Design Principles for Flexible Systems

Abstract

A fundamental aspect of designing systems with dedicated servers is identifying and improving the system bottlenecks. We extend the concept of a bottleneck to networks with heterogeneous, flexible servers. In contrast with a network with dedicated servers, the bottlenecks are not a priori obvious, but can be determined by solving a number of linear programming problems. Unlike the dedicated server case, we find that a bottleneck may span several nodes in the network. We then identify some characteristics of desirable flexibility structures. In particular, the chosen flexibility structure should not only achieve the maximal possible capacity (corresponding to full server flexibility), but should also have the feature that the entire network is the (unique) system bottleneck. The reason is that it is then possible to shift capacity between arbitrary nodes in the network, allowing the network to cope with demand fluctuations. Finally, we specify when certain flexibility structures (in particular chaining, targeted flexibility, and the "N" and "W" structures from the call center literature) possess these desirable characteristics.

1 Introduction

In this paper, we are concerned with the problem of deciding how to cross-train a collection of servers to perform a set of tasks. Absent additional constraints, we would like to identify fundamental properties of the set of skills that each server should have. We are in general interested in which sets of skills are needed to approach the same throughput performance as that of *full flexibility* (i.e., all servers are trained for all tasks) while also being adaptable to changes in the environment, manifested by perturbations in arrival and/or service rates. We would like to do this in a very general setting, so that the system topology is general, there can be many different demand types, and the servers are heterogeneous in their capabilities.

Our goal is to allow a designer to quickly evaluate candidate flexibility structures, leading to a smaller number of desirable structures that may then be further examined using more detailed analysis (simulation or other techniques, see the concluding remarks of this paper for more on the latter).

We perform our studies using queueing network models (our insights apply to more general systems). To illustrate our approach, consider the following queueing model. We have K tasks in parallel, with customers for each task arriving at rate λ_k. There are M flexible servers, where server j is capable of working (if trained) on customers from task k at rate $\mu_{j,k}$. (The main results in the paper are for more general network topologies.)

Consider a system with full flexibility in that each server is trained to work on all tasks. This flexibility structure is depicted in Figure 1, where an arrow from a server to a task indicates that the server has been trained for that task. This flexibility structure is desirable in that it is both *efficient* (it maximizes capacity) and *robust* (it can deal with fluctuations in demand and processing rates via server reassignment). However, it is *costly* in that it involves the maximal possible server training (each server is trained to work on all tasks). This raises the question of whether it is possible to achieve the *benefits of full flexibility* in a less costly manner (i.e., with fewer server skills).

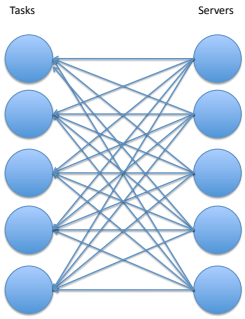

Figure 1: Full flexibility

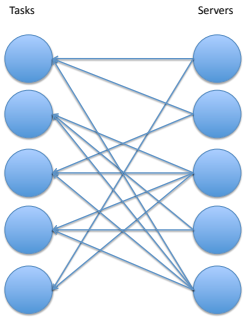

Figure 2: Candidate flexibility structure

Suppose now that we are presented with the flexibility structure in Figure 2. We would like to identify guidelines as to whether this is a good flexibility structure or not. Here, what we mean by good is that the system is both efficient and robust, and hence achieves the benefits of full flexibility (in a first order sense). We will show that efficiency is easily checked by solving two linear programs (LPs) and that robustness can be verified by solving at most another K LPs.

An important notion in our work is that of a system bottleneck, which generalizes the notion of a bottleneck for a system with dedicated servers. Just as in the dedicated server case, we find that one can make initial design decisions by simply doing a bottleneck analysis. However, when the servers are flexible, then their time can be divided between stations in the network, and the overall load at each station or set of stations depends on how much time the servers capable of working at those stations in fact spend working there (rather than simply on the number of servers allocated to each station and their respective service rates). Consequently, it is no longer sufficient to consider only individual stations when determining what "bottleneck" bounds the capacity of the system. In other words, the bottleneck set may be a set of stations, rather than a single station.

We show that in addition to achieving the maximum possible capacity, it is desirable for the unique bottleneck to consist of all of the stations in the network. When this is true, the network is robust with respect to the assumptions that it is derived under. In particular, if demands increase or service rates decrease, capacity from within the system can be shifted to compensate for these changes. We will refer to flexibility structures possessing these two properties as "capacity effective," while recognizing that a manager may also be interested in other objectives or constrained by other considerations, a point that is discussed further in Section 4.

A key notion that has appeared in the literature is that of *chaining* (see Figure 3 for an example), introduced in the seminal paper of Jordan and Graves [21] and further developed by, amongst others, Bassamboo et al. [7], Graves and Tomlin [11], Gurumurthi and Benjaafar [12], Hopp et al. [17], Iravani et al. [20], and Sheikzadeh et al. [24]. Our results reinforce the insight in [21] (and further developed in Chou et al. [9]) that chaining is desirable in homogeneous environments, but also indicate that other flexibility structures often show better performance in heterogeneous settings. A similar observation was made by Gurumurthi and Benjaafar [12] who present numerical results showing that other flexibility structures could be better than chaining (with throughput as the performance measure of interest). Using simulation studies for a specific system where workers sharing the same role are identical, Jordan et al. [22] discuss the robustness of chaining to errors in estimating system parameters (see Section 3.3 in particular), which is in the spirit of our notion of effectiveness.

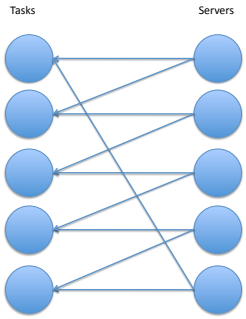

Figure 3: 2-chain

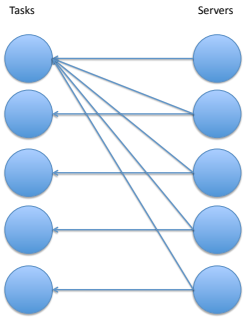

Figure 4: Single task

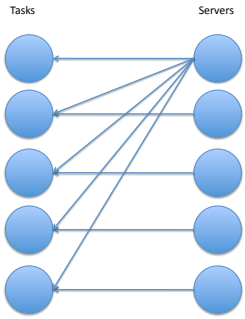

Figure 5: Single server

Our general results are used to identify when various flexibility structures are capacity effective. For example, while prior works provide insights about when chaining is desirable, we use the notion of a system bottleneck to identify *explicit conditions* guaranteeing that chaining is capacity effective. This is done for systems of arbitrary size when there is structure

in the service rates (the servers are generalists in that their service rates are products of two terms, one dependent only on the server and one dependent only on the task). For systems with three servers and three tasks, we identify a sufficient condition for when chaining is capacity effective for arbitrary service rates. While the resulting conditions are somewhat complicated, they do demonstrate how our approach yields explicit conditions. We then proceed to specify when targeted flexibility, in which all servers are trained for one task (see Figure 4) or one server is trained for all tasks (see Figure 5), is capacity effective. As for chaining, this is done for systems of arbitrary size with generalist servers and for systems with three tasks and three servers for arbitrary service rates. We also provide explicit results for the "N" and "W" flexibility structures that have arisen in the call center literature.

The organization of this paper is as follows. Section 2 gives details of the queueing model under study. Section 3 demonstrates how to locate the system bottleneck and discusses the connections between determining the bottleneck and stability properties of the queueing system. Section 4 then discusses how the bottleneck can be used to characterize capacity effective flexibility structures. Section 5 examines when specific flexibility structures (including chains) are capacity effective. Section 6 provides concluding remarks. Finally, a discussion of how our work relates to the literature on complete resource pooling is contained in the Appendix. Proofs of most of our results can be found in an online companion.

