to restrict attention to the class of stationary policies, and (2) solving the finite-horizon version of the problem defined in (5) can provide both the average-cost optimal value  $Z^*$  and the average-cost optimal policy  $\pi^*$ .

**Theorem 2** (Stationary Policy & Convergence). *If  $\tau^* > 0$ , then:*

(i) *There exists an average-cost optimal stationary policy.*

(ii) *The optimal average cost can be computed by:*

$$Z^* = \inf_{\pi \in \Pi} \{ \sum_{i \in \mathcal{N}_c} h'_i L_i^\pi \} = \lim_{\beta \rightarrow 1^-} \lim_{n \rightarrow \infty} \psi(1 - \beta) V_{n,\beta}(\tilde{\mathbf{X}}).$$

(iii) *The relative cost function  $J(\tilde{\mathbf{X}})$  defined in (7) satisfies:*

$$J(\tilde{\mathbf{X}}) = \lim_{\beta \rightarrow 1^-} \lim_{n \rightarrow \infty} [V_{n,\beta}(\tilde{\mathbf{X}}) - V_{n,\beta}(\mathbf{0})].$$

(iv) *Let  $\pi_{n,\beta}$  denote an optimal policy for the  $n$ -period (discounted cost) problem. Then any limit point  $\pi_\beta$  of the sequence  $\{\pi_{n,\beta}\}_{n \geq 1}$  (as  $n \rightarrow \infty$ ) is optimal for the infinite-horizon discounted cost. Moreover, any limit point of the sequence  $\{\pi_\beta\}_{\beta \in (0,1)}$  (as  $\beta \rightarrow 1^-$ ) is average-cost optimal.*

In the search for effective mechanisms to control the servers, we are able to restrict our attention to the class of policies that do not allow for unforced idling. This is shown in Appendix A (see Lemma 1 and Proposition 1), where we establish this result based on a proof of the monotonicity of the value function.

## 4 The “W” Structure

In the previous section, we presented some characteristics applicable to any Markovian parallel queueing system with an arbitrary server flexibility structure. In this section, to develop more insights, we consider a special structure forming a “W” (see Fig. 1 or Structure 4 in Fig. 3). This structure is an especially effective paradigm for systems with three demand types and two servers. It should be noted that the “N” structure, widely studied in the literature, is a special case of a “W” with  $\mu'_{23} = \lambda'_3 = 0$ .

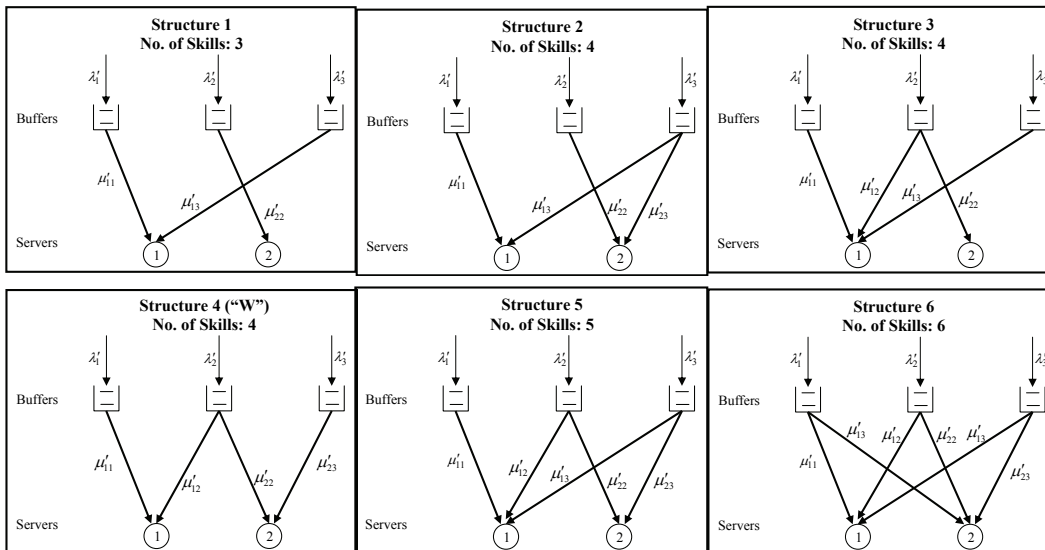
The next section shows that the “W” is an efficient design that requires only a little cross-training to achieve a performance almost as good as any design with two servers and three job types. Since cross-training the servers is costly (and sometimes infeasible) in practice, this observation shows that; for systems with three demand types and two servers, instead of fully cross-training every server, it is sufficient to make them capable to serve a shared task in addition to their dedicated/fixed one and form a “W” structure.

## 4.1 The “W” Structure: An Efficient System Design

From a system design perspective, it is crucial to understand the effective ways of cross-training servers. Please note that in this section we focus on congestion (and mean wait), so all holding costs are set to one. To understand the design problem, consider the various possible designs with three customer classes and two servers illustrated in Fig. 3. These six structures progressively add skills, except Structures 2, 3, and 4 (the “W”), which have the same number of skills. Thus, Structures 2, 3, and 4 also allow us to explore the sensitivity with respect to *where* the fourth skill is added. In Structures 3 and 4 the class with the highest arrival rate is the shared one, but it is not the case in Structure 2. Structure 2 is indeed a “W,” where the shared task is not the one with the highest arrival rate (i.e., the middle class). The goal is to find an *efficient design* among these five structures. In other words, to improve the design of Structure 1, we address two questions: (1) *where* should one implement flexibility/cross-training?, and (2) *how many* additional skills are adequate to get a reasonably good performance?

To answer these questions, we compare the performance of the above-mentioned structures under their optimal policies in various test suites (parameter settings) as presented in Table 1.C (see Online Appendix C). Notice that, considering the built-in symmetry in our test suites (symmetry between classes 1 and 3 as well as the symmetry in the speed of a server in serving different classes), the six structures considered in Fig. 3 cover all the possible designs; any other (stabilizable) structure is homomorphic to one of these six structures. Fig. 4 summarizes our computational results by depicting the optimal long-run average number of customers in each of these six structure for our test suite and under various congestion factors ( $\rho$  in Table 1.C). The mean (i.e., long-run average) number of customers (or jobs) in the system under the optimal policy is computed by numerically solving the average-cost MDP optimality equation (7) with  $h'_i = 1 (\forall i \in \mathcal{N}_c)$ .

The results depicted in Fig. 4, which is a summary of optimally solving 6 (structures)  $\times$  4 (suites)  $\times$  9 (congestion factors) = 306 problem instances, confirm that (a) flexibility usually has a diminishing rate of return, (b) a little flexibility can go a long way and (c) it usually matters where to add the additional flexibility, which has been elaborated on in studies such as Jordan and Graves (1995), Hopp et al. (2004), Iravani et al. (2005), and Bassamboo et al. (2009). The primary intent



**Figure 3:** Various possible structures with  $|\mathcal{N}_c| = 3$  and  $|\mathcal{N}_s| = 2$ .

of this section is, however, to reveal the following insight about the “W.”

- **Insight.** Structure 4, the “W” (or to be precise the “W” with the proper task being shared) is an efficient design where a little cross-training can achieve most of the flexibility of a fully flexible network (i.e., Structure 6). In test suites 2, 3, and 4, the “W” is almost as good as Structure 6 and in test suite 1, the “W” is still an efficient architecture. This observation is especially important considering the expense of cross training servers in most practical situations and reveals the benefit of implementing a “W” structure.

It should be noted that similar characteristics have been shown in the literature for chaining (see for instance Hopp et al. (2004) and Jordan and Graves (1995)) and tailored pairing (Bassamboo et al. (2009)), but the “W” is not a special case of those structures. Concurrent research in Andradottir et al. (2010) takes an alternate approach to analyzing the “W” and other structures with respect to throughput.

## 4.2 Dynamic Control of Servers in the “W” Structure

The previous section examined the benefit of implementing a “W” structure; however, this benefit cannot be fully achieved without efficiently assigning servers to jobs in real-time. Hence, the remaining question is: *what control policy should be used in real-time to extract the most benefit from*

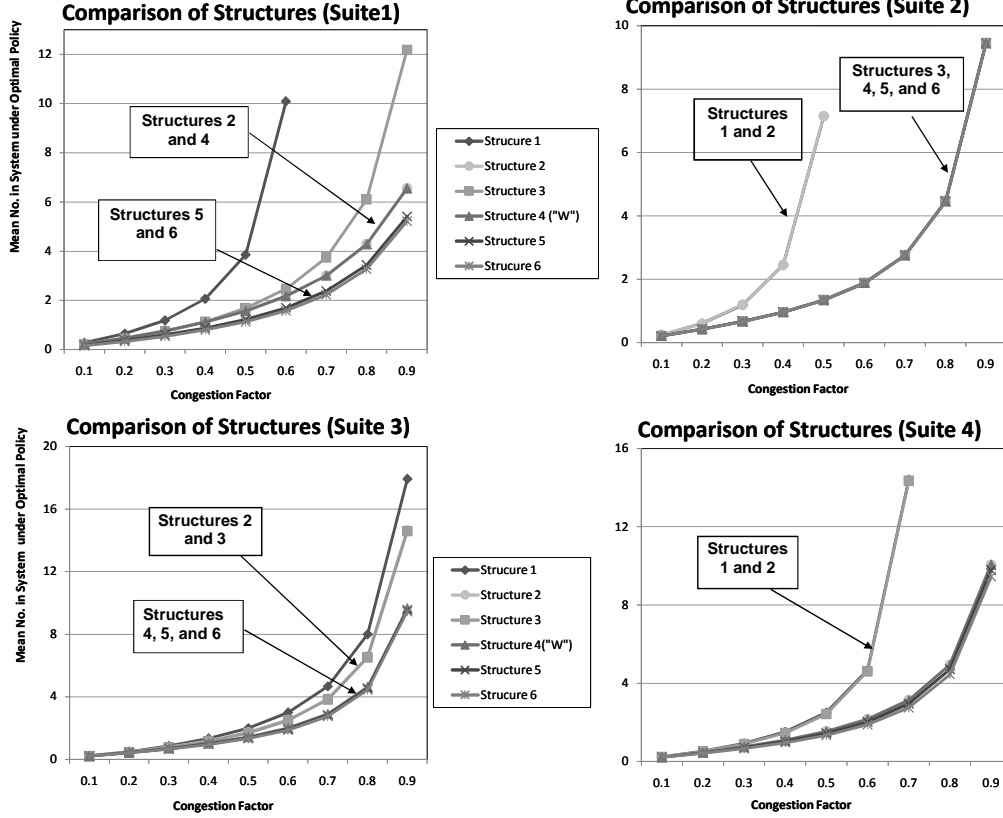


Figure 4: Comparison of possible structures under four suites of parameters using the optimal policies.

*the limited flexibility of servers in this design?* The answer to this question will also provide insight into the control of more complex queuing structures with partially flexible servers. We first state a corollary of Theorem 1 to partially characterize the stability region of a “W” design in more insightful expressions.

**Corollary 1** (Stability of “W”). *Consider a “W” structure under stochastic disruptions (or without them as a degenerate case). Let  $\rho_1 = \lambda_1 / (\mu_{11} \frac{r_1}{r_1 + \theta_1})$  and  $\rho_3 = \lambda_3 / (\mu_{23} \frac{r_2}{r_2 + \theta_2})$ . Also, define effective service rates of the shared task as  $\mu_{12}^{eff} = \mu_{12} \frac{r_1}{r_1 + \theta_1}$  and  $\mu_{22}^{eff} = \mu_{22} \frac{r_2}{r_2 + \theta_2}$ . The system is not stabilizable if  $\max\{\rho_1, \rho_3\} > 1$ . On the other hand, if  $\max\{\rho_1, \rho_3\} < 1$ , the system is stabilizable if  $(1 - \rho_1) \mu_{12}^{eff} + (1 - \rho_3) \mu_{22}^{eff} > \lambda_2$ .*

Now we characterize the optimal control policy. Hereafter, we assume the system under consideration is stabilizable. The following theorem shows the optimality of prioritizing the *fixed task before the shared* for every server under certain conditions. This policy is an analogous to the well-known  $c\mu$  ( $h\mu$  in our notation) rule as a strict priority ordering for every server.

**Theorem 3** (Optimality of the  $c\mu$  Strict Priority: Fixed before Shared). *For a “W” structure with stochastic disruptions (or without them as a degenerate case), if  $h'_1 \mu'_{11} \geq h'_2 \mu'_{12}$ ,  $h'_3 \mu'_{23} \geq h'_2 \mu'_{22}$ , and either (i) server collaboration is allowed, or (ii) server collaboration is disallowed but  $\mu'_{12} \geq \mu'_{11}$  and  $\mu'_{22} \geq \mu'_{23}$  hold, then the  $c\mu$  priority rule is optimal for each server. That is, there exists an optimal policy under which every server, when not disrupted and regardless of the other server’s allocation or disruption state, prioritizes its fixed task before the shared task whenever its fixed queue is not empty.*

The “Fixed before Shared” policy described in the Theorem 3 can be viewed as an extension of the  $c\mu$  rule for systems with partially flexible and unreliable servers. Indeed, the above theorem shows that this extension of the  $c\mu$  policy is optimal for the “W” when the  $c\mu$  index ( $h\mu$  in our terminology) gives priority to the fixed task for each server (even when servers are unreliable). Under the conditions specified in Theorem 3, using the  $c\mu$  strict priority rule for a server cannot bring the ill side effect of underutilizing the other server because of the specific flexibility structure. In other words, under these conditions, the  $c\mu$  policy is *starvation free*; it maximizes the amount of job available to the other server, and hence, remains optimal. This insights might also hold for larger systems where  $c\mu$  priorities are toward the fixed tasks for all servers. One nice feature of the above policy (i.e., Fixed before Shared) is that it defines a prescriptive rule for each server regardless of the other server’s allocation or disruption state. This feature removes the need for servers to communicate in real-time and provides a static (i.e., state-independent) rule that is easy to implement.

Our extensive MDP-based numerical computations show that the optimal policy is complex in general when  $c\mu$  priorities are not toward the fixed tasks. Relaxing all such assumptions, we observe from our extensive numerical examples that the optimal policy for a general “W” structure with server disruptions is a *state-dependent threshold type policy* which can be defined by four switching surfaces. See Online Appendix B for a detailed discussion on this observation and for numerical examples supporting it.

### 4.3 An Efficient Heuristic Policy: Largest Expected Workload Cost (LEWC)

In practice, to be implementable, a policy must be easy enough to use. In the experience of the authors, managers and researchers working with on-demand service centers usually believe that a simple policy such as LQ is preferable to the cost/effort of implementing a complex policy in real-time

(see Hopp and Van Oyen (2004) and Iravani et al. (2005)). However, our investigation has revealed that the popular LQ policy does not perform well in many situations. Moreover, as the previous section revealed, the optimal policy is complex and hard to be implemented in real-time in practice. Therefore, in this section, we develop a heuristic policy that is both easy-to-implement and highly effective.

This policy balances the expected workload cost of queues. Indeed, this heuristic prescribes that every server (whenever not disrupted) in every decision epoch should prioritize serving the queue with the *Largest Expected Workload Cost (LEWC)*, regardless of the allocation or availability (i.e., disruption state) of other servers. Under this policy, a server does not need to know all the queue lengths. Rather, each server needs visibility only of her/his duty area (skill set) to decide which queue to serve. Moreover, this policy eliminates the need for communication between servers, since each server can perform her/his job without the knowledge about the other servers' allocations, availabilities, or workloads. As a result, a manager can prescribe a rule to each server *in advance* and ensure good overall performance. In large networks more general than the “W” this is a significant advantage. However, this policy is still dynamic and requires different actions for each server depending on the real-time length of the queues within the server's skill set.

To develop this policy, we first slightly modify LP 1 presented in Section 3.3; we call the new program LP 2. The objective of this LP (applicable to any general network and not only the “W”) is to find allocations  $y_{ji}$  that maximize the minimum *percentage* excess capacity,  $\tilde{\tau}$ , among all queues.

**LP 2:**

$$\text{Max } \tilde{\tau}$$

Subject to:

$$\sum_{j \in \mathcal{S}_i^{-1}} y_{ji} \left( \frac{r_j}{\theta_j + r_j} \right) \mu_{ji} \geq \lambda_i (1 + \tilde{\tau}) \quad \forall i \in \mathcal{N}_c, \quad (12)$$

$$\sum_{i \in \mathcal{S}_j} y_{ji} \leq 1 \quad \forall j \in \mathcal{N}_s, \quad (13)$$

$$y_{ji} \geq 0 \quad \forall j \in \mathcal{N}_s, \forall i \in \mathcal{S}_j. \quad (14)$$

Next, for each queue  $i$  (with queue length  $x_i$ ), we develop an index  $J_i(x_i)$  to approximate the expected workload cost of that queue. We call this the “*LEWC index*” and define it as:

$$J_i(x_i) = \frac{h_i \times x_i}{\sum_{j \in \mathcal{S}_i^{-1}} y_{ji}^* \left( \frac{r_j}{\theta_j + r_j} \right) \mu_{ji}}, \quad (15)$$

where  $y_{ji}^*$  are the solution to LP 2, and  $\mathcal{S}_i^{-1}$  represents the set of servers able to serve queue  $i$ . In fact, if all servers that can work on queue  $i$  are assigned to work there based on the steady state allocations obtained from LP 2, a single job in the first position of queue  $i$  will take  $[\sum_{j \in \mathcal{S}_i^{-1}} y_{ji}^* (\frac{r_j}{\theta_j + r_j}) \mu_{ji}]^{-1}$  units of time to be served (assuming work sharing is permitted). Since  $x_i$  jobs are in queue  $i$ , it will take approximately (ignoring the waiting times)  $x_i \times [\sum_{j \in \mathcal{S}_i^{-1}} y_{ji}^* (\frac{r_j}{\theta_j + r_j}) \mu_{ji}]^{-1}$  units of time to serve all the jobs in queue  $i$ . This generates a workload cost of  $J_i(x_i)$  for queue  $i$ . It should be clear that the LEWC index also accounts for other system parameters, such as arrival rates, disruption rates, and repair rates through the optimal solutions  $y_{ji}^*$ . Therefore, LEWC incorporates not only the load balancing logic of LQ and the greedy cost minimization of  $c\mu$ , but also considers utilizations via solutions  $y_{ji}^*$ . The LEWC heuristic policy follows.

**LEWC Algorithm:**

- **Step 1:** Solve LP 2 to obtain the optimal allocations  $y_{ji}^*$ .
- **Step 2:** At the current state,  $\tilde{\mathbf{X}}$ , use (15) to compute indexes  $J_i(x_i)$  for all queues (i.e.,  $i \in \mathcal{N}_c$ ). Then assign each available server  $j$  to the queue  $i_j^* = \operatorname{argmax}_{i \in \mathcal{S}_j} J_i(x_i)$ , i.e., to the queue with the largest LEWC index among the queues that it can serve. If two or more queues have the same index, break the tie by assigning the server to the queue with the smallest label (i.e., the left most queue in our diagrams).

The following theorem states that our proposed policy stabilizes the system, if the system is stabilizable (i.e., if there exists a policy under which the average holding cost is finite). The ability to stabilize the system is another obvious benefit of using LEWC instead of strict priority policies, such as  $c\mu$ , which do not belong to the class of stabilizing policies (i.e., policies that always result in a finite cost if the underlying system is stabilizable).

**Theorem 4** (Stability under LEWC). *If the condition of Theorem 1 or Corollary 1 is satisfied (and hence, the system is stabilizable), then implementing the LEWC policy stabilizes the “W” system. That is, if  $Z^* = \inf_{\pi \in \Pi} Z^\pi < \infty$  and  $\Gamma$  denotes the LEWC policy, then  $Z^\Gamma < \infty$ .*

The following theorem presents the same property for the policy of serving the Longest Queue (LQ) as well as the policy of implementing the generalized  $c\mu$  ( $Gc\mu$ ) rule with quadratic holding

cost.

**Theorem 5** (Stability under LQ and  $Gc\mu$ ). *Suppose the condition of Theorem 1 or Corollary 1 is satisfied (and hence, the system is stabilizable). Then implementing either the LQ policy or the  $Gc\mu$  rule with quadratic holding costs stabilizes the “W” system. That is, if  $Z^* = \inf_{\pi \in \Pi} Z^\pi < \infty$ , and  $\nu$  denotes either of these policies policy, then  $Z^\nu < \infty$ .*

#### 4.4 Computational Results

This section compares the performance of our proposed heuristic with (1) the optimal policy, (2) the widely used policy of serving the Longest Queue (LQ), (3) the well-known  $c\mu$  rule, and (4) the generalized  $c\mu$  ( $Gc\mu$ ) rule for quadratic holding costs. Under LQ, each server prioritizes serving the queue (among its skill set) with the highest queue length. The  $c\mu$  rule, as mentioned before, prescribes server  $j$  to serve the queue  $k = \operatorname{argmax}_i c_i \mu_{ji}$ , where  $c_i$  is the holding cost of a customer in class  $i$ . Under the  $Gc\mu$ , the class to be served by server  $j$  is  $k = \operatorname{argmax}_i \mu_{ji} C'_i(x_i(t))$ , where  $x_i(t)$  is the queue length of class  $i$  at time  $t$  and  $C'_i(\cdot)$  is the derivative of the holding cost function with respect to  $x_i$ . As is prevalent in the literature, we use this policy for the case of quadratic holding cost ( $C_i(x_i) = c_i x_i^2$ ). Thus, the implemented version of the  $Gc\mu$  (also referred as Max-Weight) prescribes server  $j$  to serve class  $k = \operatorname{argmax}_i c_i \mu_{ji} x_i$ . When there is only one job in the shared queue and none in other queues, under all policies we assume the server that is (among available servers) faster in serving the shared task serves the only job in the system.

To perform the comparisons, we developed an extensive test suite of problem instances that covers various combinations of holding costs, disruption rates, service rates, arrival rates, workload distribution among the queues, and system congestions around 70% and 90% (which are common in small service centers and make-to-order manufacturing systems). Part II of the Online Appendix C presents this test suite and the methods used to cover wide ranges of parameter combinations. This test suite generates 480 problem instances for the “W” network and builds a fairly large test suite (given the computational effort for these models).

To benchmark the “W,” we employed the Markov Decision Process (MDP) of Section 3.2 to compute the optimal cost for each of our problem instances. A similar computational framework is used for *policy evaluation* to benchmark the performance of the LEWC, LQ,  $c\mu$ , and  $Gc\mu$  policies.

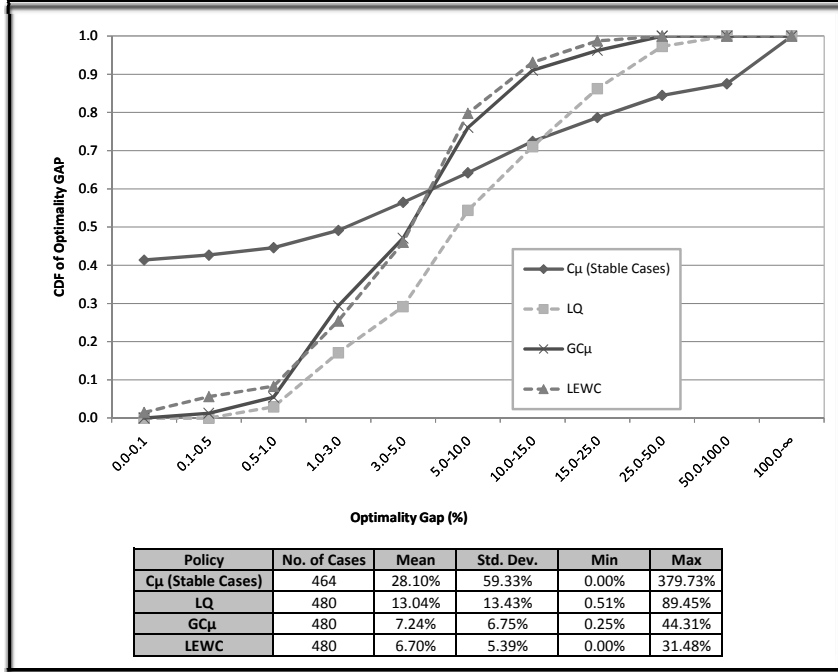


Figure 5: Performance of  $c\mu$ , LQ,  $Gc\mu$ , and LEWC relative to the optimal policy.

We used the value-iteration algorithm to solve MDPs numerically and we truncated the state space so that even for the cases with high utilization, probability of reaching the truncation limit was insignificant. Fig. 5 summarizes our computational results over the test suite by depicting the empirical Cumulative Distribution Function (CDF) for the percentage optimality gap (i.e., the CDF of percentage increase over the optimal cost) of each heuristic policy (i.e., LQ,  $c\mu$ ,  $Gc\mu$ , and LEWC). Specifically, this figure summarizes the result of our  $480 \times 5 = 2400$  MDP-based runs. Fig. 5 also presents key statistics of the obtained optimality gaps: mean, standard deviation, minimum, and maximum.

Even though the system can be stabilized (by Corollary 1) for each problem instance, we observed that the greedy  $c\mu$  policy is unstable in 16 out of 480 problem instances (i.e., 3.33% of cases) within the test suite. Hence, we considered the remaining 464 cases as the basis for computing the statistics on the  $c\mu$  rule. However, as Theorems 4 and 5 indicate, LEWC, LQ, and  $Gc\mu$  always stabilize the “W.” Fig. 5 illustrates that the proposed heuristic, LEWC, outperforms the other policies. The mean optimality gap for LEWC is 6.70% in contrast to 13.04% for LQ, 28.10% for  $c\mu$  (among stable cases), and 7.24% for  $Gc\mu$ . That is, the mean optimality gap of LQ,  $c\mu$ , and  $Gc\mu$  are 195%, 420%,

and 108% of that of LEWC, respectively. These results suggest that LEWC (as the first best) and  $Gc\mu$  (as the second best) are *nearly optimal* policies considering that the problem instances include *wide variations* on disruption rates, repair rates, arrival rates, costs, traffics, etc. This observation is especially important in light of the following:

- The optimal policy is too complex for practical application in many settings.
- Even for small systems with few servers and task types, obtaining the optimal policy becomes quickly intractable, especially when disruptions are allowed. Therefore, when the size of the systems increases, the optimality gap of a heuristic quickly becomes intractable, so comparisons to the performance of other available heuristics are appropriate.

The standard deviation column in Fig. 5 shows that LEWC is considerably more *robust* than other policies in the sense that it is more predictably effective over a wide range of model parameters. For any test case, the heuristics employ the true parameters. Thus, robustness for us is not associated with model uncertainty; rather the range of parameters over which a policy is effective. Indeed, as the figure shows, the standard deviations of LQ,  $c\mu$ , and  $Gc\mu$  are 249%, 1100%, and 125% of that of LEWC, respectively. From Fig 5 we also observe that the CDF of the optimality gap of LEWC is closer to that of  $Gc\mu$  compared to LQ and  $c\mu$ . However, LEWC outperforms all of the policies including  $Gc\mu$  in all four metrics (mean, standard deviation, min, and max). Moreover, the obtained CDF for the optimality gap of LEWC is *always* above that of LQ, illuminating the clear advantage of using LEWC over LQ. However, the CDF for the optimality gap of  $c\mu$  is *initially* above LEWC, because for about 40% of the test problems within our test suite the  $c\mu$  rule obtains the optimal cost (since its optimality conditions presented in Theorem 3 are met). Of course, one can revise the LEWC policy so that it implements the  $c\mu$  rule when its optimality conditions are met. We did not implement this obvious improvement, because the LEWC policy as stated can be applied in any general network structure for which the optimality conditions of  $c\mu$  may not be known. This way, we gain more confidence that LEWC is suitable for a wide range of applications.

Although the CDF for the optimality gap of  $c\mu$  is initially above LEWC, it should be noted that the  $c\mu$  rule is a greedy policy and is very risky to implement unless the system’s manager can ensure that its optimality conditions are not violated in advance. For instance, even under

a heavy-traffic regime, Mandelbaum and Stolyar (2004) discusses that although the  $Gc\mu$  rule is asymptotically optimal when holding costs are convex, its special case,  $c\mu$ , may not be optimal when the holding costs are linear. Our results for systems with moderate traffic and linear holding costs shows that the  $c\mu$  rule performs poorly on average. Moreover, as Fig. 5 shows,  $c\mu$  is unstable in 3.33% of cases, and among the stable cases,  $c\mu$  shows a large standard deviation of 59.33% (and a maximum of 379.73%) in its optimality gap. Our proposed algorithm, similar to  $Gc\mu$ , combines the cost minimization intuition behind the  $c\mu$  and the load balancing idea of LQ. However, unlike  $Gc\mu$ , LEWC uses an LP (LP 2) to approximate the effort levels ( $y_{ji}$ ). This use of an LP permits a more accurate estimation of the workload and allows LEWC to dynamically balance the *workload costs*. This fact makes LEWC not only a more effective policy in terms of the mean optimality gap, but also a considerably more *robust* policy with a relatively small standard deviation of 5.39%. This small standard deviation suggests another advantage of LEWC; using LEWC for comparing various queueing designs (where the optimal policy is computationally intractable) can be more reliable than implementing other policies (see, for instance, Iravani et al. (2005) and Iravani et al. (2007) where LQ is used for strategic design comparisons.)

Another observation from Fig. 5 is that the widely used LQ and  $Gc\mu$  policies are never optimal within our test suite, showing a minimum optimality gap of 0.51% and 0.25%, respectively. However, our proposed LEWC algorithm achieves the optimal cost in a few cases and, like  $c\mu$ , has a minimum gap of 0%. Moreover, LEWC, unlike  $c\mu$  and LQ, rarely results in an optimality gap of above 15%. Indeed, under LEWC the chance of obtaining a performance which is 15% worse than optimal is only 6.9% (within our test suite), but under LQ and  $c\mu$  the chances are 29.0% and 27.6%, respectively.

Another point of interest is to look at the performances of the policies in detail from the perspectives of disruption, congestion, and cost. Table 1 presents the detailed comparisons based on Settings (I)-(IV) (see Table 4.C in Online Appendix C). These four settings represent various combinations of disruption and system congestion. Setting (I) represents a system with no disruption and relatively high traffic. The scope of this research is not the heavy-traffic regime; therefore, here high traffic means relatively high congestion of around 90% and low represents 70%. In Setting (II) the servers are reliable, but the system congestion is relatively low. Settings (III) and (IV) represent scenarios where the system is under relatively lower traffic; in Setting (III), servers are completely reliable, but

they are under stochastic disruptions in Setting (IV). The results in Table 1 suggest the following observations.

- Interestingly,  $c\mu$  outperforms LQ on average under relatively low traffic (see the mean optimality gaps under Settings (II) and (IV)). Under relatively higher traffic, however, LQ is better than  $c\mu$ . This observation may suggest that the load balancing of LQ becomes more important than the greedy cost minimization of  $c\mu$  when traffic is moderate to relatively high.
- LQ is always worse than  $c\mu$  with respect to the minimum optimality gap criterion and always better with respect to the maximum optimality gap. This result is intuitive since  $c\mu$ , unlike LQ, is an extreme (and a greedy) policy. Additionally, LEWC is almost as good as  $c\mu$  (which itself outperforms LQ) under the minimum optimality gap criterion and always better than LQ (which itself outperforms  $c\mu$ ) under the maximum optimality gap criterion. Moreover, in all of these four settings, LEWC outperforms LQ,  $c\mu$ , and  $Gc\mu$  with respect to the mean optimality gap and, therefore, presents the best policy under various settings. This strength of LEWC derives from the way it accounts for different parameters of the system through the proposed LP 2 incorporated in the LEWC index.
- All of the policies show a smaller average optimality gap under lower congestion (compare Setting (I) with (II), and Setting (III) with (IV)). This observation may suggest that it is better to implement these policies for systems with low to moderate congestion rather than systems with relatively high traffic.

Table 2 compares the policies based on the various holding cost settings defined in Table 5.C in Online Appendix C. In Setting (A), all holding costs equal one, representing a symmetric situation. Settings (B)-(D) represent situations with asymmetric holding costs among customer classes where the degree of asymmetry develops from a low degree in (B) to a high degree in (D). A closer look at Table 2 provides the following observations.

- All the policies perform their best (based on the mean criterion) when there is no cost asymmetry. Moreover, the performance of both LQ and LEWC deteriorates as the level of asymmetry increases. However, this deterioration does not occur for  $c\mu$  or  $Gc\mu$ . In fact, both  $c\mu$  and  $Gc\mu$

**Table 1:** Comparison of policies based on the combinations of disruption and the system congestion using the percentage optimality gaps.