2 Queueing Model

We consider a system with N mutually independent renewal arrival (demand) streams, with rates $\lambda_i > 0$, $i = 1, \ldots, N$. Arrivals from stream i will be called type i customers. There are $K \geq N$ queues in the system. A (possibly random) sequence of tasks must be performed on each arriving customer. We will call customers stored in queue k task k customers, and we

define $i(k)$ to be the corresponding type of task k customers. The set of tasks $\mathcal{K}_i$ contains all tasks with type i customers, i.e., $\mathcal{K}_i = \{k : i(k) = i\}$. We assume that $\mathcal{K}_i \bigcap \mathcal{K}_j = \emptyset$ when $i \neq j$ and that $\bigcup_i \mathcal{K}_i = \{1, \ldots, K\}$. This can be thought of as separating different customer types into different queues (so that $i(k)$ yields a unique type), allowing the types to be treated separately, if desired. Type i arrivals from outside are routed to task $k \in \mathcal{K}_i$ with probability $p_{0,k}$. Upon completion of task k, a customer becomes a task k' customer with probability $p_{k,k'}$ and leaves the system with probability $1 - \sum_{k' \in \mathcal{K}_{i(k)}} p_{k,k'}$. In a parallel system of queues, $K = N$, $\mathcal{K}_i = \{i\}$ for all i, and $p_{0,k} = 1$ for all k. As another example, suppose that type 1 customers first have task 1 performed, followed by either task 2 or task 3 with equal probability, after which they leave the network. Type 2 arrivals first have task 4 performed, followed by task 5, after which they leave the network. Here, $\mathcal{K}_1 = \{1, 2, 3\}$, $\mathcal{K}_2 = \{4, 5\}$, $p_{0,1} = p_{0,4} = 1$, the internal routing probabilities are $p_{1,2} = p_{1,3} = 1/2$, $p_{4,5} = 1$, and all other values $p_{k,k'}$ are zero.

There are M servers. A server j has a *potential* service rate of $\mu_{j,k}$ for task k customers. Thus the server may or may not be trained to work on task k customers, and if the server is trained to work on task k, the service times form an independent and identically distributed (i.i.d.) sequence with rate $\mu_{j,k}$. Servers can either work in parallel at a task, or work together on a customer, in which case their service rates are additive. Without loss of generality, we assume that $\sum_{k=1}^{K} \mu_{j,k} > 0$ for each server $j = 1, \ldots, M$. Let $f_{j,k} = 1$ if server j is trained for task k and 0 otherwise. By varying the number and location of the ones in the set $\{f_{j,k}\}$, we can examine different flexibility structures. It is worthwhile to note here that we can extend our results to unreliable servers. This will be discussed briefly near the end of Section 4.

3 Determining the Bottleneck

Let a_k be the expected number of visits by a type $i(k)$ customer to task k. To determine a_k, $k = 1, \ldots, K$, we need to solve the following set of traffic equations for each customer type i and all tasks $k \in \mathcal{K}_i$ visited by that customer type:

$$a_k = p_{0,k} + \sum_{k' \in \mathcal{K}_i} a_{k'} p_{k',k}.$$

We assume that $(I - P')^{-1}$ exists, where P is a K by K matrix with (i, j) entry $p_{i,j}$ and $'$ denotes transpose. This is equivalent to assuming that all arrivals eventually leave the network, and, in particular, that a_k is finite for $k = 1, \ldots, K$.

Because there are multiple arrival streams, the notion of capacity is complicated, due to the tradeoff over how much capacity to give to each demand type. We approach this by measuring capacity with respect to how much a particular set of arrival rates for the K demand types can be inflated (or needs to be deflated) in order to ensure the stability of the system. In other words, we are maximizing the total capacity under the constraint that the fraction of the total capacity that is given to each demand type remains unchanged. This can be viewed as treating the demand types "fairly," and is certainly appropriate when there is just one demand type. Keeping the relative demands fixed is useful because otherwise we would have to look at a polytope of achievable demands, rather than a single number.

To accomplish this, we consider the following allocation LP, where $\Gamma \subseteq \{1, \ldots, K\}$ is a subset of the set of all tasks. The decision variables are $\{\delta_{j,k}\}$ and γ, and we will denote the

optimal value of the LP by $\gamma^*(\Gamma)$. We maximize γ subject to

$$\sum_{j=1}^{M} \delta_{j,k}\mu_{j,k}f_{j,k} \geq \gamma a_k \lambda_{i(k)}, \qquad k \in \Gamma; \tag{1}$$

$$\sum_{k=1}^{K} \delta_{j,k} \leq 1, \qquad j = 1, \ldots, M; \tag{2}$$

$$\delta_{j,k} \geq 0, \qquad j = 1, \ldots, M,\ k = 1, \ldots, K. \tag{3}$$

The above LP determines the optimal assignments $\{\delta^*_{j,k}\}$ of servers to the tasks in Γ if we increase the demands to the point of instability while keeping the relative demands fixed. The constraint (1) guarantees that the service capacity allocated to task k is at least the arrival rate. Constraint (2) prevents overallocation of a server, while (3) prevents negative server allocations.

We call any set Γ satisfying $\gamma^*(\Gamma) = \gamma^*(\{1, \ldots, K\})$ a *bottleneck set*. This is a natural definition, as the bottleneck sets determine the maximum load, which is consistent with the conventional use of the term "bottleneck." If $\{1, \ldots, K\}$ is the unique bottleneck set, we will say that the *entire system is the unique bottleneck*. Note that in the traditional (dedicated server) setting, servers are uniquely identified with tasks, so the set of bottleneck servers is identical to the set of bottleneck tasks. In our flexible server setting, there is similarly a correspondence between the sets of bottleneck servers and tasks, but this connection is less immediate. In particular, the set of bottleneck servers is composed of all servers capable of serving a task in minimal sets of bottleneck tasks (these servers are highly utilized as any such server has constraint (2) tight). In the traditional setting, the performance of the system is improved by enhancing the capacity of the bottlenecks. In our setting, as the potential service rates $\{\mu_{j,k}\}$ of the servers are given, we will not be enhancing the capacity of highly utilized servers, but will instead enhance the capacity of bottleneck tasks by training under-utilized servers to work there. This explains why the system bottlenecks are more

naturally defined as sets of tasks, rather than sets of servers, in our setting.

We will use the allocation LP above to identify capacity effective flexibility structures. However, we first make precise the notion that $\gamma^*(\{1,\ldots,K\})$ is a measure of the stability of the network. Let $Q_k(t) \geq 0$ be the queue length at task k at time t (including customers in process, if any) and let $Q(t)$ be a vector whose kth entry is $Q_k(t)$. The norm $|Q(t)|$ is defined as $\sum_{k=1}^{K} Q_k(t)$. The following result has been shown for $N = 1$, see Theorem 1 of Andradóttir et al. [4], and here it is extended to an arbitrary number of demand types. The proof is in an online companion.

Proposition 1 *(i) For any set of arrival processes with rates $\gamma\lambda_i$, $i = 1,\ldots,N$, where $\gamma < \gamma^*(\{1,\ldots,K\})$, there exists a server scheduling policy such that the distribution of the queue length process $\{Q(t)\}$ converges to a steady-state distribution φ as $t \to \infty$.*
(ii) For any set of arrival processes with rates $\gamma\lambda_i$, $i = 1,\ldots,N$, where $\gamma > \gamma^*(\{1,\ldots,K\})$, $P(|Q(t)| \to \infty) = 1$ for any server scheduling policy.

An immediate corollary indicates whether the system with the given arrival processes can be stabilized.

Corollary 1 *(i) If $\gamma^*(\{1,\ldots,K\}) > 1$, then for the set of arrival rates $\{\lambda_i\}$, a server scheduling policy exists such that the queue length process $\{Q(t)\}$ converges to a steady-state distribution φ as $t \to \infty$.*
(ii) If $\gamma^*(\{1,\ldots,K\}) < 1$, then for the set of arrival rates $\{\lambda_i\}$, $P(|Q(t)| \to \infty) = 1$ for any server scheduling policy.

4 Capacity Effective Flexibility Structures

Let $\bar{\gamma}$ be the largest possible value of $\gamma^*(\{1,\ldots,K\})$, which occurs under full flexibility when $f_{j,k} = 1$ for $j = 1,\ldots,M$, $k = 1,\ldots,K$. We are interested in identifying fundamentally sound structural properties for a given set $\{f_{j,k}\}$ that are explicitly verifiable given the system's parameters. In particular, we would like to choose $\{f_{j,k}\}$ such that

$$\gamma^*(\{1,\ldots,K\}) = \bar{\gamma} \tag{4}$$

and

$$\gamma^*(\Gamma) > \gamma^*(\{1,\ldots,K\}) \text{ for all strict subsets } \Gamma \text{ of } \{1,\ldots,K\}. \tag{5}$$

As discussed in the Introduction, the first property (4) indicates that we would like the flexibility structure to be *efficient*, i.e., it can handle the same load on the queues as full flexibility. The physical interpretation of the entire set of tasks being the unique bottleneck (5) is that it is possible to shift excess capacity from any task to any other task, which should aid in alleviating both long-term and short-term workload imbalances. Thus the flexibility structure is *robust* with respect to the assumptions it is derived under. This is discussed in more detail below. Properties (4) and (5) are likely to be appealing to a manager in that (4) ensures optimal performance in the present environment, while (5) establishes the ability to rapidly adapt to changes, and is hence aimed at future performance. Also, note that (5) is satisfied under full flexibility when $\mu_{j,k} > 0$ for all j,k, so flexibility structures satisfying (4) and (5) can be said to offer the benefits of full flexibility (in a first order sense). We call such structures *capacity effective*.