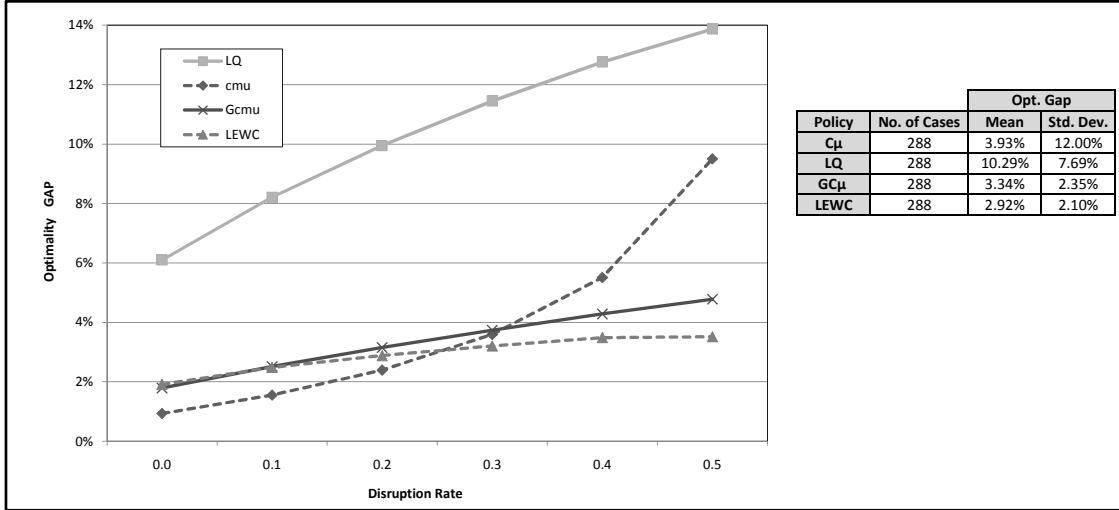
Setting	Disruption	Traffic	Policy	No. of Cases	Mean	Min	Max
(I)	No	High	$c\mu$ (Stable Cases)	112	69.52%	0.00%	379.73%
			LQ	120	22.12%	1.25%	89.45%
			$Gc\mu$	120	12.47%	0.87%	44.31%
			LEWC	120	11.92%	0.31%	31.48%
(II)	No	Low	$c\mu$ (Stable Cases)	120	7.01%	0.00%	36.75%
			LQ	120	10.45%	0.56%	43.56%
			$Gc\mu$	120	4.86%	0.25%	16.51%
			LEWC	120	4.57%	0.00%	12.06%
(III)	Yes	High	$c\mu$ (Stable Cases)	112	32.75%	0.00%	352.13%
			LQ	120	12.47%	1.23%	62.44%
			$Gc\mu$	120	7.77%	0.71%	29.12%
			LEWC	120	7.06%	0.01%	16.19%
(IV)	Yes	Low	$c\mu$ (Stable Cases)	120	6.36%	0.00%	36.95%
			LQ	120	7.11%	0.51%	34.53%
			$Gc\mu$	120	3.84%	0.33%	12.67%
			LEWC	120	3.25%	0.00%	8.03%

**Table 2:** Comparison of policies based on the holding cost settings (Level of cost asymmetry among different classes: (A) Zero, (B) Low, (C) Moderate, (D) High).

Setting	Cmu (Stable Cases)			LQ			GCmu			LEWC		
	Optimality Gap (%)			Optimality Gap (%)			Optimality Gap(%)			Optimality Gap(%)		
	Mean	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean	Min	Max
(A)	0.00	0.00	0.00	5.03	1.08	11.39	4.28	0.64	10.99	4.41	0.00	14.52
(B)	29.51	0.00	207.18	8.34	0.56	31.30	7.63	0.44	33.19	5.67	0.14	23.81
(C)	40.75	0.00	379.73	12.77	0.51	51.10	8.69	0.61	44.31	7.17	0.11	29.10
(D)	23.33	0.00	182.31	20.67	1.24	89.45	6.37	0.25	23.39	8.03	0.04	31.48
Total	28.10	0.00	379.73	13.04	0.51	89.45	7.24	0.25	44.31	6.70	0.00	31.48

perform their worst when the level of asymmetry in holding costs is moderate (Setting (C)). Although the performance of LEWC, unlike  $c\mu$  and  $Gc\mu$ , deteriorates as the level of asymmetry increases, as Table 1 showed, LEWC still outperforms both  $c\mu$  and  $Gc\mu$ .

- The proposed heuristic (LEWC) is much more robust to changes in holding costs than other policies. For instance, the mean optimality gap of LQ changes from 5.03% to 20.67% (more than 410% change) by moving from no asymmetry in costs to high asymmetry in costs whereas the mean optimality gap of LEWC only changes from 4.41% to 8.43% (less than 183% change). These results show that the performance of the widely used  $c\mu$  and LQ policies, unlike LEWC and  $Gc\mu$ , is very sensitive to the holding costs. This observation is intuitive, because LQ does not consider holding costs and  $c\mu$  depends on them in a relatively extreme way.
- To complete the previous observation, we should note that LEWC, similar to  $c\mu$  and  $Gc\mu$ , uses holding costs to determine the switching curves, but LEWC, unlike  $c\mu$  and  $Gc\mu$ , uses



**Figure 6:** Sensitivity of  $c\mu$ , LQ,  $Gc\mu$ , and LEWC to variations in disruption risks.

holding cost together with other system parameters. This fact makes LEWC less sensitive to changes in the system parameters (including holding costs and service rates). Therefore, our results recommend the use of LEWC rather than  $c\mu$  and  $Gc\mu$  when the system parameters vary over time. A similar comparison indicates that LEWC is also preferable to LQ. However, it should be noted that LQ is the rational choice in the absence of any information on the system parameters (which is perhaps its best feature).

Finally, to explore the effect of server disruptions on the performance of LEWC, LQ,  $c\mu$ , and  $Gc\mu$ , we consider the test suite presented in Tables 6.C, 7.C, and 8.C (Online Appendix C), and depict in Fig. 6 the average optimality gap (over all problem instances) of each of these policies for different levels of disruptions. As the results confirm, LEWC is the least sensitive policy to variations in the disruption risks as it explicitly incorporates disruption rates. This robustness to disruptions, provides another benefit of the proposed policy, LEWC. Among other policies,  $Gc\mu$  and LQ are more robust to disruptions compared to  $c\mu$ , since they implicitly incorporate the effect of disruptions through a consideration of queue lengths.

## 5 Conclusion

This paper considered the problem of assigning servers to various jobs in real-time to obtain good performance and thereby extract the most benefit from the flexibility of the servers. We first developed

a Markov Decision Process (MDP) modeling framework for parallel queueing systems with arbitrary number of job types, arbitrary number of servers, arbitrary flexibility structures, and heterogeneous servers subject to stochastic disruptions. We implemented a Linear Program (LP) to investigate the stability of such a queueing system, provided some convergence and monotonicity results, and showed that it is sufficient to restrict attention to the class of non-idling stationary policies.

To gain more insights into the characteristics of effective real-time server assignment mechanisms, we then considered the “W” design in which servers are trained to work on a shared task in addition to their fixed task. As a three-class network with two partially flexible servers, the “W” generalizes the “N” structure which has received considerable attention (mainly due to the intrinsic difficulties of the underlying control problem). Next, comparing the “W” design with other possible structures, we showed that the “W” queueing network is an efficient paradigm; with a little investment in flexibility, it provides performance almost as good as a fully flexible network. This paper provided specific observations into how the “W” design is the preferred structure for many systems with three demand types and two servers. Given the obstacles of fully cross-training servers in practice, our models, analyses, and numerical studies provide designers with a better understanding of *how to effectively introduce limited flexibility to a system*.

We next provided insights for a system manager on *how to benefit from the limited flexibility of servers in real-time*. We showed that, for the “W,” a version of the greedy  $c\mu$  policy that prescribes every server (whenever not disrupted) to work on *the fixed task before the shared task* is optimal under some conditions. In general, we observed that the optimal policy is of a state-dependent threshold type and can be characterized by four threshold surfaces. However, our findings confirmed that the optimal policy is complex in general, which makes it less attractive for use in practice.

Therefore, we introduced a new, powerful, and implementable policy to control the servers. Squillante et al. (2001) states that “*A fundamental understanding of the scheduling tradeoff [between achieving server load balancing and scheduling jobs where they are processed most efficiently] is of great theoretical interest.*” Our heuristic policy, LEWC, considers this trade-off and balances the *workload cost* of queues (using the traditional notion of instantaneous workload). This balance is achieved by combining the *load balancing* intuition that is effectively captured in the *queue length dependence* of the LQ policy and the greedy *cost rate minimization* concept embodied in the  $c\mu$

rule. LEWC dynamically measures the expected workload costs of the queues, and then assigns each server to the queue with Largest Expected Workload Cost (LEWC) among its skill set. We first established the stability of LEWC (as well as LQ and  $Gc\mu$ ) and then performed an extensive MDP-based numerical test to gain insights into the performance of LEWC as well as three widely used policies: the LQ policy, the  $c\mu$ , and the  $Gc\mu$  rules. Our results particularly suggested the following two conclusions: (1) LEWC is a *near optimal* policy outperforming the other policies, (2) LEWC is more *robust* than LQ,  $c\mu$ , and  $Gc\mu$  over a wide range of operating environments (holding costs, service rates, disruptions, etc.). This latter observation is an important property in practice, since system parameters often vary over time.

This robustness may also suggest that the proposed LEWC heuristic is a more reliable mechanism than the other policies for applications to strategic design, where a designer needs a fair control policy to compare the performance of alternate designs. However, future work could explore if LEWC is also robust across different cross-training designs. If so, it would also be a useful policy for the strategic design of general flexible/queueing systems. This can greatly benefit research targeting strategic design of queueing systems in the vein of Iravani et al. (2005) and Iravani et al. (2007).

**Acknowledgment.** The work of the first two authors was partially supported by NSF grant DMI-0542063.

## References

- Aksin, O.Z., F. Karaesmen. 2002. Designing flexibility: Characterizing the value of cross-training practices. Working paper, INSEAD, Fontainebleau Cedex, France.
- Andradóttir, S., H. Ayhan, D. G. Down. 2007. Compensating for failures with flexible servers. *Oper. Res.* **55**(4) 753–768.
- Andradóttir, S., H. Ayhan, D. G. Down. 2010. Design principles for flexible systems. Working Paper.
- Andradóttir, S., H. Ayhan, D.G. Down. 2001. Server assignment policies for maximizing the steady-state throughput of finite queueing systems. *Management Sci.* **47**(10) 1421–1439.
- Andradóttir, S., H. Ayhan, D.G. Down. 2003. Dynamic server allocation for queueing networks with flexible servers. *Oper. Res.* **51**(6) 952–968.
- Armony, M., N. Bambos. 2003. Queueing dynamics and maximal throughput scheduling in switched processing systems. *Queueing Systems: Theory Appl.* **44**(3) 209–252.
- Bassamboo, A., R.S. Randhawa, J.A. Van Mieghem. 2009. A little flexibility is all you need: Asymptotic optimality of tailored chaining and pairing in queueing systems. Working Paper, Kellogg School of Business, Northwestern University.
- Bell, S.L., R.J. Williams. 2001. Dynamic scheduling of a system with two parallel servers in heavy traffic with resource pooling: Asymptotic optimality of a threshold policy. *Annals of Appl. Prob.* **11**(3) 608–649.

- Bell, S.L., R.J. Williams. 2005. Dynamic scheduling of a parallel server system in heavy traffic with complete resource pooling: Asymptotic optimality of a threshold policy. *Electronic J. Prob.* **10** 1044–1115.
- Bramson, M., R.J. Williams. 2000. On dynamic scheduling of stochastic networks in heavy traffic and some new results for the workload process. *Proc. 39th IEEE Conf. Decision and Control, Sydney, Australia* 516–521.
- Buyukkoc, C., P. Varaiya, J. Walrand. 1985. The  $c\mu$  rule revisited. *Adv. Appl. Prob.* **17** 237–238.
- Chou, M.C., G.A. Chua, Zheng H. Teo, C.-P. 2010. Design for process flexibility: Efficiency of the long chain and sparse structure. *Oper. Res.* **58**(1) 43–58.
- Dai, J. G., W. Lin. 2005. Maximum Pressure Policies in Stochastic Processing Networks. *Oper. Res.* **53**(2) 197–218.
- Dai, J.G. 1999. *Stability of Fluid and Stochastic Processing Networks*. Publication 9, Centre for Mathematical Physics and Stochastics.
- Dai, J.G., S. P. Meyn. 1995. Stability and convergence of moments for multiclass queueing networks via fluid models. *IEEE Trans. Automatic Control* **40** 1889–1904.
- Down, D.G., M.E. Lewis. 2010. The N-network model with upgrades. *Probability in the Engineering and Informational Sciences* **24** 171–200.
- Harrison, J.M. 1998. Heavy traffic analysis of a system with parallel servers: Asymptotic analysis of discrete-review policies. *Annals of Appl. Prob.* **8**(3) 822–848.
- Harrison, J.M., M.J. López. 1999. Heavy traffic resource pooling in parallel-server systems. *Queueing Systems* **33** 339–368.
- Hopp, W.J., E. Tekin, M.P. Van Oyen. 2004. Benefits of skill chaining in production lines with cross-trained workers. *Management Sci.* **50**(1) 83–98.
- Hopp, W.J., M.P. Van Oyen. 2004. Agile workforce evaluation: A framework for cross-training and coordination. *IIE Transactions* **36**(10) 919–940.
- Iravani, S.M., B. Kolfal, M.P. Van Oyen. 2011. Capability flexibility: a decision support methodology for parallel service and manufacturing systems with flexible servers. *IIE Transactions* **43**(5) 363–382.
- Iravani, S.M.R., B. Kolfal, M.P. Van Oyen. 2007. Call center labor cross-training: It’s a small world after all. *Management Sci.* **53**(7) 1102–1112.
- Iravani, S.M.R., M.P. Van Oyen, K.T. Sims. 2005. Structural flexibility: A new perspective on the design of manufacturing and service operations. *Management Sci.* **51**(2) 151–166.
- Jordan, W.J., S.C. Graves. 1995. Principles on the benefits of manufacturing process flexibility. *Management Sci.* **41**(4) 577–594.
- Lippman, S. 1975. Applying a new device in the optimization of exponential queueing system. *Oper. Res.* **23**(4) 687–710.
- Mandelbaum, A., A.L. Stolyar. 2004. Scheduling flexible servers with convex delay costs: Heavy-traffic optimality of the generalaized  $c\mu$ -rule. *Oper. Res.* **52**(6) 836–855.
- Meyn, Sean P. 2003. Sequencing and routing in multiclass queueing networks part ii: Workload relaxations. *SIAM Journal on Control and Optimization* **42**(1) 178–217.
- Saghafian, S., M.P. Van Oyen. 2011. The value of flexible backup suppliers and disruption risk information: Newsvendor analyses with recourse. Working Paper, Dept. of Industrial and Operations Eng., University of Michigan.
- Sennott, L.I. 1999. *Stochastic dynamic programming and the control of queueing systems*. Wiley Series in Probability and Statistics, John Wiley and Sons, New York.
- Squillante, M. S., C. H. Xia, D. D. Yao, L. Zhang. 2001. Threshold-based priority policies for parallel-server systems with affinity scheduling. *Proc. 2001 Amer. Control Conf., Arlington, VA* 2992–2999.

- Van Mieghem, J. A. 1995. Dynamic scheduling with convex delay costs: The generalized  $c\mu$  rule. *Annals of Appl. Prob.* **5**(3) 809–833.
- Veatch, M. H. 2010. A  $c\mu$  rule for parallel servers with two tiered  $c\mu$  preference. Working Paper, Mathematics Dept., Gordon College.
- Walrand, J. 1988. *An Introduction to Queueing Networks*. Prentice-Hall, New York.

Online Supplement for “The “W” Network and the Dynamic Control of  
Unreliable Flexible Servers“ by Saghafian, Van Oyen, and Kolfal

**ONLINE APPENDIX A  
PROOFS**

**Proof of Theorem 1** (i) We use the timed round-robin policy of §5.1 of Andradottir et al. (2007) and show that if  $\tau^* > 0$ , this policy stabilizes the system (although it might not have a good performance). In other words, we show that, if  $\tau^* > 0$ , this policy results in a *finite* steady-state distribution or equivalently a *finite* performance cost. To show that, consider a special case of the queueing network in Andradottir et al. (2007) with routing matrix  $P = 0$  and setups  $s_j^\pi = 0$ , where the long-run availability of server  $j$  is given by  $\bar{a}_j = \frac{r_j}{\theta_j + r_j}$ . Let the optimal decision variables obtained from LP 1 be  $y_{ji}^*$ . Also, let  $\lambda^*$  be the optimal objective function of the following LP (used in §5.1 of Andradottir et al. (2007)):

$$\text{Max } \lambda \tag{16}$$

subject to:

$$\sum_{j \in \mathcal{N}_s} \delta_{ji} \mu_{ji} \geq \lambda \alpha_i \quad \forall i \in \mathcal{N}_c, \tag{17}$$

$$\sum_{i \in \mathcal{N}_c} \delta_{ji} \leq \bar{a}_j \quad \forall j \in \mathcal{N}_s, \tag{18}$$

$$\delta_{ji} \geq 0 \quad \forall j \in \mathcal{N}_s, \forall i \in \mathcal{N}_c, \tag{19}$$

where  $\alpha_i = \frac{\lambda_i}{\sum_{i \in \mathcal{N}_c} \lambda_i}$  is the splitting probability from the current aggregated arrival rate to the system (i.e.,  $\lambda = \sum_{i \in \mathcal{N}_c} \lambda_i$ ) to class  $i$ . Notice that because  $y_{ji}^*$  and  $\tau^* > 0$  are feasible for LP (8)-(11), decision variables  $\delta_{ji} = y_{ji}^* (\frac{r_j}{\theta_j + r_j})$  and  $\hat{\lambda} = (\sum_{i \in \mathcal{N}_c} \lambda_i) + \frac{\tau^*}{\alpha_i}$  yield a feasible solution to LP (16)-(19). Since  $\lambda^*$  is the optimum solution to LP (16)-(19), we have  $\lambda^* \geq \hat{\lambda}$ . Also, since  $\tau^* > 0$ , we have  $\hat{\lambda} > \lambda = \sum_{i \in \mathcal{N}_c} \lambda_i$ . Hence,  $\lambda^* > \lambda$ . Now consider the capacity  $\lambda$  (the current aggregated arrival rate to the system) and use the timed round-robin policy of §5.1 of Andradottir et al. (2007). Following the same line of proof as part (i) of Theorem 3 of that paper and using Theorem 2.4.9 of Dai (1999), we see that the corresponding fluid model of this policy is stable. Hence, using Theorem 4.2 of Dai (1995) and Theorem 4.1 of Dai and Meyn (1995), the underlying Markov process that represents the dynamics of the queueing system under the above-mentioned round-robin policy is positive Harris recurrent and converges to a steady-state distribution with finite moments. This proves part (i).

To prove part (ii), we show that if  $\tau^* < 0$ , the current aggregated arrival rate to the system ( $\lambda = \sum_{i \in \mathcal{N}_c} \lambda_i$ ) cannot be achieved through LP (16)-(19). In other words, we show that if  $\tau^* < 0$ ,

then  $\lambda > \lambda^*$ , i.e., the current arrival to the system is larger than  $\lambda^*$ . The rest of the proof then follows Theorem 3 (ii) of Andradottir et al. (2007). To show that  $\lambda > \lambda^*$  if  $\tau^* < 0$ , we prove the result by contradiction. Suppose  $\lambda \leq \lambda^*$  and let  $\delta_{ji}^*$  be the optimal decision variables of LP (16)-(19). Then notice that since the variables  $\delta_{ji}^*$  are feasible to this LP, the set of variables  $y_{ji} = \delta_{ji}^* (\frac{\theta_j + r_j}{r_j})$  together with  $\tau = 0$  provides a feasible solution to LP (8)-(11). However, since  $\tau^*$  is the optimum solution to this LP, we will have  $\tau^* \geq 0$  which contradicts the assumption  $\tau^* < 0$ .  $\square$

**Proof of Theorem 2.** In the proof of Theorem (1) we showed that if  $\tau^* > 0$ , then there exists a policy under which the stochastic process defining the system is positive (Harris) recurrent. Moreover, notice that for any  $c > 0$ , the set  $\{\tilde{\mathbf{X}} : \mathbf{h}\tilde{\mathbf{X}}^T \leq c\}$  is finite. Hence, the assumptions of Corollary 7.5.10 of Sennott (1999) hold, and from Theorem 7.5.6.(i) of this reference, the set of assumptions [SEN] hold. Now, first notice that the convergence of finite-horizon discounted problem to the infinite-horizon discounted problem follows from Proposition 4.3.1 of Sennott (1999). Then, since the set of assumptions [SEN] hold, parts (i), (ii), and (iii) follow from Theorem 7.2.3 of the same reference, if we also notice that  $Z^* = \psi Z_V^*$ . Part (iv) also follows from Proposition 4.3.1 of Sennott (1999) for the infinite-horizon problem and then from Theorem 7.2.3 of the same reference for the average-cost case.  $\square$

**Lemma 1** (Monotonicity). *For any  $\tilde{\mathbf{X}} \in \mathcal{S}$ ,  $n \in \mathbb{Z}^+$ ,  $\beta \in [0, 1]$ , and  $k \in \mathcal{N}_c$ :  $V_{n,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) \geq V_{n,\beta}(\tilde{\mathbf{X}})$ .*

**Proof of Lemma 1.** We prove this lemma by induction on  $n$ . For  $n = 0$ , we have  $V_{0,\beta}(\tilde{\mathbf{X}}) = 0$  ( $\forall \tilde{\mathbf{X}} \in \mathcal{S}$ ). Hence,  $V_{0,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) = V_{0,\beta}(\tilde{\mathbf{X}}) = 0$  ( $\forall \tilde{\mathbf{X}} \in \mathcal{S}$ ,  $\forall k \in \mathcal{N}_c$ ). Now assume for some  $n \in \mathbb{Z}^+$  we have  $V_{n,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) \geq V_{n,\beta}(\tilde{\mathbf{X}})$  for any  $\tilde{\mathbf{X}} \in \mathcal{S}$ ,  $\beta \in [0, 1]$ , and  $k \in \mathcal{N}_c$ . We show the same monotonicity result is also true for  $n + 1$ . From (5) we have:

$$\begin{aligned}
V_{n+1,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) = & \\
\mathbf{h}(\tilde{\mathbf{X}} + \mathbf{e}_k)^T + \beta & \left[ \sum_{i \in \mathcal{N}_c} \lambda_i V_{n,\beta}(A_i \tilde{\mathbf{X}} + \mathbf{e}_k) + \sum_{j \in \mathcal{N}_s} [\theta_j a_j V_{n,\beta}(B_j \tilde{\mathbf{X}} + \mathbf{e}_k) + r_j (1 - a_j) V_{n,\beta}(R_j \tilde{\mathbf{X}} + \mathbf{e}_k)] \right. \\
+ \min_{\mathbf{u} \in \mathcal{U}(\tilde{\mathbf{X}} + \mathbf{e}_k)} & \left\{ \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \mu_{ji} V_{n,\beta}(D_i \tilde{\mathbf{X}} + \mathbf{e}_k) \right. \\
& \left. \left. + \left( 1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \mu_{ji} - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] \right) V_{n,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) \right\} \right]
\end{aligned}$$

Now, if the minimization occurs for some control vector  $\mathbf{u}^* \in \mathcal{U}(\tilde{\mathbf{X}})$ , then after replacing  $\mathcal{U}(\tilde{\mathbf{X}} + \mathbf{e}_k)$  with  $\mathcal{U}(\tilde{\mathbf{X}})$  the proof would be straightforward since by induction assumption every term in  $V_{n+1,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k)$  is greater than or equal to the same term in  $V_{n+1,\beta}(\tilde{\mathbf{X}})$  defined in (5). However, since  $\mathcal{U}(\tilde{\mathbf{X}} + \mathbf{e}_k)$  is a larger admissible set than  $\mathcal{U}(\tilde{\mathbf{X}})$ , it is not always the case (i.e.,  $\mathbf{u}^*$  may not belong to  $\mathcal{U}(\tilde{\mathbf{X}})$ ). If  $\mathbf{u}^* \notin \mathcal{U}(\tilde{\mathbf{X}})$ , writing terms related to the control of class  $k$  separately and using the optimal action  $\mathbf{u}^*$ , we have:

$$\begin{aligned}
V_{n+1,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) = & \\
\mathbf{h}(\tilde{\mathbf{X}} + \mathbf{e}_k)^T + \beta & \left[ \sum_{i \in \mathcal{N}_c} \lambda_i V_{n,\beta}(A_i \tilde{\mathbf{X}} + \mathbf{e}_k) + \sum_{j \in \mathcal{N}_s} [\theta_j a_j V_{n,\beta}(B_j \tilde{\mathbf{X}} + \mathbf{e}_k) + r_j (1 - a_j) V_{n,\beta}(R_j \tilde{\mathbf{X}} + \mathbf{e}_k)] \right. \\
& + \left. \left\{ \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j^* = k\} \mu_{jk} [V_{n,\beta}(\tilde{\mathbf{X}}) - V_{n,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k)] + \sum_{i \neq k \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j^* = i\} \mu_{ji} V_{n,\beta}(D_i \tilde{\mathbf{X}} + \mathbf{e}_k) \right. \right. \\
& \left. \left. + \left( 1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{i \neq k \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j^* = i\} \mu_{ji} - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] \right) V_{n,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) \right\} \right] \quad (20)
\end{aligned}$$

Now we show that if one uses the same allocation policy as  $\mathbf{u}^*$  for every class  $i \neq k$  but idles one of the servers (say server  $j'$ ), which is allocated to class  $k$  in  $\mathbf{u}^*$ , and instead changes  $\tilde{\mathbf{X}} + \mathbf{e}_k$  to  $\tilde{\mathbf{X}}$ , obtains a lower value than  $V_{n+1,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k)$ . We will then show that this new value is an upper bound for  $V_{n+1,\beta}(\tilde{\mathbf{X}})$  since the new policy (say  $\mathbf{u}'$ ) is admissible (but not necessarily optimal) at state  $\tilde{\mathbf{X}}$ . Precisely, because we have uniformized the rates

$$\left( 1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{i \neq k \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j^* = i\} \mu_{ji} - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{j'k} \right) \geq 0,$$

and thus by induction assumption we have:

$$\left( 1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{i \neq k \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j^* = i\} \mu_{ji} - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{j'k} \right) [V_{n,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) - V_{n,\beta}(\tilde{\mathbf{X}})] \geq 0.$$

Now subtracting this positive term from (20), we get:

$$\begin{aligned}
& V_{n+1,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) \geq \\
& \mathbf{h}(\mathbf{X} + \mathbf{e}_k)^T + \beta \left[ \sum_{i \in \mathcal{N}_c} \lambda_i V_{n,\beta}(A_i \tilde{\mathbf{X}} + \mathbf{e}_k) + \sum_{j \in \mathcal{N}_s} [\theta_j a_j V_{n,\beta}(B_j \tilde{\mathbf{X}} + \mathbf{e}_k) + r_j (1 - a_j) V_{n,\beta}(R_j \tilde{\mathbf{X}} + \mathbf{e}_k)] \right. \\
& + \left. \left\{ \sum_{j \neq j' \in \mathcal{N}_s} \mathbb{1}\{u_j^* = k\} \mu_{jk} [V_{n,\beta}(\tilde{\mathbf{X}}) - V_{n,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k)] + \sum_{i \neq k \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j^* = i\} \mu_{ji} V_{n,\beta}(D_i \tilde{\mathbf{X}}) \right. \right. \\
& \quad \left. \left. + \left( 1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{i \neq k \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j^* = i\} \mu_{ji} - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] \right) V_{n,\beta}(\tilde{\mathbf{X}}) \right\} \right] \geq \\
& \mathbf{h}(\mathbf{X})^T + \beta \left[ \sum_{i \in \mathcal{N}_c} \lambda_i V_{n,\beta}(A_i \tilde{\mathbf{X}}) + \sum_{j \in \mathcal{N}_s} [\theta_j a_j V_{n,\beta}(B_j \tilde{\mathbf{X}}) + r_j (1 - a_j) V_{n,\beta}(R_j \tilde{\mathbf{X}})] \right. \\
& + \left. \left\{ \sum_{j \neq j' \in \mathcal{N}_s} \mathbb{1}\{u_j^* = k\} \mu_{jk} [V_{n,\beta}(\tilde{\mathbf{X}}) - V_{n,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k)] + \sum_{i \neq k \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j^* = i\} \mu_{ji} V_{n,\beta}(D_i \tilde{\mathbf{X}}) \right. \right. \quad (21) \\
& \quad \left. \left. + \left( 1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{i \neq k \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j^* = i\} \mu_{ji} - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] \right) V_{n,\beta}(\tilde{\mathbf{X}}) \right\} \right],
\end{aligned}$$

where the last inequality is obtained by using the induction assumption. Now we show that (21) is an upper bound for  $V_{n+1,\beta}(\tilde{\mathbf{X}})$ , and hence  $V_{n+1,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) \geq V_{n+1,\beta}(\tilde{\mathbf{X}})$ . To see that (21) is an upper bound for  $V_{n+1,\beta}(\tilde{\mathbf{X}})$ , let  $\mathbf{u}'$  be a policy that uses the same allocation as  $\mathbf{u}^*$  for every class  $i \neq k$  but idles server  $j'$  (i.e.,  $u'_{j'} = 0$  and  $u'_j = u_j^*$  for every  $j \neq j' \in \mathcal{N}_s$ ). Using  $\mathbf{u}'$  and from (21), we have:

$$\begin{aligned}
& V_{n+1,\beta}(\tilde{\mathbf{X}} + \mathbf{e}_k) \geq \mathbf{h}\mathbf{X}^T + \beta \left[ \sum_{i \in \mathcal{N}_c} \lambda_i V_{n,\beta}(A_i \tilde{\mathbf{X}}) + \sum_{j \in \mathcal{N}_s} [\theta_j a_j V_{n,\beta}(B_j \tilde{\mathbf{X}}) + r_j (1 - a_j) V_{n,\beta}(R_j \tilde{\mathbf{X}})] \right. \\
& + \left. \left\{ \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u'_j = i\} \mu_{ji} V_{n,\beta}(D_i \tilde{\mathbf{X}}) \right. \right. \quad (22) \\
& \quad \left. \left. + \left( 1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u'_j = i\} \mu_{ji} - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] \right) V_{n,\beta}(\tilde{\mathbf{X}}) \right\} \right].
\end{aligned}$$

Finally, notice that since  $\mathbf{u}^*$  is admissible at state  $\tilde{\mathbf{X}}$ , the policy  $\mathbf{u}'$  constructed above is admissible (but not necessarily optimal) at state  $\tilde{\mathbf{X}}$  based on (6). Hence, this policy provides an upper bound for  $V_{n+1,\beta}(\tilde{\mathbf{X}})$  defined in (5) and the proof is complete.  $\square$

**Proposition 1** (Non-idling). *There exists an optimal policy (for the finite, infinite, or the average-cost problems) which does not allow for unforced idling.*

**Proof of Proposition 1.** The following sample path argument shows the result for the finite horizon case. The result for the other cases then follows from Theorem 2. Consider first a  $t+n$  period problem. If every optimal policy does not allow for unforced idling, the proof is complete. Otherwise, assume a policy  $\pi^*$  exists that is optimal and allows for unforced idling of a server  $j$ . Let  $t$  denote the first decision epoch at which  $\pi^*$  allows for unforced idling, i.e., it idles server  $j$  although it can assign a class  $k$  job to that server. Construct another policy  $\pi'$  that follows the same allocation as  $\pi^*$  until decision epoch  $t$ , but at  $t$  it assigns server  $j$  to a class  $k$  job. Since preemption is allowed, at  $t+1$  preempt every server and follow the optimal policy  $\pi^*$ . Let  $\tilde{\mathbf{X}}'$  denote the state of the system under policy  $\pi'$ . Notice that until time  $t$  both policies have the same cost. However, from  $t+1$ , the cost of the policy  $\pi'$  is  $V_{n+1,\beta}(\tilde{\mathbf{X}}')$ , and the cost of policy  $\pi^*$  is either  $V_{n,\beta}(\tilde{\mathbf{X}}' + \mathbf{e}_k)$  or  $V_{n+1,\beta}(\tilde{\mathbf{X}}')$ . Hence, using Lemma 1, total cost of policy  $\pi'$  cannot be worse than that of policy  $\pi^*$ . Therefore, since  $\pi^*$  is optimal,  $\pi'$  would be optimal too. Now for the average cost problem notice that:

$$Z^* = \inf_{\pi \in \Pi} \lim_{\beta \rightarrow 1^-} \lim_{n \rightarrow \infty} \psi(1 - \beta) V_{n,\beta}^\pi(\tilde{\mathbf{X}}).$$

Hence, since the above argument holds for every  $n \in \mathbb{Z}^+$  and  $\beta \in (0, 1)$ ,  $Z^{\pi'} \leq Z^{\pi^*}$ . Therefore, if the policy  $\pi^*$  is optimal with respect to average cost criterion, then policy  $\pi'$  constructed above would be optimal for the average cost problem.  $\square$

**Proof of Corollary 1.** First, LP 1 for the “W” can be written as:

$$\text{Max } \tau \tag{23}$$

Subject to:

$$y_{11} \mu_{11} \left( \frac{r_1}{\theta_1 + r_1} \right) \geq \lambda_1 + \tau \tag{24}$$

$$y_{23} \mu_{23} \left( \frac{r_2}{\theta_2 + r_2} \right) \geq \lambda_3 + \tau \tag{25}$$

$$y_{12} \mu_{12} \left( \frac{r_1}{\theta_1 + r_1} \right) + y_{22} \mu_{22} \left( \frac{r_1}{\theta_1 + r_1} \right) \geq \lambda_2 + \tau \tag{26}$$

$$y_{11} + y_{12} \leq 1 \tag{27}$$

$$y_{22} + y_{23} \leq 1 \tag{28}$$

$$y_{11}, y_{12}, y_{22}, y_{23} \geq 0. \tag{29}$$

Let  $\tau^*$  denote the optimal value of the objective function of this LP. Notice that if  $\max\{\rho_1, \rho_3\} > 1$ , then either constraint (24) (together with (27) and (29)), or constraint (25) (together with (28) and (29)), forces  $\tau^* < 0$ . Therefore, by Theorem 1 part (ii) the system is unstable if  $\max\{\rho_1, \rho_3\} > 1$ .