Note that if (4) and (5) are satisfied but the wrong scheduling policy is chosen, an artificial bottleneck may develop at demand rates much less than optimal. Fortunately, avoiding this situation is always possible, see Proposition 1. Thus we can decouple the

question of how to design a flexible system in terms of choosing skills for the available servers from the question of how to identify a scheduling policy to ensure a given level of throughput performance. From this point on, we will be concerned only with the issue of choosing a flexibility (skill) structure. Using numerical results, Gurumurthi and Benjaafar [12] also show that performance can be policy dependent for Markovian systems (optimal policies are not identified).

In general, the conditions (4) and (5) must both be checked. Consider a system with $M = 3$ servers and $N = K = 3$ tasks in parallel, with arrival rates $\lambda_1 = \lambda_2 = \lambda_3 = \lambda$. Let $\mu_{1,1} = \mu_{1,2} = \mu_{2,2} = \mu_{2,3} = \mu_{3,1} = \mu_{3,3} = \mu$ and $\mu_{1,3} = \mu_{2,1} = \mu_{3,2} = 2\mu$. Then, the (chaining) flexibility structure $f_{1,1} = f_{1,2} = f_{2,2} = f_{2,3} = f_{3,1} = f_{3,3} = 1$ and $f_{j,k} = 0$ otherwise satisfies (5), but $\gamma^*(\{1, \ldots, K\}) = \mu/\lambda < \bar{\gamma} = 2\mu/\lambda$, so (4) is not satisfied. On the other hand, the (dedicated) flexibility structure $f_{1,3} = f_{2,1} = f_{3,2} = 1$ and $f_{j,k} = 0$ otherwise satisfies (4), but not (5) ($\gamma^*(\{k\}) = \gamma^*(\{1, 2, 3\})$ for $k = 1, 2, 3$).

On the computation side, we do not need to evaluate (5) for all Γ, as if $\Gamma \subseteq \Gamma'$, $\gamma^*(\Gamma) \geq \gamma^*(\Gamma')$ (see (1)). This implies that we need only check (5) for at most all subsets of Γ consisting of $K - 1$ tasks. Thus, in order to check the conditions, we need to solve at most $K + 2$ LPs (one for the given flexibility structure, one for full flexibility, and K with one task removed).

To make the notion of being able to shift capacity more precise, we see that (5) implies the following more formal result. The first part of Theorem 1 states that if the entire system is the unique bottleneck, and there is a change in the underlying environment such that a demand λ_i decreases, then the system can accommodate increased demand for *all* other customer types $i' \neq i$. Flipping this around, if there is a change in the underlying environment such that a demand increases, *any* other demand may be decreased to compensate (at least

in part). The second part shows that we cannot shift capacity into a bottleneck set from outside. For example, this implies that increases in demand within the bottleneck set can only be compensated for by decreases in other demands within the bottleneck set.

Theorem 1 *(i) Suppose that (5) holds. For any $i \in \{1, \ldots, N\}$, if λ_i is decreased, then $\gamma^*(\{1, \ldots, K\})$ is increased.*

(ii) Suppose that (5) does not hold for some strict subset Γ of $\{1, \ldots, K\}$. Then, for all i such that $\mathcal{K}_i \cap \Gamma = \emptyset$, if we decrease λ_i by any amount, *$\gamma^*(\{1, \ldots, K\})$ remains unchanged.*

Proof. Let $\{\delta^*_{j,k}\}$ be the solution for the allocation LP (1)-(3) under the original arrival rates. If we decrease λ_i, then for any task $k \in \mathcal{K}_i$, (1) is not tight.

From (5), $\gamma^*(\{1, \ldots, K\} \setminus \mathcal{K}_i) > \gamma^*(\{1, \ldots, K\})$. This implies that there exists a server j_1 and tasks $k \in \mathcal{K}_i$, $k_1 \notin \mathcal{K}_i$ satisfying $\delta^*_{j_1,k}\mu_{j_1,k}f_{j_1,k} > 0$ and $\mu_{j_1,k_1}f_{j_1,k_1} > 0$. Hence, we can decrease $\delta^*_{j_1,k}$ and increase $\delta^*_{j_1,k_1}$, without reducing $\gamma^*(\{1, \ldots, K\})$, so that (1) is not tight for any $k' \in \mathcal{K}_i \cup \{k_1\}$.

Since $\gamma^*(\{1, \ldots, K\} \setminus (\mathcal{K}_i \cup \{k_1\})) > \gamma^*(\{1, \ldots, K\})$, there exists a server j_2 and tasks $k \in \mathcal{K}_i \cup \{k_1\}$ and $k_2 \notin \mathcal{K}_i \cup \{k_1\}$ such that $\delta^*_{j_2,k}\mu_{j_2,k}f_{j_2,k} > 0$ and $\mu_{j_2,k_2}f_{j_2,k_2} > 0$. Hence, as for j_1 and k_1, we can decrease $\delta^*_{j_2,k}$ and increase $\delta^*_{j_2,k_2}$, without reducing $\gamma^*(\{1, \ldots, K\})$, so that (1) is not tight for any $k' \in \mathcal{K}_i \cup \{k_1, k_2\}$.

It is clear that if we proceed in this manner, eventually (1) is not tight for any $k' \in \{1, \ldots, K\}$, and hence if we define $\tilde{\gamma}^*(\{1, \ldots, K\})$ to be the optimal solution of the LP with λ_i decreased, we have $\tilde{\gamma}^*(\{1, \ldots, K\}) > \gamma^*(\{1, \ldots, K\})$.

Part (ii) follows immediately as $\mathcal{K}_i \subseteq \Gamma^c$ and (5) not holding imply that Γ and Γ^c do not have a server j in common such that $\delta^*_{j,k}\mu_{j,k}f_{j,k} > 0$ and $\mu_{j,k'}\delta_{j,k'} > 0$ where $k \in \Gamma^c$ and $k' \in \Gamma$. $\diamondsuit$

It may also be worthwhile to express our results about the sensitivity of system performance to changes in the service rates. The interpretation of Theorem 2 below is similar to that of Theorem 1, with the uncertainty addressed being in the service rates rather than the arrival rates.

Theorem 2 *(i) Suppose that (5) holds. Fix* $j \in \{1, \ldots, M\}$ *and* $k \in \{1, \ldots, K\}$ *such that* $\delta^*_{j,k} > 0$. *If we increase* $\mu_{j,k} f_{j,k}$, *then* $\gamma^*(\{1, \ldots, K\})$ *is increased.*
(ii) Suppose (5) does not hold for some strict subset Γ *of* $\{1, \ldots, K\}$. *Fix* $k \notin \Gamma$. *If we increase* $\mu_{j,k} f_{j,k}$ *by* any amount, *then* $\gamma^*(\{1, \ldots, K\})$ *remains unchanged.*

Proof. The proof follows that of Theorem 1, with $\mathcal{K}_i$ replaced by $\{k\}$. $\diamondsuit$

Theorem 2 bears some resemblance to work on "complete resource pooling" in the diffusion limit literature, even though the questions being addressed are different. For the sake of continuity, the discussion of connections between that body of work and ours is included in Appendix A.

Our results are easily extended to unreliable servers. If the proportion of time server j is up is given by u_j, then we can simply replace $\mu_{j,k}$ by $u_j \mu_{j,k}$ in the preceding development, so that capacity effective flexibility structures that account for server failures would simply be based on the effective rates $u_j \mu_{j,k}$. One could also extend these results to more complex failure models (allowing for task failures and dependencies), but one would have to examine a similar generalization of the allocation LP given by (4)-(6) in [5].

Finally, while we believe (4) and (5) provide valuable insights and guidelines for designing desirable flexibility structures, they are certainly not the final word on this subject. For example, our observations are specific to throughput as the performance measure, the situation becomes more complex for performance measures such as mean waiting times or holding

costs. Also, while our results give guarantees as to when changes in demand or processing rates may be accommodated, a manager may also be interested in flexibility that targets sources of variability. Moreover, we have not considered constraints on server flexibility (e.g., certain tasks may not be performed by the same server, while other tasks may require a dedicated server) or on cost. Such constraints may lead one to make lower investments than recommended here (e.g., if the return on investment is not sufficient), or additional investments. More detailed analysis would need to be performed for such additional concerns.

5 Specific Flexibility Structures

In this section, we examine specific flexibility structures with the goal of identifying when they are capacity effective (in the sense of equations (4) and (5)). We start by considering chains, which have been suggested as desirable flexibility structures in the literature (see Section 1). We by no means claim to be the first to explore this issue, but with the pioneering work of [21] as a starting point, we endeavor to identify explicit conditions under which chaining is and is not capacity effective, for specific systems. This is the subject of Sections 5.1 and 5.2. Section 5.3 examines when targeted flexibility (where all flexibility is concentrated on either one task or one server) is capacity effective. In both Sections 5.1 and 5.3, we present results for systems of arbitrary size and structured service rates, while in Sections 5.2 and 5.3, we present results for three tasks and three servers for arbitrary service rates. Finally, Section 5.4 discusses when two popular structures (the "N" and "W" structures from the call center literature) for small systems are capacity effective (for arbitrary service rates). The proofs of all results in this section are provided in an online companion.

Define $\tilde{\lambda}_k = a_k \lambda_{i(k)}$. In the remainder of this section, our insights depend only on the values of $\tilde{\lambda}_k$, and hence will hold for all network topologies that give rise to the same values

of $\tilde{\lambda}_k$. Just to give one small example, let $K = 3$. The following three systems all give rise to $\tilde{\lambda}_1 = \tilde{\lambda}_2 = 1.0$ and $\tilde{\lambda}_3 = 0.5$.

1. The three tasks are in parallel (with $N = 3$, $\lambda_1 = \lambda_2 = 1$ and $\lambda_3 = 0.5$, $\mathcal{K}_1 = \{1\}$, $\mathcal{K}_2 = \{2\}$, $\mathcal{K}_3 = \{3\}$, $p_{0,1} = p_{0,2} = p_{0,3} = 1$, and $p_{k,k'} = 0$, $k, k' = 1, 2, 3$).

2. The three tasks are in tandem (with $N = 1$, $\lambda_1 = 1$, $\mathcal{K}_1 = \{1, 2, 3\}$, $p_{0,1} = p_{1,2} = 1$, $p_{2,3} = 0.5$, and $p_{2,1} = p_{2,2} = p_{3,1} = p_{3,2} = p_{3,3} = 0$).

3. The three tasks are in tandem with feedback (with $N = 1$, $\lambda_1 = 0.5$, $\mathcal{K}_1 = \{1, 2, 3\}$, $p_{0,1} = p_{1,2} = 1$, $p_{2,1} = p_{2,3} = 0.5$, and $p_{3,1} = p_{3,2} = p_{3,3} = 0$).

Much of the literature is based on the parallel server model (in particular [11] and [21] base their insights on such a model). In addition, the literature on call centers is concerned with such models. For excellent overviews of the vast literature in this area, see Aksin et al. [1, 3] and Gans et al. [10]. Hopp et al. [17] and Iravani et al. [20] examine the balanceability of tandem and parallel servers, respectively, under chaining flexibility structures. Here, our insights are for more general systems, but include the systems above as special cases.

5.1 Chains with structured service rates

Let $\mu_{j,k} = \mu = 100$, $K = M = 10$, and $\tilde{\lambda} = [64, 53, 123, 99, 78, 118, 82, 84, 117, 132]$, where the kth entry in the vector $\tilde{\lambda}$ is the arrival rate $\tilde{\lambda}_k$. Now, consider the "2-chain" flexibility structure $f_{j,j} = f_{j,j+1} = 1$ for $j = 1, \ldots, 9$, $f_{10,10} = f_{10,1} = 1$, and $f_{j,k} = 0$ otherwise (see Figure 3 for a "2-chain" with $M = K = 5$). One can verify that this structure satisfies (4) and (5) with $\bar{\gamma} = 1.0526$, and thus Theorem 1 (i) holds and the system is stable. Suppose that the servers are "generalists" in the sense that $\mu_{j,k}$ is given by $\beta_j \mu_k$ for $j = 1, \ldots, M$ and $k = 1, \ldots, K$. Here, β_j characterizes the intrinsic speed of server j and μ_k captures

the inherent difficulty of task k. Also, let $\Gamma_{i,n}$ be the set of consecutive tasks starting at i and containing n tasks, where K and 1 are also considered consecutive tasks. For example, $\Gamma_{i,1} = \{i\}$ and $\Gamma_{K-1,3} = \{K-1, K, 1\}$. Finally, define $\Gamma_{0,n} = \Gamma_{K,n}$.

Proposition 2 *If $K = M$, $\mu_{j,k} \equiv \beta_j \mu_k > 0$ for all j, k, and*

$$\frac{\sum_{j \in \Gamma_{i-1,n+1}} \beta_j}{\sum_{k \in \Gamma_{i,n}} \tilde{\lambda}_k / \mu_k} > \frac{\sum_{j=1}^{M} \beta_j}{\sum_{k=1}^{K} \tilde{\lambda}_k / \mu_k}, \tag{6}$$

for $i = 1, \ldots, K$ and $n = 1, \ldots, K-2$, then the "2-chain" flexibility structure $f_{j,j} = f_{j,j+1} = 1$, $j = 1, \ldots, M-1$, $f_{j,M} = f_{j,1} = 1$, and $f_{j,k} = 0$ otherwise, is capacity effective.

The condition (6) states that the offered load due to any subset consisting of adjacent tasks in isolation must be less than the overall system load. It may be useful to note that (6) automatically holds for $n \in \{K-1, K\}$ and all $i = 1, \ldots, K$.

To further illustrate under what circumstances 2-chaining is capacity effective, we examine three special cases. We first consider the case where the servers are identical. For the following corollary, define the average arrival rate (over all tasks) as $\bar{\lambda} = \sum_{k=1}^{K} \tilde{\lambda}_k / K$.

Corollary 2 *If $K = M$, $\mu_{j,k} \equiv \mu$, for all j, k, and*

$$\sum_{k \in \Gamma_{i,n}} \tilde{\lambda}_k < (n+1)\bar{\lambda}, \tag{7}$$

for $i = 1, \ldots, K$ and $n = 1, \ldots, K-2$, then the "2-chain" flexibility structure $f_{j,j} = f_{j,j+1} = 1$, $j = 1, \ldots, M-1$, $f_{j,M} = f_{j,1} = 1$, and $f_{j,k} = 0$ otherwise, is capacity effective.

The condition (7) requires the arrival rates to be balanced in an appropriate manner. In particular, it limits to what degree groups of neighboring arrival rates can differ from the average, and also limits the maximum arrival rate to be less than twice the average ($\bar{\lambda}$). One particular example where (7) trivially holds is if $\tilde{\lambda}_k = \tilde{\lambda}$ for $k = 1, \ldots, K$.

Suppose that in the above setting $\tilde{\lambda}_k \equiv \tilde{\lambda}$ and $\mu_{j,k} = \mu_k$ for all j, k (i.e., the service rates depend only on the task, a common assumption in the literature). Similar to Corollary 2, (6) translates into a condition which requires groups of neighboring mean service times to be close to the average mean service time, given by $\bar{m} = \left(\sum_{k=1}^{K} 1/\mu_k\right)/K$.

Corollary 3 *If $K = M$, $\tilde{\lambda}_k \equiv \tilde{\lambda}$, for all k, $\mu_{j,k} = \mu_k$ for all j, k and*

$$\sum_{k \in \Gamma_{i,n}} 1/\mu_k < (n+1)\bar{m},$$

for $i = 1, \ldots, K$ and $n = 1, \ldots, K-2$, then the "2-chain" flexibility structure $f_{j,j} = f_{j,j+1} = 1$, $j = 1, \ldots, M-1$, $f_{j,M} = f_{j,1} = 1$, and $f_{j,k} = 0$ otherwise, is capacity effective.