Now suppose  $\max\{\rho_1, \rho_3\} < 1$  and  $(1 - \rho_1)\mu_{12}^{eff} + (1 - \rho_3)\mu_{22}^{eff} > \lambda_2$ . Let  $\epsilon = 1 - \max\{\rho_1, \rho_3\}$  and  $\delta = (1 - \rho_1)\mu_{12}^{eff} + (1 - \rho_3)\mu_{22}^{eff} - \lambda_2$ . Choose  $n \in \mathbb{N}$  large enough such that  $\epsilon/n(\mu_{12}^{eff} + \mu_{22}^{eff}) < \delta$ . Then set  $y_{11} = \rho_1 + \epsilon/n$ ,  $y_{12} = 1 - y_{11} = 1 - \rho_1 - \epsilon/n$ ,  $y_{22} = 1 - \rho_3 - \epsilon/n$ , and  $y_{23} = 1 - y_{22} = \rho_3 + \epsilon/n$ . Then notice that, because  $\epsilon > 0$  and  $\delta > 0$ , the set of variables  $y_{ij}$  constructed above yield a feasible (but not necessarily optimal) solution for the LP with a positive objective value  $\tau$ . Since  $\tau^*$  is the optimal objective value,  $\tau^* \geq \tau > 0$ . Hence, the system is stable by Theorem 1 part (i), and the proof is complete.  $\square$

**Proof of Theorem 3 (Non-Collaborative Case).** We first prove the case where collaboration is not allowed. We show the result for the finite horizon discounted cost problem defined in (5). For the infinite horizon and the average cost criterion, the result would follow from Theorem 2 part (iv). Let  $\mathbf{V}$  be the set of real-valued functions on state space  $\mathcal{S}$  and define functional operators  $T_h, T_{\lambda,a}, T_u, T: \mathbf{V} \rightarrow \mathbf{V}$  as follows:

$$T_h V(\tilde{\mathbf{X}}) = \mathbf{h}\mathbf{X}^T \quad (30)$$

$$T_{\lambda,a} V(\tilde{\mathbf{X}}) = \sum_{i \in \mathcal{N}_c} \lambda_i V(A_i \tilde{\mathbf{X}}) + \sum_{j \in \mathcal{N}_s} [\theta_j a_j V(B_j \tilde{\mathbf{X}}) + r_j (1 - a_j) V(R_j \tilde{\mathbf{X}})] \quad (31)$$

$$T_u V(\tilde{\mathbf{X}}) = \min_{\mathbf{u} \in \mathcal{U}(\tilde{\mathbf{X}})} \left\{ \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \mu_{ji} V(D_i \tilde{\mathbf{X}}) + \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \mu_{ji} - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}}) \right\} \quad (32)$$

$$= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}}) - \max_{\mathbf{u} \in \mathcal{U}(\tilde{\mathbf{X}})} \left\{ \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \mu_{ji} \Delta_i V(D_i \tilde{\mathbf{X}}) \right\} \quad (33)$$

$$TV(\tilde{\mathbf{X}}) = T_h V(\tilde{\mathbf{X}}) + \beta [T_{\lambda,a} V(\tilde{\mathbf{X}}) + T_u V(\tilde{\mathbf{X}})]. \quad (34)$$

From (5) and (34), and after fixing discount factor  $\beta$ , the N-Period problem satisfies the optimality equation:

$$V_n(\tilde{\mathbf{X}}) = TV_{n-1}(\tilde{\mathbf{X}}) \quad n \in \{1, \dots, N\}, \quad (35)$$

with terminal condition  $V_0(\cdot) = 0$ .

Define  $\mathcal{A}_1, \mathcal{A}_2 \subset \mathcal{S}$  as:

$$\mathcal{A}_1 = \{\tilde{\mathbf{X}} \in \mathcal{S} : a_1 = 1, x_1, x_2 \geq 1\} \text{ and } \mathcal{A}_2 = \{\tilde{\mathbf{X}} \in \mathcal{S} : a_2 = 1, x_2, x_3 \geq 1\}. \quad (36)$$

Now suppose the condition of the theorem holds. Hence, after uniformization we have  $h_1 \mu_{11} \geq h_2 \mu_{12}$ ,  $h_3 \mu_{23} \geq h_2 \mu_{22}$ ,  $\mu_{12} \geq \mu_{11}$ , and  $\mu_{22} \geq \mu_{23}$ . First notice that if the shared queue is empty (i.e.,

if  $x_2 = 0$ ) the result of the theorem is trivial since unforced idling is suboptimal by Proposition 1. Thus, it remains to show that it is optimal (1) for server 1 to serve its fixed task for every  $\tilde{\mathbf{X}} \in \mathcal{A}_1$ , and (2) for server 2 to serve its fixed task for every  $\tilde{\mathbf{X}} \in \mathcal{A}_2$ . From (33), to show that the optimal allocation in operator  $T_u$  follows these rules, it is sufficient to show that the following properties hold:

$$\begin{aligned} (i) \quad & \mu_{11}\Delta_1V(\tilde{\mathbf{X}} - \mathbf{e}_1) \geq \mu_{12}\Delta_2V(\tilde{\mathbf{X}} - \mathbf{e}_2) \quad \text{for all } \tilde{\mathbf{X}} \in \mathcal{A}_1, \\ (ii) \quad & \mu_{23}\Delta_3V(\tilde{\mathbf{X}} - \mathbf{e}_3) \geq \mu_{22}\Delta_2V(\tilde{\mathbf{X}} - \mathbf{e}_2) \quad \text{for all } \tilde{\mathbf{X}} \in \mathcal{A}_2. \end{aligned} \tag{37}$$

Now, let  $\mathcal{V} \subset \mathbf{V}$  be the set of real-valued functions such that if  $V \in \mathcal{V}$  then  $V$  satisfies properties (i) and (ii) given in (37). Notice that, since  $V_0(\cdot) = 0$ ,  $V_0 \in \mathcal{V}$ . Hence, from Lemma 2 (see below) and (35) we have  $V_n \in \mathcal{V}$  ( $\forall n \in \mathbb{Z}^+$ ). Thus, the proof is complete for the finite-horizon problem. The proof for infinite-horizon and average cost scenarios then follow from Theorem 2 part (iv).  $\square$

**Lemma 2 [Preservation (Non-Collaborative Case)].** Suppose  $h_1\mu_{11} \geq h_2\mu_{12}$ ,  $h_3\mu_{23} \geq h_2\mu_{22}$ ,  $\mu_{12} \geq \mu_{11}$ , and  $\mu_{22} \geq \mu_{23}$  hold and  $V \in \mathcal{V}$ . Then  $TV \in \mathcal{V}$ . That is, operator  $T$  preserves properties (i) and (ii) in (37) when  $h_1\mu_{11} \geq h_2\mu_{12}$ ,  $h_3\mu_{23} \geq h_2\mu_{22}$ ,  $\mu_{12} \geq \mu_{11}$ , and  $\mu_{22} \geq \mu_{23}$ .

**Proof.** We first show that, under the conditions of this lemma,  $TV$  satisfies property (i) in (37). To show that, we divide  $\mathcal{A}_1$  to five partitions  $\mathcal{A}_1^1, \mathcal{A}_1^2, \dots, \mathcal{A}_1^5$ , where

$$\begin{aligned} \mathcal{A}_1^1 &= \{\tilde{\mathbf{X}} \in \mathcal{A}_1 : x_1 \geq 2, x_2 \geq 1, x_3 \geq 1\}, \\ \mathcal{A}_1^2 &= \{\tilde{\mathbf{X}} \in \mathcal{A}_1 : x_1 \geq 2, x_2 \geq 1, x_3 = 0\}, \\ \mathcal{A}_1^3 &= \{\tilde{\mathbf{X}} \in \mathcal{A}_1 : x_1 = 1, x_2 \geq 1, x_3 \geq 1\}, \\ \mathcal{A}_1^4 &= \{\tilde{\mathbf{X}} \in \mathcal{A}_1 : x_1 = 1, x_2 \geq 2, x_3 = 0\}, \\ \mathcal{A}_1^5 &= \{\tilde{\mathbf{X}} \in \mathcal{A}_1 : x_1 = 1, x_2 = 1, x_3 = 0\}. \end{aligned}$$

Now, for each partition we show that  $TV$  satisfies property (i) in (37), i.e., we show that  $\mu_{11}\Delta_1TV(\tilde{\mathbf{X}} - \mathbf{e}_1) \geq \mu_{12}\Delta_2TV(\tilde{\mathbf{X}} - \mathbf{e}_2)$  holds for each partition.

Case 1 ( $\tilde{\mathbf{X}} \in \mathcal{A}_1^1$ )

Since  $V \in \mathcal{V}$ , from (33) we have:

$$\begin{aligned}
T_u V(\tilde{\mathbf{X}}) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}}) \\
&\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - a_2 \mu_{23} \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3) \\
T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - 2\mathbf{e}_1) - a_2 \mu_{23} \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3) \\
T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{23} \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3)
\end{aligned}$$

From above:

$$\begin{aligned}
\Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - 2\mathbf{e}_1)) - a_2 \mu_{23} (\Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3) - \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3)) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - 2\mathbf{e}_1)) - a_2 \mu_{23} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3)),
\end{aligned}$$

where the last equality is obtained since  $\Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3) - \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3) = \Delta_3 \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3) = \Delta_1 \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3) = \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3)$ .

Similarly,

$$\begin{aligned}
\Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) - a_2 \mu_{23} (\Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3) - \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3)) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) - a_2 \mu_{23} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3)).
\end{aligned}$$

where the last equality is obtained since  $\Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3) - \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3) = \Delta_3 \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3) = \Delta_2 \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3) = \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3)$ .

Therefore, from the above and after simplification we have:

$$\begin{aligned}
& \mu_{11}\Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) = \\
& \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{11} - a_2 \mu_{23}\right) [\mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)] \\
& + \mu_{11} [\mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - 2\mathbf{e}_1) - \mu_{12}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)] \\
& + a_2 \mu_{23} [\mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3) - \mu_{12}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3)] \\
& \geq 0,
\end{aligned}$$

where the inequality holds, since every term is nonnegative as  $V \in \mathcal{V}$  (and hence  $V$  satisfies property (i)). Thus, operator  $T_u$  preserves property (i) in this case. (It should be clear that the coefficient of the first term is nonnegative due to uniformization). Also, notice that clearly operator  $T_{\lambda,a}$  preserves this property as well. Moreover,  $\mu_{11}\Delta_1 T_h V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 T_h V(\tilde{\mathbf{X}} - \mathbf{e}_2) = h_1 \mu_{11} - h_2 \mu_{12} \geq 0$ . Hence, all of the operators  $T_h$ ,  $T_{\lambda,a}$ , and  $T_u$  preserve property (i). Thus, operator  $T = T_h + \beta[T_{\lambda,a} + T_u]$  preserves property (i) in (37).

### Case 2 ( $\tilde{\mathbf{X}} \in \mathcal{A}_1^2$ )

Since  $V \in \mathcal{V}$ , for this case from (33) we have:

$$\begin{aligned}
T_u V(\tilde{\mathbf{X}}) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}}) \\
&\quad - \mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - a_2 \mu_{22}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - 2\mathbf{e}_1) - a_2 \mu_{22}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) \\
T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{22} \mathbf{1}\{x_2 \geq 2\} \Delta_3 V(\tilde{\mathbf{X}} - 2\mathbf{e}_2)
\end{aligned}$$

Therefore,

$$\begin{aligned}
\Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - 2\mathbf{e}_1)) - a_2 \mu_{22} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - 2\mathbf{e}_1)) - a_2 \mu_{22} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)),
\end{aligned}$$

where the last equality is obtained since  $\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) = \Delta_2 \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) = \Delta_1 \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) = \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)$ .

Also,

$$\begin{aligned}
\Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) \\
&\quad - a_2 \mu_{22} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \mathbf{1}\{x_2 \geq 2\} \Delta_2 V(\tilde{\mathbf{X}} - 2\mathbf{e}_2)) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) \\
&\quad - a_2 \mu_{22} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \mathbf{1}\{x_2 \geq 2\} \Delta_2 V(\tilde{\mathbf{X}} - 2\mathbf{e}_2))
\end{aligned}$$

where the last equality is obtained since  $\Delta_1 \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) = \Delta_2 \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)$ , as introduced above in this case.

Hence, from above and after simplification we have:

$$\begin{aligned}
&\mu_{11} \Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12} \Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) = \\
&\left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{11} - a_2 \mu_{22}\right) [\mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)] \\
&\quad + \mu_{11} [\mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - 2\mathbf{e}_1) - \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)] \\
&\quad + a_2 \mu_{22} [\mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - \mu_{12} \mathbf{1}\{x_2 \geq 2\} \Delta_2 V(\tilde{\mathbf{X}} - 2\mathbf{e}_2)] \\
&\geq 0,
\end{aligned}$$

where the inequality holds, since if  $\mathbf{1}\{x_2 \geq 2\} = 1$ , every term is nonnegative as  $V \in \mathcal{V}$  (and hence  $V$  satisfies property (i)). Also, if  $\mathbf{1}\{x_2 \geq 2\} = 0$ , then the first and second term are nonnegative as  $V \in \mathcal{V}$ , and the last term is nonnegative since  $\Delta_1 V(\cdot) \geq 0$  from Lemma 1. Thus, operator  $T_u$  preserves property (i) in this case. Also, notice that clearly operator  $T_{\lambda,a}$  preserves this property. Moreover,  $\mu_{11} \Delta_1 T_h V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12} \Delta_2 T_h V(\tilde{\mathbf{X}} - \mathbf{e}_2) = h_1 \mu_{11} - h_2 \mu_{12} \geq 0$ , and thus, operator  $T_h$  preserves property (i). Hence, similar to Case 1, all of the operators  $T_h$ ,  $T_{\lambda,a}$ , and  $T_u$  preserve property (i). Thus, operator  $T = T_h + \beta[T_{\lambda,a} + T_u]$  preserves property (i) in (37).

Case 3 ( $\tilde{\mathbf{X}} \in \mathcal{A}_1^3$ )

Since  $V \in \mathcal{V}$ , from (33) we have:

$$\begin{aligned}
T_u V(\tilde{\mathbf{X}}) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}}) \\
&\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - a_2 \mu_{23} \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3) \\
T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{23} \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3) \\
T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{23} \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3)
\end{aligned}$$

From above:

$$\begin{aligned}
\Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) + \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{23} (\Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3) - \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3)) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) + \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{23} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3)),
\end{aligned}$$

where the last equality is obtained since  $\Delta_3 \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3) = \Delta_1 \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3)$ , as introduced in Case 1.

Similarly,

$$\begin{aligned}
\Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) - a_2 \mu_{23} (\Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3) - \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3)) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) - a_2 \mu_{23} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3)).
\end{aligned}$$

where the last equality is obtained since  $\Delta_3 \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3) = \Delta_2 \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3)$ , as introduced in Case 1.

Therefore, from the above and after simplification we have:

$$\begin{aligned}
& \mu_{11}\Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) = \\
& (1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{11} - a_2 \mu_{23}) [\mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)] \\
& + a_2 \mu_{23} [\mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_3) - \mu_{12}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2 - \mathbf{e}_3)] \\
& \geq 0,
\end{aligned}$$

where the inequality holds, since every term is nonnegative as  $V \in \mathcal{V}$  (and hence  $V$  satisfies property (i)). Thus, operator  $T_u$  preserves property (i) in this case. Also, it is trivial that operator  $T_{\lambda,a}$  preserves this property. Moreover,  $\mu_{11}\Delta_1 T_h V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 T_h V(\tilde{\mathbf{X}} - \mathbf{e}_2) = h_1 \mu_{11} - h_2 \mu_{12} \geq 0$ . Hence, similar to the previous cases, all of the operators  $T_h$ ,  $T_{\lambda,a}$ , and  $T_u$  preserve property (i). Thus, operator  $T = T_h + \beta[T_{\lambda,a} + T_u]$  preserves property (i) in (37).

#### Case 4 ( $\tilde{\mathbf{X}} \in \mathcal{A}_1^4$ )

Since  $V \in \mathcal{V}$ , for this case from (33) we have:

$$\begin{aligned}
T_u V(\tilde{\mathbf{X}}) &= (1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]) V(\tilde{\mathbf{X}}) \\
&\quad - \mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - a_2 \mu_{22}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= (1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]) V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{12}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{22}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) \\
T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= (1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]) V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{22}\Delta_2 V(\tilde{\mathbf{X}} - 2\mathbf{e}_2)
\end{aligned}$$

Therefore,

$$\begin{aligned}
\Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&= (1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]) \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) + \mu_{12}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{22} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) \\
&= (1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]) \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - 2\mathbf{e}_1)) - a_2 \mu_{22} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)),
\end{aligned}$$

where the last equality is obtained since  $\Delta_2 \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) = \Delta_1 \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)$ , as introduced in Case 2.

Similarly,

$$\begin{aligned}
\Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) \\
&\quad - a_2 \mu_{22} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - 2\mathbf{e}_2)). \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) \\
&\quad - a_2 \mu_{22} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - 2\mathbf{e}_2)).
\end{aligned}$$

where the last equality is obtained since  $\Delta_1 \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) = \Delta_2 \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)$ , as introduced in Case 2.

Hence, from above and after simplification we have:

$$\begin{aligned}
&\mu_{11} \Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12} \Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) = \\
&\left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{11} - a_2 \mu_{22}\right) [\mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)] \\
&\quad + \mu_{11} [\mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - 2\mathbf{e}_1) - \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)] \\
&\quad + a_2 \mu_{22} [\mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - 2\mathbf{e}_2)] \\
&\geq 0,
\end{aligned}$$

where the inequality holds since every term is nonnegative as  $V \in \mathcal{V}$  (and hence  $V$  satisfies property (i)). Thus, operator  $T_u$  preserves property (i) in this case. Also, notice that clearly operator  $T_{\lambda,a}$  preserves this property. Moreover,  $\mu_{11} \Delta_1 T_h V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12} \Delta_2 T_h V(\tilde{\mathbf{X}} - \mathbf{e}_2) = h_1 \mu_{11} - h_2 \mu_{12} \geq 0$ , and thus, operator  $T_h$  preserves property (i). Hence, similar to the previous cases, all of the operators  $T_h$ ,  $T_{\lambda,a}$ , and  $T_u$  preserve property (i). Thus, operator  $T = T_h + \beta[T_{\lambda,a} + T_u]$  preserves property (i) in (37).

#### Case 5 ( $\tilde{\mathbf{X}} \in \mathcal{A}_1^5$ )

Since  $V \in \mathcal{V}$ , from (33) we have:

$$\begin{aligned}
T_u V(\tilde{\mathbf{X}}) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}}) \\
&\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - a_2 \mu_{22} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)
\end{aligned}$$

$$\begin{aligned}
T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \max\{\mu_{12}, a_2 \mu_{22}\} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) \\
T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2).
\end{aligned}$$

Therefore,

$$\begin{aligned}
\Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\
&\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) + \max\{\mu_{12}, a_2 \mu_{22}\} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{22} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2),
\end{aligned}$$

and

$$\begin{aligned}
\Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) - a_2 \mu_{22} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&\quad - \mu_{11} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) - a_2 \mu_{22} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2).
\end{aligned}$$

where the last equality is obtained since  $\Delta_1 \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) = \Delta_2 \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)$ , as introduced in Case 2.

Then, from above and after simplification we obtain:

$$\begin{aligned}
&\mu_{11} \Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12} \Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) = \\
&\left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{11}\right) [\mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)] \\
&\quad + a_2 \mu_{22} (\mu_{12} - \mu_{11}) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) + \mu_{11} (\max\{\mu_{12}, a_2 \mu_{22}\} - \mu_{12}) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) \\
&\geq 0,
\end{aligned}$$

where the inequality holds since all the terms in the last expression is nonnegative. The first term is nonnegative as (1) its coefficient is nonnegative due to uniformization, and (2)  $V \in \mathcal{V}$  (and hence  $V$  satisfies property (i)). Similarly, the second term is also nonnegative as (1)  $\mu_{12} \geq \mu_{11}$  is a condition

of the lemma, and (2)  $\Delta_2 V(\cdot) \geq 0$  by Lemma 1. Finally, the third term is nonnegative as (1)  $\max\{\mu_{12}, a_2 \mu_{22}\} - \mu_{12} \geq 0$  always holds (note that this also covers the cases with  $a_2 = 0$ ), and (2)  $\Delta_2 V(\cdot) \geq 0$  by Lemma 1.

Therefore, operator  $T_u$  preserves property (i). Also, notice that clearly operator  $T_{\lambda,a}$  preserves this property. Moreover,  $\mu_{11} \Delta_1 T_h V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12} \Delta_2 T_h V(\tilde{\mathbf{X}} - \mathbf{e}_2) = h_1 \mu_{11} - h_2 \mu_{12} \geq 0$ , and thus, operator  $T_h$  preserves property (i). Hence, similar to the previous cases, all of the operators  $T_h$ ,  $T_{\lambda,a}$ , and  $T_u$  preserve property (i). Thus, operator  $T = T_h + \beta[T_{\lambda,a} + T_u]$  preserves property (i) in (37).

From the proofs for Cases 1 to 5 above, operator  $T$  preserves property (i) for  $\tilde{\mathbf{X}} \in \mathcal{A}_1^j$  ( $j = 1, \dots, 5$ ). Hence,  $T$  preserves this property for  $\tilde{\mathbf{X}} \in \cup_{j=1}^5 \mathcal{A}_1^j = \mathcal{A}_1$ . It remains to show that  $T$  preserves property (ii) in (37) for all  $\tilde{\mathbf{X}} \in \mathcal{A}_2$ . To prove this, notice that because of the complete symmetry in the “W,” the proof would follow exactly the same lines as the proof for preservation of property (i), only after *relabeling* servers and customer classes. (Notice that properties (i) and (ii), and all the conditions of the lemma are symmetric). Therefore,  $T$  also preserves property (ii) for all  $\tilde{\mathbf{X}} \in \mathcal{A}_2$  and the proof is complete.  $\square$

**Proof of Theorem 3 (Collaborative Case).** When servers can collaborate while serving jobs, majority of the proof of Theorem 3 and Lemma 2 stay the same. The functional operators  $T_h$ ,  $T_{\lambda,a}$ ,  $T_u$ , and  $T$  and the optimality equation provided in (30) to (35) of the Theorem 3 proof stay the same, but in order to allow for collaboration, we modify the set of admissible control actions at  $\tilde{\mathbf{X}}$  given in (6) as follows:

$$\mathcal{U}(\tilde{\mathbf{X}}) = \left\{ \mathbf{u} = (u_j \in \mathcal{N}_c \cup \{0\} \text{ s.t. } \forall i \in \mathcal{N}_c : \mathbf{1}\{u_j = i\} \leq a_j \mathbf{1}\{i \in \mathcal{S}_j\}, \sum_{j \in \mathcal{N}_s} \mathbf{1}\{u_j = i\} \leq |\mathcal{N}_s| \mathbf{1}\{X_i > 0\}) \right\}.$$

Let  $\mathcal{A}_1, \mathcal{A}_2 \subset \mathcal{S}$  be as defined in (36). Now suppose we have  $h_1 \mu_{11} \geq h_2 \mu_{12}$ ,  $h_3 \mu_{23} \geq h_2 \mu_{22}$ , and server collaboration while serving jobs is allowed. Due to the skill structure of the “W,” servers can only collaborate on the shared task jobs. Note that, if the shared queue is empty (i.e., if  $x_2 = 0$ ) the result of the theorem is trivial since unforced idling is suboptimal by Proposition 1. Thus, it remains to show that it is optimal (1) for server 1 to serve its fixed task for every  $\tilde{\mathbf{X}} \in \mathcal{A}_1$ , and (2) for server 2 to serve its fixed task for every  $\tilde{\mathbf{X}} \in \mathcal{A}_2$ . From (33), to show that the optimal allocation in operator  $T_u$  follows these rules, it is sufficient to show that the following properties hold:

$$\begin{aligned} (i) \quad & \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \geq \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \quad \text{for all } \tilde{\mathbf{X}} \in \mathcal{A}_1, \\ (ii) \quad & \mu_{23} \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3) \geq \mu_{22} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \quad \text{for all } \tilde{\mathbf{X}} \in \mathcal{A}_2. \end{aligned} \tag{38}$$

which are the same properties used in the proofs of Theorem 3 and Lemma 2. Properties (i) and (ii) ensure that the server collaboration is the unique optimal action only when (both servers are

available and)  $\mathbf{X} = (0, 1, 0)$  (where both servers are assigned to fixed task job) and servers follow the  $c\mu$  rule, as stated in the Theorem 3, in the rest of the states. To see this, assume  $V$  satisfies properties (i) and (ii). As unforced idling is not optimal,  $T_u$  in (33) is as follows:

$$T_u V(\tilde{\mathbf{X}}) = \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}}) - \max_{\mathbf{u} \in \mathcal{U}(\tilde{\mathbf{X}})} \left\{ \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j = i\} \mu_{ji} \Delta_i V(D_i \tilde{\mathbf{X}}) \right\},$$

where (when both servers are available)  $\max_{\mathbf{u} \in \mathcal{U}(\tilde{\mathbf{X}})} \left\{ \sum_{i \in \mathcal{N}_c} \sum_{j \in \mathcal{N}_s} \mathbb{1}\{u_j = i\} \mu_{ji} \Delta_i V(D_i \tilde{\mathbf{X}}) \right\} =$

$$\begin{aligned} & \max \left\{ \mathbb{1}\{x_2 \geq 1\} ((\mu_{12} + \mu_{22}) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)), \mathbb{1}\{x_2 \geq 2\} (\mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) + \mu_{22} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)), \right. \\ & \quad \mathbb{1}\{x_1, x_3 \geq 1\} (\mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) + \mu_{23} \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3)), \\ & \quad \mathbb{1}\{x_2, x_3 \geq 1\} (\mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) + \mu_{23} \Delta_3 V(\tilde{\mathbf{X}} - \mathbf{e}_3)), \\ & \quad \left. \mathbb{1}\{x_1, x_2 \geq 1\} (\mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) + \mu_{22} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)) \right\} \end{aligned} \quad (39)$$

The first and the second options in the maximum function are collaboration and assigning each server to separate shared task jobs (if possible), respectively. Properties (i) and (ii) ensure that the first two actions in the maximum function are never the unique optimal action as long as one of the other server assignments is allowable at that state. Therefore, except for states with  $\mathbf{X} = (0, x_2, 0)$  where  $x_2 \geq 1$ , we can ignore the first two actions and construct an optimal policy. For states  $\mathbf{X} = (0, x_2, 0)$  where  $x_2 \geq 2$ , the first two terms in (39) are equal. Therefore, as long as  $x_2 \geq 2$ , we can construct an optimal policy that ignores the collaboration action. As a result, if  $V$  satisfies properties (i) and (ii), then collaboration is only uniquely optimal at  $\mathbf{X} = \{0, 1, 0\}$ . Now, similar to the proof of the non-collaborative case, let  $\mathcal{V} \subset \mathbf{V}$  be the set of real-valued functions such that if  $V \in \mathcal{V}$  then  $V$  satisfies properties (i) and (ii). Notice that, since  $V_0(\cdot) = 0$ ,  $V_0 \in \mathcal{V}$ . Hence, from Lemma 3 (see below) and (35) we have  $V_n \in \mathcal{V} (\forall n \in \mathbb{Z}^+)$ . Thus, the proof is complete for the finite-horizon problem. The proof for infinite-horizon and average cost scenarios then follow from Theorem 2 part (iv) (notice that Theorem 2 is proved under a non-collaborations assumption, but is also true for the collaborative case).  $\square$

**Lemma 3 [Preservation (Collaborative Case)].** Suppose  $h_1 \mu_{11} \geq h_2 \mu_{12}$ ,  $h_3 \mu_{23} \geq h_2 \mu_{22}$ , server collaboration is allowed, and  $V \in \mathcal{V}$ . Then  $TV \in \mathcal{V}$ . That is, under these conditions, operator  $T$  preserves properties (i) and (ii) in (38).

**Proof.** Note that we can use the same partitioning used in the proof of Lemma 2, and in addition, since  $V \in \mathcal{V}$ , collaboration is the unique optimal action only when (both servers are available and)  $\mathbf{X} = \{0, 1, 0\}$ . Hence, the proofs for the first four partitions  $\mathcal{A}_1^1$  to  $\mathcal{A}_1^4$  will not change. In partition  $\mathcal{A}_1^5$

however,  $T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) = T_u V(0, 1, 0, a_1, a_2)$  and therefore collaboration is optimal when  $a_1 = a_2 = 1$ . Below we present the proof of Case 5 ( $\tilde{\mathbf{X}} \in \mathcal{A}_1^5$ ) for the collaborative case.

Case 5 ( $\tilde{\mathbf{X}} \in \mathcal{A}_1^5$ )

Under the server collaboration assumption, it is optimal to assign both servers to the fixed task when  $\mathbf{X} = (0, 1, 0)$  and  $a_1 = a_2 = 1$ . Therefore,  $T_u V(\tilde{\mathbf{X}})$  and  $T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2)$  of Case 5 ( $\tilde{\mathbf{X}} \in \mathcal{A}_1^5$ ) proof for the non-collaborative case stay the same, but (regardless of whether  $a_2 = 0$  or  $a_2 = 1$ ) the  $\max\{\mu_{12}, a_2\mu_{22}\}$  term of  $T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1)$  is replaced with  $\mu_{12} + a_2\mu_{22}$ . Hence, since  $V \in \mathcal{V}$ , from (33) we have:

$$\begin{aligned} T_u V(\tilde{\mathbf{X}}) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}}) \\ &\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - a_2 \mu_{22} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\ T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\ &\quad - (\mu_{12} + a_2 \mu_{22}) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) \\ T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\ &\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2). \end{aligned}$$

Therefore,

$$\begin{aligned} \Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\ &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) \\ &\quad - \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) + (\mu_{12} + a_2 \mu_{22}) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) - a_2 \mu_{22} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2), \end{aligned}$$

and

$$\begin{aligned} \Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) &= T_u V(\tilde{\mathbf{X}}) - T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\ &= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)]\right) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\ &\quad - \mu_{11} (\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) - a_2 \mu_{22} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2). \end{aligned}$$

Then, from above and after simplification we obtain:

$$\begin{aligned}
& \mu_{11}\Delta_1 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 T_u V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{11}\right) [\mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)] \\
&+ a_2 \mu_{22} (\mu_{12} - \mu_{11}) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) + a_2 \mu_{22} \mu_{11} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{11} - a_2 \mu_{22}\right) [\mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)] \\
&+ a_2 \mu_{22} \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - a_2 \mu_{22} \mu_{12} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) + a_2 \mu_{22} (\mu_{12} - \mu_{11}) \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) \\
&+ a_2 \mu_{22} \mu_{11} \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{11} - a_2 \mu_{22}\right) [\mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)] \\
&+ a_2 \mu_{22} \mu_{11} (\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2) + \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)) \\
&= \left(1 - \sum_{i \in \mathcal{N}_c} \lambda_i - \sum_{j \in \mathcal{N}_s} [\theta_j a_j + r_j (1 - a_j)] - \mu_{11} - a_2 \mu_{22}\right) [\mu_{11}\Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1) - \mu_{12}\Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_2)] \\
&+ a_2 \mu_{22} \mu_{11} \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) \geq 0, \tag{40}
\end{aligned}$$

where the last term is obtained from  $\Delta_1 \Delta_2 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2) = \Delta_2 \Delta_1 V(\tilde{\mathbf{X}} - \mathbf{e}_1 - \mathbf{e}_2)$ , as introduced in Case 2 in the proof of the non-collaborative case.

Note that the inequality in (40) holds since all the terms in the last expression are nonnegative. The first term of (40) is nonnegative as (1) its coefficient is nonnegative due to uniformization even when  $a_2 = 1$ , and (2)  $V \in \mathcal{V}$  (and hence  $V$  satisfies property (i)). Similarly, the second term is also nonnegative as  $\Delta_1 V(\cdot) \geq 0$  by Lemma 1 (which is true when collaboration is allowed).

Hence, by following the same steps employed at the end of the proof for the non-collaborative case, we can conclude that the operator  $T$  preserves properties (i) and (ii) and the proof is complete.  $\square$

**Proof of Theorem 4.** Let  $\tilde{\tau}^*$  and  $y_{ji}^*$  denote the optimal objective function and the corresponding solutions of LP 2, respectively. Suppose  $\tau^* > 0$  (or equivalently  $\tilde{\tau}^* > 0$ ). Then solutions  $y_{ji}^*$  satisfy:

$$\sum_{j \in \mathcal{S}_i^{-1}} y_{ji}^* \left( \frac{r_j}{\theta_j + r_j} \right) \mu_{ji} \geq \lambda_i + \tau^* \quad \forall i \in \mathcal{N}_c, \tag{41}$$

$$\sum_{i \in \mathcal{S}_j} y_{ji}^* \leq 1 \quad \forall j \in \mathcal{N}_s, \tag{42}$$

$$y_{ji} \geq 0 \quad \forall j \in \mathcal{N}_s, \forall i \in \mathcal{S}_j. \tag{43}$$

Define  $\kappa_i = h_i [\sum_{j \in \mathcal{S}_i^{-1}} y_{ji}^* (\frac{r_j}{\theta_j + r_j}) \mu_{ji}]^{-1}$  so that  $\mathcal{J}_i(x_i) = \kappa_i x_i$ , where  $\mathcal{J}_i(\cdot)$  is the LEWC index of queue  $i$ . Next suppose the system under LEWC, denoted  $\Gamma$ , is unstable (i.e.,  $Z^\Gamma = \infty$ ) and, therefore, at

least one of the queues does not have finite average queue length. Let  $\mathcal{M} \neq \emptyset$  denote the set of such queues. That is,  $\mathcal{M} = \{i \in \mathcal{N}_c : L_i^\Gamma = \infty\}$ . We prove that LEWC stabilizes the system by showing that  $\mathcal{M} = \emptyset$  (which is a trivial contradiction).