Finally, suppose that $\tilde{\lambda}_k \equiv \tilde{\lambda}$ and $\mu_{j,k} = \beta_j$ for all j, k (i.e., the service rates depend only on the server). Similar to Corollary 2, (6) translates into a condition which requires groups of neighboring mean service rates to be close to the average mean service rate, given by $\bar{\beta} = \sum_{j=1}^{M} \beta_j/M$.

Corollary 4 *If $K = M$, $\tilde{\lambda}_k \equiv \tilde{\lambda}$, for all k, $\mu_{j,k} = \beta_j$ for all j, k and*

$$\sum_{j \in \Gamma_{i,n+1}} \beta_j > n\bar{\beta},$$

for $i = 1, \ldots, K$ and $n = 1, \ldots, K-2$, then the "2-chain" flexibility structure $f_{j,j} = f_{j,j+1} = 1$, $j = 1, \ldots, M-1$, $f_{j,M} = f_{j,1} = 1$, and $f_{j,k} = 0$ otherwise, is capacity effective.

The results above are consistent with the work of Chou et al. [9], who show that long 2-chains perform well when demand and service rates are homogeneous. Our results quantify the degree to which demand and service rates may deviate from homogeneity in long 2-chains, while still being capacity effective.

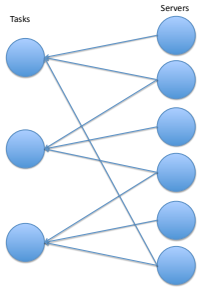

Figure 6: Tailored Chaining

Our conditions may also be useful in analyzing structures within which a chain exists. One example of this is the "tailored chaining" approach of [7]. A particular system that has this structure is given in Figure 6. For the model in [7], where $\mu_{j,k} \equiv \mu$ and $\tilde{\lambda}_k \equiv \tilde{\lambda}$ for all j, k, and K arbitrary ($M = 2K$ for this structure), it is not difficult to show that the proposed "tailored chaining" structure is capacity effective. It is also not difficult to show that the "tailored pairing" approach of [7] is capacity effective for arbitrary K ($M = K + K(K+1)/2$ for this structure) under the same assumptions on the arrival and service rates.

We will next discuss when chaining is the minimal capacity effective flexibility structure (i.e., if any one skill is removed, the structure is no longer capacity effective). For the example at the beginning of this section, if we "break" the chain by setting $f_{3,4} = 0$, we then have $\gamma^*(\{1, \ldots, K\}) = 0.9859$, the system is unstable, and (5) does not hold. One general result that we can show is that, if $\tilde{\lambda}_k \equiv \tilde{\lambda}$ for all k, then chaining is a minimal capacity effective flexibility structure. This is consistent with the observation on the value of "completing the chain" in [17, 21].

Proposition 3 *Suppose that $K = M$, $\mu_{j,k} \equiv \mu$, and $\tilde{\lambda}_k \equiv \tilde{\lambda}$ for all j, k. If for the "2-chain" structure described in Proposition 2 we change $f_{j,k}$ from 1 to 0 for some j, k, then (5) is violated.*

Proposition 2 and Corollaries 2, 3, and 4 illustrate how the order of the tasks and servers matters. For example, Corollaries 2, 3, and 4 require, respectively, the average adjacent arrival rates, mean service times, or service rates not to differ too far from their overall averages $\bar{\lambda}$, $\bar{m}$, and $\bar{\beta}$. Thus, some "2-chain" flexibility structures may be capacity effective, but not others. The third paragraph of Section 4 provides an example with a "2-chain" structure which satisfies (5) but not (4). The "2-chain" structure $f_{1,3} = f_{2,1} = f_{3,2} = f_{1,2} = f_{2,3} = f_{3,1} = 1$ and $f_{j,k} = 0$ otherwise, is capacity effective.

5.2 Chains with arbitrary service rates

We present a general result for when 2-chaining is capacity effective, in the case when $K = M = 3$ and service rates are arbitrary. As far as we know, this is the first explicit result in such generality, as previous analytic results [9, 21] have been for systems with homogeneous servers. The conditions in the first part of Proposition 4 quantify the intuition that in order for 2-chaining to be desirable, no one arrival rate should be too dominant ((8) can be rewritten to reflect this for task k) and no server should be too dominant (both (9) and (10) can be rewritten to reflect this for server j). Moreover, the proof of Proposition 4 shows that it is not capacity effective to target all flexibility on a single demand when condition (8) holds, or on one server when conditions (9) and (10) hold. The conditions are actually written in terms of service rates normalized by corresponding arrival rates, i.e., we let $\bar{\mu}_{j,k} = \mu_{j,k}/\tilde{\lambda}_k$ for $j, k = 1, 2, 3$.

The second part of Proposition 4 gives sufficient conditions under which we can choose a specific 2-chain that is capacity effective for $K = M = 3$. For all servers j and pairs of tasks (k, k'), let $\mu_{j,k}/\mu_{j,k'}$ denote the relative ability of server j at task k (relative to the server's ability at task k'). Condition (11) of Proposition 4 shows that if each server has the largest

relative ability for one pair of tasks (1,2), (2,3), or (3,1) and the smallest relative ability for another pair, then each server should be assigned to the pair of tasks where his/her abilities are the most balanced. This chain ensures that the servers never spend time at tasks where they have the lowest relative ability (relative to any other task).

Proposition 4 *Suppose that* $\mu_{j,k} > 0$ *for* $j, k = 1, 2, 3$ *and each of the following hold:*

1. *for all* j, j', j'' *and* k, k', k'' *such that* $\{j, j', j''\}$, $\{k, k', k''\} = \{1, 2, 3\}$, *and* $\mu_{j,k} \leq \min\{\mu_{j',k'}, \mu_{j'',k''}\}$, *either*

$$\bar{\mu}_{j',k'} \leq \bar{\mu}_{j'',k''} \left(\frac{\bar{\mu}_{j,k} + \bar{\mu}_{j'',k}}{\bar{\mu}_{j'',k''} + \bar{\mu}_{j'',k}} \right) \quad \text{or} \quad \bar{\mu}_{j'',k''} \leq \bar{\mu}_{j',k'} \left(\frac{\bar{\mu}_{j,k} + \bar{\mu}_{j',k}}{\bar{\mu}_{j',k'} + \bar{\mu}_{j',k}} \right); \tag{8}$$

2. *for all* j, j', j'' *and* k, k', k'' *such that* $\{j, j', j''\}$, $\{k, k', k''\} = \{1, 2, 3\}$, *and* $\mu_{j,k} \geq \max\{\mu_{j',k'}, \mu_{j'',k''}\}$, *either*

$$\bar{\mu}_{j',k'} \geq \bar{\mu}_{j,k} \left(\frac{\bar{\mu}_{j,k''} + \bar{\mu}_{j'',k''}}{\bar{\mu}_{j,k} + \bar{\mu}_{j,k''}} \right) \quad \text{or} \quad \bar{\mu}_{j'',k''} \geq \bar{\mu}_{j,k} \left(\frac{\bar{\mu}_{j,k'} + \bar{\mu}_{j',k'}}{\bar{\mu}_{j,k} + \bar{\mu}_{j,k'}} \right); \tag{9}$$

3. *for all* $j, k = 1, 2, 3$,

$$\sum_{j' \neq j} \bar{\mu}_{j',k} \geq \frac{\prod_{k' \neq k} \bar{\mu}_{j,k'}}{\sum_{k' \neq k} \bar{\mu}_{j,k'}}. \tag{10}$$

Then a 2-chain flexibility structure is capacity effective. In particular, if

$$\left.\begin{array}{rcl} \frac{\mu_{2,1}}{\mu_{2,2}} & \leq & \frac{\mu_{1,1}}{\mu_{1,2}} \leq \frac{\mu_{3,1}}{\mu_{3,2}}, \\ \frac{\mu_{3,2}}{\mu_{3,3}} & \leq & \frac{\mu_{2,2}}{\mu_{2,3}} \leq \frac{\mu_{1,2}}{\mu_{1,3}}, \\ \frac{\mu_{1,3}}{\mu_{1,1}} & \leq & \frac{\mu_{3,3}}{\mu_{3,1}} \leq \frac{\mu_{2,3}}{\mu_{2,1}}, \end{array}\right\} \tag{11}$$

then (4) and (5) hold for the 2-chain flexibility structure with the non-zero values of $f_{j,k}$ *being given by* $f_{1,1} = f_{1,2} = f_{2,2} = f_{2,3} = f_{3,3} = f_{3,1} = 1$.