To show this, we use a fluid model analysis (see, for instance, Dai and Meyn (1995) and Dai (1999)). The goal is to show that the analog fluid model of the system drains in a finite time. To this end, fix the control policy and let  $Q_i(t)$  and  $T_{j,i}(t)$  denote the length of queue  $i$  at time  $t > 0$  and the time server  $j$  spends actively working on class  $i$  up to  $t$ , respectively. Next, let  $(\bar{Q}_i(t), \bar{T}_{j,i}(t) : i \in \mathcal{N}_c; j \in \mathcal{N}_s)$  denote the *fluid limit* of the system (i.e., limit point of  $(q^{-1}Q_i(qt), q^{-1}T_{j,i}(qt) : i \in \mathcal{N}_c; j \in \mathcal{N}_s)$  when  $q \rightarrow \infty$ ). First, assume  $\mathcal{M} = \{1\}$ , i.e., only queue 1 is unstable. We will show that it results in a contradiction by showing that if  $\mathcal{M} = \{1\}$ , then (under LEWC) queue 1 will indeed drain in a finite time. An important step is to show that for any time  $t$  with  $\bar{Q}_1(t) > 0$  the length of queue 1 is decreasing on average (and as a result this queue will be drained in a finite time). To show this rigorously, suppose  $\bar{Q}_1(t) > 0$  for some  $t$ . From page 20 of Dai (1999), every component of the fluid model including  $\bar{Q}_1(t)$  is Lipschitz continuous, and thus, (similar to the argument used in the proof of Dai and Meyn (1995) Proposition 4.2 part (vi)) we can choose a subsequence  $\{q_n^{-1}Q_1(q_n s), n \geq 1\}$  with  $q_n \rightarrow \infty$  as  $n \rightarrow \infty$ , such that  $q_n^{-1}Q_1(q_n s) \rightarrow \bar{Q}_1(s) > 0$  uniformly on a small interval  $s \in [t, t+h]$ . Therefore, for any  $M \in \mathbb{R}^+$  there exists  $N \in \mathbb{N}$  such that  $Q_1(q_n s) > M$  for all  $n \geq N$  and all  $s \in [t, t+h]$ . Since queue 2 is stable, take  $M = \kappa_2/\kappa_1 \sup_{q_n s} Q_2(q_n s)$ . This choice of  $M$  implies that  $\kappa_1 Q_1(q_n s) > \kappa_2 Q_2(q_n s)$  for all  $n \geq N$ , during interval  $[t, t+h]$ , server 1 spends all his (available) time on queue 1 under LEWC (because queue 1's LEWC index is larger than that of queue 2). Using the definition of derivative, we have  $\frac{d}{du} \bar{T}_{1,1}(u)|_{u=t} = \frac{r_1}{\theta_1 + r_1}$ . That is in the fluid model the departure rate of queue 1 at time  $t$  is  $\bar{D}_1(t) = \mu_{11} \frac{d}{du} \bar{T}_{1,1}(u)|_{u=t} = \mu_{11} \frac{r_1}{\theta_1 + r_1}$ . Moreover, since  $y_{11}^*$  satisfies (41)-(43),  $\bar{D}_1(t) = \mu_{11} \frac{r_1}{\theta_1 + r_1} \geq y_{11}^* \mu_{11} \left(\frac{r_1}{\theta_1 + r_1}\right) \geq \lambda_1 + \tau^*$ , where  $\tau^* > 0$ . In other words, the departure rate is (strictly) greater than the arrival rate (the queue length is decreasing) at any time  $t$  with  $Q_1(t) > 0$ . Hence, based on Theorem 2.4.9 of Dai (1999) this queue drains in a finite time. Therefore, from Theorem 4.1 of Dai and Meyn (1995) (which connects the stability of the fluid model with that of the original underlying Markov process),  $L_1^\Gamma < \infty$ . This is a contradiction with  $\mathcal{M} = \{1\}$ . Thus,  $\mathcal{M} \neq \{1\}$ . A similar discussion shows that  $\mathcal{M} \neq \{3\}$  (because of the symmetry in the ‘‘W’’, it only requires a change in the notation: consider queue 3 as queue 1, and server 2 as server 1).

Next assume  $\mathcal{M} = \{2\}$  i.e., queue 2 (the shared queue) is the only unstable queue. Suppose  $\bar{Q}_2(t) > 0$  for some  $t > 0$ . Using a similar discussion, we can choose a subsequence  $\{q_n^{-1}Q_2(q_n s), n \geq 1\}$  with  $q_n \rightarrow \infty$  as  $n \rightarrow \infty$ , such that  $q_n^{-1}Q_2(q_n s) \rightarrow \bar{Q}_2(s) > 0$  on a small interval  $s \in [t, t+h]$ . Therefore, for any  $M \in \mathbb{R}^+$  there exists  $N \in \mathbb{N}$  such that  $Q_2(q_n s) > M$  for all  $n \geq N$  and all

$s \in [t, t + h]$ . Now take  $M = \max\{\kappa_1/\kappa_2 \sup_{q_n s} Q_1(q_n s), \kappa_3/\kappa_2 \sup_{q_n s} Q_3(q_n s)\}$  and notice that under LEWC both servers 1 and 2 spend their time on queue 2 during a small interval  $[t, t+h]$ . Therefore, we have  $\frac{d}{du}[\bar{T}_{1,2}(u) + \bar{T}_{2,2}(u)]|_{u=t} = \sum_{j=1,2} \frac{r_j}{\theta_j + r_j}$ . That is, in the fluid model, the effective departure rate of queue 2 at time  $t$  is  $\bar{D}_2(t) = \mu_{11} \frac{r_1}{\theta_1 + r_1} + \mu_{22} \frac{r_2}{\theta_2 + r_2}$ . Moreover, since  $y_{12}^*$  and  $y_{22}^*$  satisfy (41)-(43), we have  $\bar{D}_2(t) \geq y_{11}^* \mu_{11} \frac{r_1}{\theta_1 + r_1} + y_{22}^* \mu_{22} \frac{r_2}{\theta_2 + r_2} \geq \lambda_2 + \tau^*$ , where  $\tau^* > 0$ . Hence, based on Theorem 2.4.9 of Dai (1999), queue 2 drains in a finite time. Therefore, from Dai and Meyn (1995) Theorem 4.1,  $L_2^\Gamma < \infty$  which is a contradiction with  $\mathcal{M} = \{2\}$ .

Next assume  $\mathcal{M} = \{1, 2\}$  and suppose  $\bar{Q}_2(t) > 0$  for some  $t > 0$ . Similar to the above, first notice that in the fluid model, server 2 spends all his time on queue 2 during a small interval  $[t, t+h]$ . Moreover, we can always choose a sequence  $\{q_m, m \geq 1\}$  and  $L \in \mathbb{N}$  such that  $\kappa_1 Q_1(q_m s) < \kappa_2 Q_2(q_m s)$  for all  $m \geq L$  and all  $s$  in a small interval  $[t, t + \delta]$  (since otherwise after some finite time server 1 would be always serving queue 1 under LEWC and hence queue 1 cannot be unstable). Next, similar to the above and since  $Q_2(t)$  is Lipschitz continuous, we can consider a subsequence of this sequence,  $\{q_n, n \geq 1\}$ , such that  $q_n^{-1} Q_2(q_n s) \rightarrow \bar{Q}_2(s)$  on a small interval  $[t, t + \epsilon]$ . Thus, in the fluid model, server 1 is serving queue 2 during an interval  $[t, t + \epsilon]$ . Therefore, on  $[t, t + \min\{h, \epsilon\}]$  both servers are serving queue 2 in the fluid model. Taking limit we have  $\frac{d}{du}[\bar{T}_{1,2}(u) + \bar{T}_{2,2}(u)]|_{u=t} = \sum_{j=1,2} \frac{r_j}{\theta_j + r_j}$ . Hence, the effective departure rate from queue  $q$  at time  $t$  is  $\bar{D}_2(t) = \mu_{11} \frac{r_1}{\theta_1 + r_1} + \mu_{22} \frac{r_2}{\theta_2 + r_2}$ . Moreover, since  $y_{12}^*$  and  $y_{22}^*$  satisfy (41)-(43), we have  $\bar{D}_2(t) \geq y_{11}^* \mu_{11} \frac{r_1}{\theta_1 + r_1} + y_{22}^* \mu_{22} \frac{r_2}{\theta_2 + r_2} \geq \lambda_2 + \tau^*$ , where  $\tau^* > 0$ . Hence, based on Theorem 2.4.9 of Dai (1999) queue 2 drains in a finite time. Therefore, from Dai and Meyn (1995) Theorem 4.1,  $L_2^\Gamma < \infty$  which is a contradiction. Thus,  $M \neq \{1, 2\}$ . Moreover, because of the symmetry in ‘‘W’’, a relabeling of the queues shows that  $M \neq \{2, 3\}$ .

The analysis for the only remaining case, i.e,  $M = \{1, 2, 3\}$  also follows a similar line of proof but by choosing appropriate subsequences such that  $\kappa_2 Q_2(q_n s) > \max\{\kappa_i Q_i(q_n s), i = 1, 3\}$ . This choice will result in the stability of queue 2 which is a contradiction.  $\square$

### **Proof of Theorem 5.**

For the case of LQ policy, set  $\kappa'_i = 1$  and notice that each server serves the queue with highest  $\kappa'_i x_i$  among its skill set. For the  $Gc\mu$  rule with quadratic holding cost define  $\kappa''_{ji} = h_i \mu_{ji}$  and notice that server  $j$  serves the queue with highest  $\kappa''_{ji} x_i$ . The proof is then similar to the proof of Theorem 4, after replacing  $\kappa_i = h_i [\sum_{j \in \mathcal{S}_i^{-1}} y_{ji}^* (\frac{r_j}{\theta_j + r_j}) \mu_{ji}]^{-1}$  of LEWC with  $\kappa'_i$  for the LQ policy and with  $\kappa''_{ji}$  for the  $Gc\mu$  rule.  $\square$

## ONLINE APPENDIX B STRUCTURE OF THE OPTIMAL POLICY

Our extensive MDP-based numerical computations show that the optimal policy for a “W” structure with server disruptions is complex in general. The following conjecture characterizes the optimal policy as a *state-dependent threshold type policy* which can be defined by four switching surfaces. We state this result as a conjecture and support it through some numerical examples.

**Conjecture** (Optimality of a State-Dependent Threshold Policy). *Consider a stabilizable “W” with stochastic disruptions (or without them as a degenerate case) and define threshold surfaces from  $\mathbb{N}^{+2}$  to the extended positive integers as:*

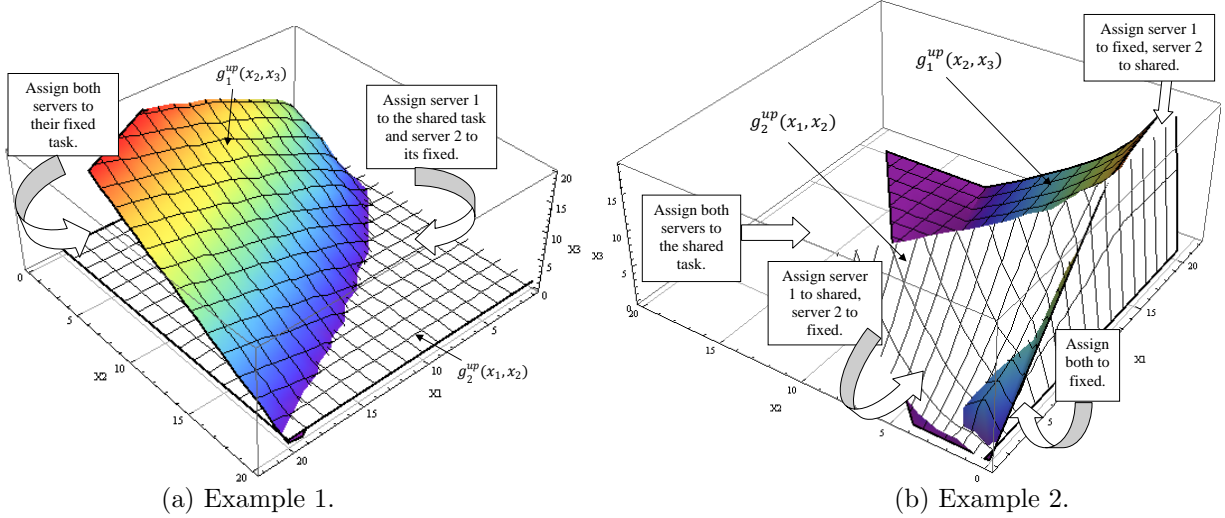
$$\begin{aligned} g_1^{up}(x_2, x_3) &= \inf \{x_1 \in \mathbb{N} : \mu_{11} \Delta_1 J(x_1 - 1, x_2, x_3, 1, 1) \geq \mu_{12} \Delta_2 J(x_1, x_2 - 1, x_3, 1, 1)\}, \\ g_1^{down}(x_2, x_3) &= \inf \{x_1 \in \mathbb{N} : \mu_{11} \Delta_1 J(x_1 - 1, x_2, x_3, 1, 0) \geq \mu_{12} \Delta_2 J(x_1, x_2 - 1, x_3, 1, 0)\}, \\ g_2^{up}(x_1, x_2) &= \inf \{x_3 \in \mathbb{N} : \mu_{23} \Delta_3 J(x_1, x_2, x_3 - 1, 1, 1) \geq \mu_{22} \Delta_2 J(x_1, x_2 - 1, x_3, 1, 1)\}, \\ g_2^{down}(x_1, x_2) &= \inf \{x_3 \in \mathbb{N} : \mu_{23} \Delta_3 J(x_1, x_2, x_3 - 1, 0, 1) \geq \mu_{22} \Delta_2 J(x_1, x_2 - 1, x_3, 0, 1)\}, \end{aligned}$$

with the convention that  $\inf \emptyset = \infty$ . Then there exists an optimal state-dependent threshold type policy that acts as follows. In each state  $\tilde{\mathbf{X}}$ , the optimal policy assigns the working server 1 (server 2) to its fixed task if, and only if,  $x_1 \geq g_1^{up}(x_2, x_3)$  ( $x_3 \geq g_2^{up}(x_1, x_2)$ ) when the other server is up, and  $x_1 \geq g_1^{down}(x_2, x_3)$  ( $x_3 \geq g_2^{down}(x_1, x_2)$ ) when the other server is down.

The above conjecture states that the four threshold/switching surfaces  $g_j^{up}(\cdot)$  and  $g_j^{down}(\cdot)$  (for  $j = 1, 2$ ) characterize an optimal policy in general. This conjecture is based on the intuition (observed numerically), that the optimal policy is monotone in the number of jobs in the fixed queues: if it is optimal to assign server 1 (server 2) to its fixed task in state  $\tilde{\mathbf{X}}$ , then it is also optimal to do so in state  $\tilde{\mathbf{X}} + \mathbf{e}_1$  ( $\tilde{\mathbf{X}} + \mathbf{e}_3$ ). Therefore, for every server, it is sufficient to recognize the minimum number of jobs in that server’s fixed task such that it is optimal to start serving the fixed task. Such states are defined by the switching surfaces  $g_j^{up}(\cdot)$  and  $g_j^{down}(\cdot)$  (for  $j = 1, 2$ ).

**Remark.** Theorem 3 presented a special case of the above conjecture by setting all the surface functions to the value 1. Another special case can be seen in Down and Lewis (2010) for the so-called “N” structure without disruptions, where the authors show the optimality of a threshold type policy for the case when all servers have an equal service rate (see Theorem 5.4 part 4 of Down and Lewis (2010)).

The following numerical examples support the above conjecture and clarify the behavior of threshold surfaces by illustrating the optimal policy.

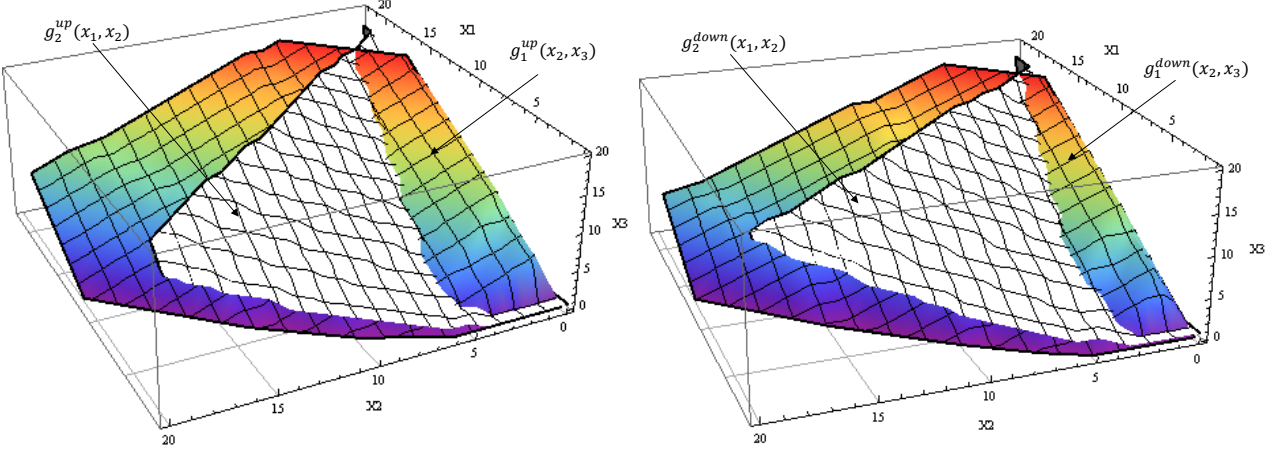


**Figure 7:** Illustration of the optimal policy in examples 1 and 2.

**Example 1.** Consider a “W” queueing network where servers are reliable (i.e., a system without disruptions) with parameters  $\lambda'_1 = 0.7$ ,  $\lambda'_2 = 0.9$ ,  $\lambda'_3 = 0.4$ ,  $\mu'_{11} = \mu'_{12} = 1.0$ ,  $\mu'_{22} = 1.2$ ,  $\mu'_{23} = 1.5$ ,  $h'_1 = 1.0$ ,  $h'_2 = 1.2$ , and  $h'_3 = 1.0$ . Fig. 7(a) illustrates the optimal policy. Since there is no disruption, the optimal policy can be described by threshold surfaces  $g_1^{up}(x_2, x_3)$  and  $g_2^{up}(x_1, x_2)$ . As Fig. 7(a) shows, server 2 follows the  $c\mu$  rule by giving strict priority to its fixed task. Thus,  $g_2^{up}(x_1, x_2) = 1$  for every  $x_1, x_2 \in \mathbb{Z}^+$ . However, for server 1, serving the fixed is optimal if, and only if,  $x_1 \geq g_1^{up}(x_2, x_3)$ . It should be noted that there are states where servers serve the shared task to avoid idling. However, these states are hidden behind the surface  $g_2^{up}(x_1, x_2) = 1$  in Fig. 7. Since one of the servers is following a strict priority rule, the optimal policy is still simple and intuitive. However, as the next examples show, the optimal policy can be much more complex.