5.3 Targeted Flexibility

It is not true that (4) and (5) only hold if the flexibility structure is a chain. Consider the following example. Let $M = K = 2$, $\mu_{j,k} = 1$ for all j, k and $\tilde{\lambda}_1 = 1 - 2\varepsilon$, $\tilde{\lambda}_2 = 1 - \varepsilon$ for some $0 < \varepsilon < 1/2$. Now, for this example the "2-chain" structure and full flexibility are identical, with $\bar{\gamma} = 2/(2 - 3\varepsilon)$. If we set $f_{2,1} = 0$, then (4) and (5) still hold. However, if we set $f_{1,2} = 0$, then $\gamma^*(\{2\}) = \gamma^*(\{1, 2\}) = 1/(1 - \varepsilon)$, and thus both (4) and (5) are violated. This discrepancy is due to the unbalanced demand.

The above idea can be generalized to the following result, which can be thought of as the other extreme from balanced demand. The resulting flexibility structure is in some sense the opposite of chaining: all servers must be trained for task 1, while $M - 1$ servers are each trained for a different one of the remaining tasks (see Figure 4 for an example with $K = M = 5$). It is instructive to note that this structure requires fewer skills than the "2-chain" structure.

Proposition 5 *Suppose $K = M > 2$, $\mu_{j,k} \equiv \mu$ for all j, k, $\tilde{\lambda}_1 > (M - 1)\mu$, and $\sum_{i=1}^{K} \tilde{\lambda}_i < M\mu$.*

(i) The structure $f_{j,1} = 1$, $j = 1, \ldots, M$, $f_{j,j} = 1$, $j = 2, \ldots, M$, and $f_{j,k} = 0$ otherwise, is capacity effective.

(ii) The "2-chain" structure described in Proposition 2 does not satisfy (4).

As for the chaining structures in the previous subsections, we can generalize Proposition 5 to arbitrary service rates in the case when there are three servers and tasks. In what follows we use the convention that $0/0 = 0$ and $x/0 = \infty$ when $x > 0$. Note that conditions (13) and (14) imply that server 3 (2) has the largest relative ability at task 3 (2) (relative to the server's ability at task 1), and that condition (12) implies that server 2 (3) can cover

the work at task 2 (3) while also helping with task 1. Moreover, Proposition 6 is consistent with Proposition 5 in that when $\mu_{j,k} = \mu$ for all j, k (the case considered in Proposition 5), conditions (13) and (14) always hold, and condition (12) holds if $\tilde{\lambda}_1$ is large relative to $\tilde{\lambda}_2, \tilde{\lambda}_3$.

Proposition 6 *If $K = M = 3$, $\mu_{2,1}, \mu_{3,1} > 0$,*

$$\frac{\bar{\mu}_{1,1} + \bar{\mu}_{3,1}}{\bar{\mu}_{3,1} + \bar{\mu}_{3,3}} < \frac{\bar{\mu}_{2,2}}{\bar{\mu}_{3,3}} < \frac{\bar{\mu}_{2,1} + \bar{\mu}_{2,2}}{\bar{\mu}_{1,1} + \bar{\mu}_{2,1}}, \tag{12}$$

$$\frac{\mu_{3,3}}{\mu_{3,1}} \geq \max\left\{\frac{\mu_{1,3}}{\mu_{1,1}}, \frac{\mu_{2,3}}{\mu_{2,1}}\right\}, \quad \textit{and} \tag{13}$$

$$\frac{\mu_{2,2}}{\mu_{2,1}} \geq \max\left\{\frac{\mu_{1,2}}{\mu_{1,1}}, \frac{\mu_{3,2}}{\mu_{3,1}}\right\}, \tag{14}$$

then the flexibility structure

$$f_{j,k} = \begin{cases} 1 & \textit{if } (j,k) \in \{(1,1),(2,1),(3,1),(2,2),(3,3)\}, \\ 0 & \textit{otherwise} \end{cases}$$

in which all servers are trained at task 1 is capacity effective.

We now consider the case where one server dominates. Consider the following example. Suppose that $M = K = 3$, $\tilde{\lambda}_1 = \tilde{\lambda}_2 = \tilde{\lambda}_3 = 3.5$, and the service rates are $\mu_{1,k} = \mu_{3,k} = 1$ for all k and $\mu_{2,k} = 10$ for all k. If we set $f_{1,1} = f_{3,3} = f_{2,1} = f_{2,2} = f_{2,3} = 1$ and all other $f_{j,k} = 0$, then it is not difficult to show that (4) and (5) hold. If we use the "2-chain" structure, i.e., $f_{1,1} = f_{1,2} = f_{2,2} = f_{2,3} = f_{3,3} = f_{3,1} = 1$, and $f_{j,k} = 0$ otherwise, we see that (4) and (5) are both violated.

We can generalize this example. If one server is sufficiently dominant in terms of its service rate, we have the following result, similar in spirit to Proposition 5. In this case, one should simply train the dominant server for all tasks, with the remaining servers trained for exactly one task (see Figure 5 for one example of this with $K = M = 5$). Not only does this result in a capacity effective flexibility structure, it requires fewer skills than chaining.

Proposition 7 *Suppose $K = M > 2$, $\mu_{j,k} = \mu$, $j = 2, \ldots, M$, $k = 1, \ldots, K$. In addition, assume that $\tilde{\lambda}_i = \tilde{\lambda}$, $i = 1, \ldots, K$. If, for some $d > K + 1$, $\mu_{1,k} = d\mu$, $k = 1, \ldots, K$, then*

(i) The structure $f_{1,k} = 1$, $k = 1, \ldots, K$, $f_{j,j} = 1$, $j = 2, \ldots, M$, and $f_{j,k} = 0$ otherwise, is capacity effective.

(ii) The "2-chain" structure described in Proposition 2 does not satisfy (4).

Note that it is not difficult to relax the condition that $K = M > 2$ and $d > K + 1$ to $2K + M \geq 2$ and $d > K - 1$ in part (i) of Proposition 7.

We can also give the following result for arbitrary service rates, similar in spirit to Proposition 6. Note that conditions (16) and (17) imply that server 2 (3) is relatively better at task 2 (3) than server 1 (relative to tasks 1,3 (1,2)), and condition (15) implies that server 1 can cover the work at task 1 while also helping with tasks 2 and 3. Moreover, Proposition 8 is consistent with Proposition 7 in that when $\mu_{1,k} = d\mu$ and $\mu_{j,k} = \mu$ for $j = 2, \ldots, M$ and all k and $\tilde{\lambda}_i = \tilde{\lambda}$ for all i (the case considered in Proposition 7), conditions (16) and (17) always hold, and condition (15) holds if $d > 1$, so that $\mu_{1,k}$ is large relative to $\mu_{2,k}, \mu_{3,k}$ for all k.

Proposition 8 *If $K = M = 3$, $\mu_{1,2}, \mu_{1,3} > 0$,*

$$\bar{\mu}_{1,1} > \max\left\{\bar{\mu}_{2,2}\left(\frac{\bar{\mu}_{1,1} + \bar{\mu}_{1,3}}{\bar{\mu}_{1,3} + \bar{\mu}_{3,3}}\right), \bar{\mu}_{3,3}\left(\frac{\bar{\mu}_{1,1} + \bar{\mu}_{1,2}}{\bar{\mu}_{1,2} + \bar{\mu}_{2,2}}\right)\right\}, \tag{15}$$

$$\frac{\mu_{3,2}}{\mu_{3,3}} \leq \frac{\mu_{1,2}}{\mu_{1,3}}, \quad \frac{\mu_{2,3}}{\mu_{2,2}} \leq \frac{\mu_{1,3}}{\mu_{1,2}}, \tag{16}$$

$$\frac{\mu_{1,1}}{\mu_{1,2}} \geq \frac{\mu_{2,1}}{\mu_{2,2}}, \quad \textit{and} \quad \frac{\mu_{1,1}}{\mu_{1,3}} \geq \frac{\mu_{3,1}}{\mu_{3,3}}, \tag{17}$$

then the flexibility structure

$$f_{j,k} = \begin{cases} 1 & \textit{if } (j,k) \in \{(1,1),(1,2),(1,3),(2,2),(3,3)\}, \\ 0 & \textit{otherwise} \end{cases}$$

in which server 1 is trained at all tasks is capacity effective.

The results to this point have given scenarios that suggest either chaining or concentrating all training on either one demand type or one server. For these extremes, we provide explicit results for the flexibility structures to be capacity effective. This shows that capacity effective flexibility structures could range from being balanced like the "2-chain" structure to focusing all flexibility (beyond satisfying base demand) on one task or on one server. To enumerate all intermediate possibilities and study their performance is impractical, but one can envision that anything between these two extremes would be possible, depending on the level of heterogeneity.

5.4 The "N" and "W" structures

In this subsection, we discuss two other flexibility structures for small, parallel systems that have been studied in the call center literature. The first is sometimes referred to as the *"N" structure*, see Figure 7. It has two servers, two tasks ($M = K = 2$), and a flexibility structure

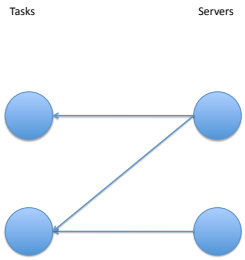

Figure 7: The "N" structure

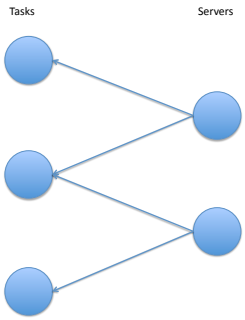

Figure 8: The "W" structure

where one server is trained for both tasks, the other for just one task. Such a structure has arisen in a number of settings, in particular the bilingual call center model of Stanford and Grassman [26]. This model has also been studied by Shumsky [25]. The following result

generalizes part (i) of Proposition 7 to more general arrival and service rates when there are two servers and tasks.

Proposition 9 *Assume that the servers and tasks are labelled such that* $\mu_{1,1} > 0$, $\mu_{2,2} > 0$, $\mu_{1,2} > 0$, $\mu_{1,1}\mu_{2,2} \geq \mu_{2,1}\mu_{1,2}$, *and* $\tilde{\lambda}_1/\mu_{1,1} < \tilde{\lambda}_2/\mu_{2,2}$. *The "N" flexibility structure given by* $f_{1,1} = f_{1,2} = f_{2,2} = 1$ *and* $f_{2,1} = 0$ *is capacity effective.*

The interpretation of the above result is quite straightforward. First, to achieve maximum throughput, server 2 should be at task 2, server 1 at task 1 (unless idle). To be able to shift capacity in an appropriate manner, the load at task 2 using server 2 only must be greater than the load at task 1 for server 1 only. Here, server 1 must be able to serve task 2, but fluctuations in the load at task 1 can be handled by server 1 alone. It is easy to see that the wrong "N" structure may achieve arbitrarily poor capacity. Fix $\tilde{\lambda}_1 = \tilde{\lambda}_2 = \mu_{2,1} = \mu_{2,2} = 1$. Also, let $\mu_{1,1}$ and $\mu_{1,2}$ go to zero. Here, the assumption $\tilde{\lambda}_1/\mu_{1,1} < \tilde{\lambda}_2/\mu_{2,2}$ is violated. Then $\gamma^*(\{1,2\})$ for the "N" structure described above approaches zero, while if we change $f_{2,1}$ to 1 and $f_{1,2}$ to 0 (also an "N" structure), then $\gamma^*(\{1,2\})$ goes to 1/2.

The second flexibility structure from the call center literature that we consider is sometimes called the *"W" structure*, see Figure 8 (see also Saghafian et al. [23] for a detailed study of the "W" structure). It has $M = 2$ servers and $K = 3$ tasks, with each server trained for two tasks in such a way that all three tasks are covered. Without loss of generality, we assume that the servers and tasks are labelled such that

$$\frac{\mu_{1,3}}{\mu_{2,3}} \leq \frac{\mu_{1,2}}{\mu_{2,2}} \leq \frac{\mu_{1,1}}{\mu_{2,1}}. \tag{18}$$

Furthermore, we assume that $\mu_{1,1} > 0$, $\mu_{1,2} > 0$, $\mu_{2,2} > 0$, $\mu_{2,3} > 0$, and the following inequalities hold:

$$\frac{\tilde{\lambda}_1}{\mu_{1,1}} < \frac{\tilde{\lambda}_2}{\mu_{2,2}} + \frac{\tilde{\lambda}_3}{\mu_{2,3}}, \tag{19}$$

$$\frac{\tilde{\lambda}_3}{\mu_{2,3}} < \frac{\tilde{\lambda}_1}{\mu_{1,1}} + \frac{\tilde{\lambda}_2}{\mu_{1,2}}. \tag{20}$$

In light of (18), this flexibility structure corresponds to training both servers for the task that has the most balanced rates between servers, and having only server 1 serve task 1 and only server 2 serve task 3. The assumptions (19) and (20) require the load at each of the tasks where there is only one server to be strictly less than the load at the remaining two tasks if served by the other server. This means that the two servers must both work at task 2 at optimal throughput levels. This gives simple sufficient conditions in terms of the system parameters.

Proposition 10 *Let $M = 2$ and $K = 3$. Suppose that $\mu_{1,1} > 0$, $\mu_{1,2} > 0$, $\mu_{2,2} > 0$, $\mu_{2,3} > 0$, and the service rates satisfy (18)-(20). The "W" structure defined by $f_{1,1} = f_{1,2} = f_{2,2} = f_{2,3} = 1$ and $f_{2,1} = f_{2,3} = 0$ is capacity effective.*

Note that an arbitrarily defined "W" structure can be significantly worse than that described above, so that the labeling of tasks and servers (18) and the assumptions (19)-(20) are crucial. Let $\tilde{\lambda}_1 = \tilde{\lambda}_2 = \tilde{\lambda}_3 = 1$ and begin with $\mu_{j,k} = 1$ for $j, k = 1, 2, 3$. If we change $\mu_{1,1}$ to be close to zero, then the capacity will be close to zero. Here, (18) and (19) are violated. Choosing any other "W" structure that has $f_{1,1} = 0$ results in capacity $2/3$. If we instead let $\mu_{1,2}$ be close to zero, then the capacity will be close to $1/2$, compared to the capacity of $2/3$ for any "W" structure with $f_{1,2} = 0$. In this case, (18) is violated.

It appears that it would be quite straightforward to use (4) and (5) to quickly evaluate other structures.

6 Concluding Remarks

We have provided means to identify flexibility structures that are throughput optimal and adaptable to changes in the environment, manifested by perturbations in arrival and/or service rates. Our approach is not only intuitive but is also computationally efficient. To accomplish this, we have introduced the notion of a bottleneck set of tasks to queueing networks with flexible, heterogeneous servers, so that the bottleneck set may include several queues and servers. As a result, we have identified minimal conditions that should be required of any flexibility structure, if possible. We have further specialized these results to obtain insights for more specific structures, including chains.

Our research yields the following managerial insights:

1. As in a system with dedicated servers, the bottleneck set limits system performance (in this case throughput and adaptation to changes in the environment). The bottleneck set may span several tasks and may not be obvious a priori, however it is easily determined by solving several associated LPs.

2. It is desirable for the unique bottleneck set to be the entire set of tasks because this allows capacity to be shifted to compensate for fluctuations in demand and/or service rates.

3. When demand and service ability are sufficiently balanced, skill chaining is known to be an effective strategy, but it is suboptimal in more heterogeneous settings. We have provided explicit criteria for determining precisely when chaining and other cross training strategies are capacity effective (i.e., throughput optimal and adaptable to changes in the environment), including the well known "N" and "W" structures defined in the call center literature.

In terms of future work, one obvious question is: given a number of flexibility structures that are all capacity effective, how could one make a more refined choice? This is a topic of interest, see in particular the work of Aksin and Karaesmen [2] and Iravani et al. [18, 19, 20]. To this end, we are currently interested in how the work in this paper can be leveraged into developing metrics that compare flexibility structures. Here, one is typically interested in performance metrics such as (mean) waiting times, holding costs, etc.

Acknowledgments

This research was supported by the National Science Foundation under Grant CMMI-0856600. The research of the third author was also supported by the Natural Sciences and Engineering Research Council of Canada.

References

[1] O.Z. Aksin, M. Armony, and V. Mehrotra. The modern call-center: A multi-disciplinary perspective on Operations Management research. *Production and Operations Management*, 16:665-688, 2007.