**Example 2.** Consider a “W” queueing network where servers are reliable (i.e., a system without disruptions) with parameters  $\lambda'_1 = 0.7$ ,  $\lambda'_2 = 0.9$ ,  $\lambda'_3 = 0.4$ ,  $\mu'_{11} = 1.2$ ,  $\mu'_{12} = 1.0$ ,  $\mu'_{22} = 1.0$ ,  $\mu'_{23} = 1.2$ ,  $h'_1 = 1.0$ ,  $h'_2 = 1.5$ , and  $h'_3 = 1.0$ . Fig. 7(b) illustrates the optimal policy. As can be seen in Fig. 7(b), the optimal policy is much more complicated in this case, since both servers use threshold surfaces. These two threshold surfaces divide the state space into four regions. As is shown in Fig. 7(b) (using wide arrows), in each region a specific allocation remains optimal. As with all other numerical examples we performed, the observed behavior is consistent with the conjecture made earlier. The next example presents a case where the servers are subject to stochastic disruptions and again adheres to the earlier conjecture.

**Example 3.** Consider a “W” structure where servers are not reliable and the parameters of the system are:  $\lambda'_1 = 0.3$ ,  $\lambda'_2 = 0.2$ ,  $\lambda'_3 = 0.1$ ,  $\mu'_{11} = 1.0$ ,  $\mu'_{12} = 1.2$ ,  $\mu'_{22} = 1.2$ ,  $\mu'_{23} = 1.0$ ,  $h'_1 = 1.1$ ,



**Figure 8:** Illustration of the optimal policy in example 3.

$h'_2 = 1.2$ ,  $h'_3 = 1.3$ ,  $\theta'_1 = 0.4$ ,  $\theta'_2 = 0.45$ ,  $r'_1 = 0.6$ , and  $r'_2 = 0.55$ . Fig. 8 illustrates the optimal policy. In Fig. 8, the graph on left depicts  $g_1^{up}(x_2, x_3)$  and  $g_2^{up}(x_1, x_2)$ . The graph on right illustrates  $g_1^{down}(x_2, x_3)$  and  $g_2^{down}(x_1, x_2)$ . Hence, the left graph shows the optimal policy when both servers are available. The right graph, depicts the optimal behavior of one server when the other server is disrupted. The four threshold surfaces depicted in Fig. 8 completely define the optimal policy, and the result is consistent with the earlier conjecture. A first glance at Fig. 8 might indicate that the threshold surfaces are the same regardless of the other server's availability (i.e.,  $g_j^{up}(\cdot) = g_j^{down}(\cdot)$ ). However, a closer look reveals that the threshold surfaces  $g_j^{up}(\cdot)$  and  $g_j^{down}(\cdot)$  (for both  $j = 1, 2$ ) are different at many points. For instance, near the origin,  $g_1^{down}(\cdot)$  is lower than  $g_1^{up}(\cdot)$  (e.g.,  $(2,1,6)$  is on  $g_1^{up}(\cdot)$ , but  $(2,1,1)$  is on  $g_1^{down}(\cdot)$ ). In this example, to start serving its fixed task, server 1 requires fewer jobs at queue 3 when server 2 is down than when it is working.

## ONLINE APPENDIX C

### TEST SUITES FOR NUMERICAL COMPARISONS

This appendix presents the test suites for our numerical studies. Part I describes the test suite for comparing the “W” design with other possible designs and is related to Figures 3 and 4. Part II presents the test suite for comparing the performance of our proposed Largest Expected Workload Cost (LEWC) heuristic with other well-known control policies (Section 4.4).

#### PART I: COMPARISONS OF VARIOUS STRUCTURES (Figures 3 and 4)

The parameter settings used in Fig. 4 are presented in Table 1.C. As is the case throughout the paper, we distinguish between non-uniformized parameters and the uniformized parameters using prime notation. In this table,  $\rho$  denotes the congestion factor of the system, which varies from 0.1 to 0.9 as is illustrated in Fig. 4. In Table 1.C, as is introduced in the main body,  $\mathcal{S}_j$  represents the skill set or capability set of server  $j$ . For instance, in Suite 1, server 1 works at a speed of 3 and server 2 works at a speed of 2. To eliminate the homomorphic designs in these test suites, we assume job type two (the shared task) has an arrival rate greater than or equal to the fixed types. Moreover, based on Theorem 1, for large values of  $\rho$  in some test suites, some structures with low flexibility can be unstable. This fact is considered in the illustration presented in Fig. 4. For instance, as is clear from Fig. 4, Structure 1 is not stabilizable for large values of  $\rho$  in Suites 1, 2, and 4.

**Table 1.C.** Various Suites of Parameter Settings for Comparisons of Alternate Structures.

Parameters	Suite 1	Suite 2	Suite 3	Suite 4
$\lambda'_1$	$4/3 \rho$	$0.25 \rho$	$1 \rho$	$4/3 \rho$
$\lambda'_2$	$4/3 \rho$	$3.50 \rho$	$2 \rho$	$4/3 \rho$
$\lambda'_3$	$4/3 \rho$	$0.25 \rho$	$1 \rho$	$4/3 \rho$
$\mu'_{1i} (\forall i \in \mathcal{S}_1)$	3.0	2.0	2.0	2.0
$\mu'_{2i} (\forall i \in \mathcal{S}_2)$	2.0	2.0	2.0	2.0
$\theta'_1$	0.1	0	0	0
$\theta'_2$	0.1	0	0	0
$r'_1$	0.9	0	0	0
$r'_2$	0.9	0	0	0

## PART II: COMPARISONS OF LEWC WITH OTHER WELL-KNOWN POLICIES

Assume that every source (worker, server, machine, agent, etc.) is at least as good as a standard source (which serves in expectation one unit of work content in one unit of time). We consider different skill levels (heterogeneity in service rates) and allow some sources to be 20% faster in working on some job types. Table 2.C presents various combinations of interest assuming that servers are faster (in a non-strict sense) in serving their specialized (i.e., fixed) task. (Notice that otherwise, there would be another design for which the current shared task is fixed for that rapid server, and the current fixed task of that rapid server is shared between both servers.) Moreover, for every service rate setting, we consider four workload distributions as shown in Table 3.C. These workload distributions introduce asymmetry in the amount of the time that each server needs to spend among the queues. Setting (a) in this table represents a case where both servers need to spend their time equally between their fixed task and the shared task. This case represents a symmetric workload distribution among the queues. Settings (b)-(d) consider other extreme workload distributions where a server may need to spend 80% of its time on either the fixed task or the shared task. Notice that (1) because of the inherent symmetry in the “W”, these settings cover all key workload distributions with the above assumption, and (2) these workload distributions are approximate and the actual workloads of servers depend on the control policy implemented.

Considering these workload distributions, Table 2.C also presents the corresponding arrival rates as a function of a congestion factor,  $\rho$ . Assuming that both servers are 100% available, a congestion factor of 1 indicates the maximum possible arrival rates to each queue (which corresponds to a system with 100% utilization) based on the corresponding workload distribution. Hence, by computing the arrival rates in this way: (1) we can ensure that the problem instances are within the feasible/stable region, and (2) by changing  $\rho$ , we can control the overall utilization (or traffic) of the system. As shown in Table 3.C, we choose  $\rho$  such that for every setting we can include a system in both 90% (relatively high) and 70% (relatively low) utilization. Notice that the actual system utilization also depends on the control policy implemented. Since there is no analytical method to exactly determine the real utilization of the system, we have reported the approximate system utilization based on the ratio of work content over available capacity as is presented in Table 4.C.

Table 4.C also presents disruption and repair rates corresponding to combinations of 100% (representing no-disruption scenarios), 90% (representing relatively high availability), and 70% (representing relatively low availability) of servers. Specifically, Settings (I) and (II) in this table illustrate systems with completely reliable servers under relatively high and low traffic, respectively. Settings (III) and (IV) represent systems in which one server is of low reliability and the other server is highly reliable. Since disruption and repair dynamics form a two-state Markov chain, we can precisely compute the steady-state availabilities. In fact, the ratio  $r_j/(r_j + \theta_j)$  is the steady-state availability of server  $j$  and is reported in Table 4.C. Moreover, we assume that both servers are repaired through a similar process. Hence, the repair rates are selected to be close. However, by considering different repair rates, we also include cases in which servers have asymmetric reliability. Because of the symmetry in the “W,” it is sufficient to consider server 1 as the server that possesses a lower reliability. To capture the effect of disruptions (for scenarios that allow disruptions), repair rates are also selected such that when a server is disrupted, it takes approximately 3 (precisely  $1/0.35$ ) standard service durations until the server returns to a working state.

Finally, Table 5.C presents different holding cost settings. In Setting (A), all job types have a similar holding cost, representing a symmetric case. Settings (B)-(D) introduce asymmetry in holding costs. More precisely, in Settings (B), (C), and (D) the asymmetry among the costs is obtained by using multipliers of 0.25, 0.5, and 1, respectively. Hence, the degree of asymmetry in holding costs (i.e., differences in costs) increases from Setting (A) to (D). Moreover, costs within each of these settings are chosen such that they cover all the possible  $c\mu$  order-based permutations (among the three queues). In fact, since it is sufficient to consider only the highest and the second highest  $c\mu$  rankings (as the third one will follow), the last column of Table 5.C reports the corresponding  $c\mu$  ranking permutation among the queues. Because of the symmetry between fixed queues, we can assume the holding cost associated with one of the fixed queues ( $h_1$  or  $h_3$ ) is always greater than the holding cost associated with the other fixed queue. We assume ( $h_3 \geq h_1$ ) to eliminate the degenerate cases. These cost settings together with service rate settings (Table 2.C) and disruption settings (Table 4.C) generate  $10 \times 12 \times 4 = 480$  different problem instances that cover a fair and extensive range of possibilities regarding arrivals, holding costs, disruptions, traffic, service rates, etc.

**Table 2.C.** Various service rate combinations and arrival rates in the test suites.

Setting	Service Rates				Workload Distribution Setting	Arrivals		
	$\mu'_{11}$	$\mu'_{12}$	$\mu'_{22}$	$\mu'_{23}$		$\lambda'_1$	$\lambda'_2$	$\lambda'_3$
1	1.0	1.0	1.0	1.0	(a)	$0.5\rho$	$1\rho$	$0.5\rho$
2	1.0	1.0	1.0	1.0	(b)	$0.8\rho$	$1\rho$	$0.2\rho$
3	1.0	1.0	1.0	1.0	(c)	$0.8\rho$	$0.4\rho$	$0.8\rho$
4	1.0	1.0	1.0	1.0	(d)	$0.2\rho$	$1.6\rho$	$0.2\rho$
5	1.0	1.0	1.0	1.2	(a)	$0.5\rho$	$1\rho$	$0.6\rho$
6	1.0	1.0	1.0	1.2	(b)	$0.8\rho$	$1\rho$	$0.24\rho$
7	1.0	1.0	1.0	1.2	(c)	$0.8\rho$	$0.4\rho$	$0.96\rho$
8	1.0	1.0	1.0	1.2	(d)	$0.2\rho$	$1.6\rho$	$0.24\rho$
9	1.2	1.0	1.0	1.2	(a)	$0.6\rho$	$1\rho$	$0.6\rho$
10	1.2	1.0	1.0	1.2	(b)	$0.96\rho$	$1\rho$	$0.24\rho$
11	1.2	1.0	1.0	1.2	(c)	$0.96\rho$	$0.4\rho$	$0.96\rho$
12	1.2	1.0	1.0	1.2	(d)	$0.24\rho$	$1.6\rho$	$0.24\rho$

**Table 3.C.** Workload distributions on skills used in Table 2.C (S: Server; C: Class of Customer).

Setting	Workload Distributions			
	S1-C1	S1-C2	S2-C2	S2-C3
(a)	50%	50%	50%	50%
(b)	80%	20%	80%	20%
(c)	80%	20%	20%	80%
(d)	20%	80%	80%	20%

Tables 6.C, 7.C, and 8.C present the test suite used for investigating the effect of disruptions (Fig. 6). Table 6.C corresponds to Table 2.C with a congestion factor of  $\rho = 0.5$ . Table 7.C presents the various holding cost settings considered, and Table 8.C presents the settings related to disruptions and repair. Server 1 is considered to be completely reliable. However, disruption rate of Server 2 changes from 0.0 to 0.5, while its repair rate is kept constant to generate different steady-state realisability levels.

**Table 4.C.** Settings of disruption rates, repair rates, and system congestion factor ( $\rho$ ) in addition to the corresponding system utilization

Setting	Disruption&Repair Rates				Availability		Congestion Factor ( $\rho$ )	Sys. Utilization
	$r_1$	$r_2$	$\theta_1$	$\theta_2$	Server 1	Server 2		
(I)	0	0	0	0	100%	100%	0.9	$\sim 90\%$
(II)	0	0	0	0			0.7	$\sim 70\%$
(III)	0.35	0.36	0.15	0.04	70%	90%	0.63	$< 90\%$
(IV)	0.35	0.36	0.15	0.04			0.49	$< 70\%$

**Table 5.C.** Holding cost settings and the corresponding highest and second highest  $c\mu$  rankings among the queues (SF: Highest=Shared, Second Highest=Fixed; FS: Highest=Fixed, Second Highest=Shared, FF: First and Second Highest=Fixed).

Setting	Cost Asymmetry	Holding Cost			$c\mu$ Ranking Permutation	No. of Cases
		$h_1$	$h_2$	$h_3$		
(A)	Zero	1	1	1	Symmetric	48
(B)	Low	1	1.5	1.25	SF	48
		1	1.25	1.5	FS	48
		1.25	1	1.5	FF	48
(C)	Moderate	1	2	1.5	SF	48
		1	1.5	2	FS	48
		1.5	1	2	FF	48
(D)	High	1	3	2	SF	48
		1	2	3	FS	48
		2	1	3	FF	48

**Table 6.C.** Service and arrival rates considered for the results on the effect of server disruption risk (Fig. 6).

Setting	Service Rates				Arrival Rates		
	$\mu_{11}$	$\mu_{12}$	$\mu_{22}$	$\mu_{23}$	$\lambda_1$	$\lambda_2$	$\lambda_3$
1	1.0	1.0	1.0	1.0	0.25	0.50	0.25
2	1.0	1.0	1.0	1.0	0.40	0.50	0.10
3	1.0	1.0	1.0	1.0	0.40	0.20	0.40
4	1.0	1.0	1.0	1.0	0.10	0.80	0.10
5	1.0	1.0	1.0	1.2	0.25	0.50	0.30
6	1.0	1.0	1.0	1.2	0.40	0.50	0.12
7	1.0	1.0	1.0	1.2	0.40	0.20	0.48
8	1.0	1.0	1.0	1.2	0.10	0.80	0.12
9	1.2	1.0	1.0	1.2	0.30	0.50	0.30
10	1.2	1.0	1.0	1.2	0.48	0.50	0.12
11	1.2	1.0	1.0	1.2	0.48	0.20	0.48
12	1.2	1.0	1.0	1.2	0.12	0.80	0.12

**Table 7.C.** Holding cost setting considered for the results on the effect of server disruption risk (Fig. 6).

Setting	Holding Cost		
	$h_1$	$h_2$	$h_3$
1	1	1	1
2	1	3	2
3	1	2	3
4	2	1	3

**Table 8.C.** Disruption and repair rate setting considered for the results on the effect of server disruption risk (Fig. 6).

Setting	Disruption/Repair Rates			
	$r_1$	$r_2$	$\theta_1$	$\theta_2$
1	0	0.5	0	0
2	0	0.5	0	0.1
3	0	0.5	0	0.2
4	0	0.5	0	0.3
5	0	0.5	0	0.4
6	0	0.5	0	0.5