[2] O.Z. Aksin and F. Karaesmen. Characterizing the performance of process flexibility structures. *Operations Research Letters*, 35:477-484, 2007.

[3] O.Z. Aksin, F. Karaesmen, and E.L. Ormeci. Workforce cross-training in call centers from an Operations Management perspective. Chapter 8 in *Workforce Cross Training Handbook*, ed. D. Nembhard, CRC Press, 2007.

[4] S. Andradóttir, H. Ayhan, and D.G. Down. Dynamic server allocation for queueing networks with flexible servers. *Operations Research*, 51:952-968, 2003.

[5] S. Andradóttir, H. Ayhan, and D.G. Down. Compensating for failures with flexible servers. *Operations Research*, 55:753-768, 2007.

[6] R. Atar. Scheduling control for queueing systems with many servers: Asymptotic optimality in heavy traffic. *Annals of Applied Probability*, 15:2606-2650, 2005.

[7] A. Bassamboo, R.S. Randhawa, and J.A. Van Mieghem. A little flexibility is all you need: On the asymptotic value of flexible capacity in parallel queueing systems. Preprint, 2011.

[8] M. Bramson and R.J. Williams. Two workload properties for Brownian networks. *Queueing Systems*, 45:191-221, 2003.

[9] M.C. Chou, G.A. Chua, C.-P. Teo, and H. Zheng. Design for process flexibility: Efficiency of the long chain and sparse structure. *Operations Research*, 58:43-58, 2010.

[10] N. Gans, G. Koole, and A. Mandelbaum. Telephone call centers: Tutorial, review, and research prospects. *Manufacturing and Service Operations Management*, 5:79-141, 2003.

[11] S.C. Graves and B.T. Tomlin. Process flexibility in supply chains. *Management Science*, 49:907-919, 2003.

[12] S. Gurumurthi and S. Benjaafar. Modeling and analysis of flexible queueing systems. *Naval Research Logistics*, 51:755-782, 2004.

[13] I. Gurvich and W. Whitt. Queue-and-idleness-ratio controls in many-server service systems. *Mathematics of Operations Research*, 34:363-396, 2009.

[14] J.M. Harrison. Brownian models of open processing networks: Canonical representation of workload. *Annals of Applied Probability*, 10:75-103, 2000.

[15] J.M. Harrison. Correction - Brownian models of open processing networks: Canonical representation of workload. *Annals of Applied Probability*, 13:390-393, 2003.

[16] J.M. Harrison and M.J. López. Heavy traffic resource pooling in parallel-server systems. *Queueing Systems*, 33:339-368, 1999.

[17] W.J. Hopp, E. Tekin, and M.P. van Oyen. Benefits of skill chaining in serial production lines with cross-trained workers. *Management Science*, 50:83-98, 2004.

[18] S.M.R. Iravani, B. Kolfal, and M.P. van Oyen. Call-center labor cross-training: It's a small world after all. *Management Science*, 53:1102-1112, 2007.

[19] S.M.P. Iravani, B. Kolfal, and M.P. van Oyen. Capability flexibility: A decision support methodology for parallel service and manufacturing systems with flexible servers. *IIE Transactions*, 43:363-382, 2011.

[20] S.M. Iravani, M.P. van Oyen, and K.T. Sims. Structural flexibility: A new perspective on the design of manufacturing and service operations. *Management Science*, 51:151-166, 2005.

[21] W.C. Jordan and S.C. Graves. Principles on the benefits of manufacturing process flexibility. *Management Science*, 41:577-594, 1995.

[22] W.C. Jordan, R.R. Inman, and D.E. Blumenfeld. Chained cross-training of workers for robust performance. *IIE Transactions*, 36:953-967, 2004.

[23] S. Saghafian, M.P. van Oyen, and B. Kolfal. The "W" network and the dynamic control of unreliable flexible servers. *IIE Transactions*, 43:893-907, 2011.

[24] M. Sheikhzadeh, S. Benjaafar, and D. Gupta. Machine sharing in manufacturing systems: Flexibility versus chaining. *International Journal of Flexible Manufacturing*, 10:351-378, 1998.

[25] R.A. Shumsky. Approximation and analysis of a queueing system with flexible and specialized servers. *OR Spectrum*, 26:307-330, 2004.

[26] D.A. Stanford and W.K. Grassmann. Bilingual server call centres. *Analysis of Communication Networks: Call Centres, Traffic and Performance*, D. R. McDonald and S. R. E. Turner, editors. Fields Institute Communications, 31-47, 2000.

[27] A.L. Stolyar. Optimal routing in output-queued flexible server systems. *Probability in the Engineering and Informational Sciences*, 19:141-189, 2005.

A Relation to Complete Resource Pooling

The allocation LP considered in this paper is closely related to the Static Planning Problem given in Definition 4.1 of Bramson and Williams [8] (this problem was introduced earlier by Harrison and López [16] and Harrison [14, 15]). If the inequality in (1) is replaced by an equality (which can be done without loss of generality), the allocation LP and the Static Planning Problem can be shown to be equivalent. The analysis in [8] continues under the assumption of heavy traffic, which by their definition means that the Static Planning Problem has a unique optimal solution with value 1 (corresponding to $\gamma^* = 1$ for our allocation LP). They then show that the resulting diffusion scaled workload process has reduced dimension. If the resulting scaled workload process is one dimensional, it is said that "complete resource pooling" (CRP) has taken place.

To connect to our results, first note that we do not require a unique solution to the allocation LP. So, for example, let $N = K = 3$ (parallel servers), $\mu_{j,k} = 1$, and $\lambda_k = 1$ for $j, k = 1, 2, 3$. For the flexibility structure, choose the 2-chain $f_{1,1} = f_{1,2} = f_{2,2} = f_{2,3} = f_{3,3} = f_{3,1} = 1$, and $f_{1,3} = f_{2,1} = f_{3,2} = 0$. We have an infinite number of solutions to the allocation LP such that (4) holds with $\gamma^* = \bar{\gamma} = 1$, and (5) also holds. The non-uniqueness of the solution means that CRP cannot be demonstrated. If we modify this example so that $\mu_{1,3} = \mu_{2,1} = \mu_{3,2} = 2$, $\lambda_1 = \lambda_2 = 0.95$, $\lambda_3 = 1.1$, and $f_{3,1} = 0$, then the allocation LP

has a unique optimal solution with $\gamma^* = 1$, so the heavy traffic assumption of [8] holds, and CRP can be demonstrated. However, here (4) does not hold as $\bar{\gamma} \approx 1.95$, and hence this structure does not achieve the maximum possible capacity. From these examples, we see that in general, (4) and (5) do not imply CRP, and CRP does not imply (4). That CRP implies (5) is obvious. This is not to say that there are not cases where the two conditions coincide. If we return to the original service rates of $\mu_{j,k} = 1$, $j, k = 1, 2, 3$, but set $\lambda_1 = 2.8$, $\lambda_2 = 0.1$, $\lambda_3 = 0.1$, $f_{1,1} = f_{2,1} = f_{3,1} = f_{2,2} = f_{3,3} = 1$, and $f_{1,2} = f_{3,2} = f_{1,3} = f_{2,3} = 0$, we see that $\bar{\gamma} = \gamma^* = 1$, so (4) holds. It is also straightforward to check that (5) holds. The solution to the allocation LP is unique, and from there it is not difficult to see that the heavy traffic assumption of [8] holds and CRP can be demonstrated.

The observations above are consistent with work in Stolyar [27], where it is shown that for a system of parallel queues (i.e., $N = K$), then if there is a unique solution to the allocation LP with $\gamma^* = 1$ and the graph that has an arc between a node representing server j and a node representing task k is a fully connected tree (no cycles are permitted), then CRP occurs. Any flexibility structure satisfying (5) has a subgraph that is a fully connected tree (see the proof of Theorem 1). In some cases, a tree structure may satisfy (4) and (5) (for one example, see Proposition 5), but in general it does not (see Proposition 2). Atar [6] and Gurvich and Whitt [13] show that CRP can occur if the allocation LP does not have a unique solution, but they both require that all solutions have a tree structure. The additional structure in our case is due to the desire to protect against changes in the environment (i.e., changes in the means of the underlying distributions), rather than to protect against variability due to (unchanging) underlying distributions. The structures satisfying (4) and (5) protect against both (see Theorems 1 and 2 and also the Introduction, where we discuss that our main goal is to look at effective design in the face of a changing environment, in the spirit of [22]).