Dynamic Server Allocation for Unstable Queueing Networks with Flexible Servers

Salih Tekin
Industrial Engineering Department
TOBB Economy and Technology University
Söğütözü, Ankara, 06560, Turkey

Sigrún Andradóttir
H. Milton Stewart School of
Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0205, U.S.A.

Douglas G. Down
Department of Computing and Software
McMaster University
Hamilton, Ontario L8S 4L7, Canada

July 19, 2011

Abstract

This paper is concerned with the dynamic assignment of servers to tasks in queueing networks where demand may exceed the capacity for service. The objective is to maximize the system throughput. We use fluid limit analysis to show that several quantities of interest, namely the maximum possible throughput, the maximum throughput for a given arrival rate, the minimum arrival rate that will yield a desired feasible throughput, and the optimal allocations of servers to classes for a given arrival rate and desired throughput, can be computed by solving linear programming problems. We develop generalized round robin policies for assigning servers to classes for a given arrival rate and desired throughput, and show that our policies achieve the desired throughput as long as this throughput is feasible for the arrival rate. We conclude with numerical examples that illustrate the points discussed and provide insights into the system behavior when the arrival rate deviates from the one the system is designed for.

Key words and phrases: multi-class queueing networks, stability, fluid model, maximum throughput, Jackson networks.

1 Introduction

This paper investigates multi-class discrete-flow networks with flexible servers when demand is allowed to exceed the capacity for service. Multiple types of customers are serviced by flexible servers that are able to work on several different classes. Offered demand to each class can come from both external sources as well as internal transitions. The same server can have different service rates for different classes. Moving the servers among the classes is assumed to incur switching times that can be different for each origin-destination pair of classes. More than one server can be assigned to a given class, possibly with different service rates. In that case, servers at a class can either cooperate by working simultaneously on a customer, or work in parallel and process the customers separately. We concentrate on the case where the servers work in parallel and there is one arrival stream routed to various classes (cooperating servers and multiple arrival streams are straightforward extensions).

Our aim in this paper is to find the best assignment of servers to classes so that the throughput of the system is maximized. We will refer to the process that the servers use to decide what classes to work on as a policy. To motivate our analysis, consider manufacturing processes where demand exceeds the production capacity and work in process can be either salvaged for some profit or scrapped at small cost compared to the final product value. In these cases, allowing instability in the system might be desirable given the right parameters. Flexible server systems with unstable nodes have not been studied to date. We will quantify the effects of allowing instability on both throughput and server assignments.

In recent years, there has been a growing interest in queueing systems with flexible servers, with most of the work examining holding costs or throughput. Works that minimize total holding costs by studying how servers should be assigned to stations include Ahn, Duenyas, and Lewis [1], Ahn, Duenyas, and Zhang [2, 3], Bell and Williams [9, 10], Bramson and Williams [12], Farrar [21], Hajek [24], Harrison and López [26], Pandelis and Teneketzis [35], Rosberg, Varaiya, and Walrand [36] and Williams [44]. Works that aim to maximize the long-run average throughput through dynamic assignment of reliable servers include Andradóttir, Ayhan, and Down [4, 5, 6] and Tassiulas et al. [39, 40]. By contrast, Andradóttir, Ayhan, and Down [7, 8] and Wu, Lewis, and Veatch [45] determine the optimal allocation of flexible servers in a tandem-line system where servers are not necessarily reliable.

The earliest work we are aware of that considers overloaded systems is by Goodman and Massey [23]. They study non-ergodic Jackson networks and propose a way to determine the maximal subnetwork that achieves steady state. Weiss [43] considers a Jackson network in which some nodes have an infinite supply of customers. He shows that when only customers in transit are counted as congestion, the stable subset of nodes has the usual product-form

distribution. Similarly, a marginal distribution for the number of customers in transit exists for nodes with infinite supply of work, but the joint distribution does not have product-form. Kopzon, Nazarathy, and Weiss [31] and Nazarathy and Weiss [34] determine policies for push-pull networks that ensure that the networks are working at full utilization.

Chen and Mandelbaum [13] conduct a bottleneck analysis of a dynamic, discrete-flow network, where customers are indistinguishable. They use a fluid approximation to identify the system throughput, and show that calculating equilibrium throughput rates is equivalent to identifying the bottlenecks of the original network. Unlike our work, in their network, servers are dedicated to a single class. We will find that allowing the servers to be flexible considerably complicates the analysis, as it is difficult to precisely control the amount of time a server spends at each class. A diffusion approximation for the fluid model in Chen and Mandelbaum [13] is described by the same authors in [14]. Andradóttir, Ayhan, and Down [6] identify a tight upper bound on the capacity, while maintaining stability, and provide a method to construct server assignment policies with performance arbitrarily close to this bound. By contrast, our paper does not require the system to be stable, which also significantly complicates the analysis. Note that if the class of a customer determines the server (that is if only one server is allowed per class) and the servers are not allowed to move, then our problem reduces to production scheduling of classes at each node.

Overloaded systems have also been considered in nonstandard queueing networks where the service rates at individual classes are not independent, but depend deterministically on the state of the entire system. In such a network, Jonckheere, van der Mei, and van der Weij [29] obtain necessary conditions for rate stability at each class, and also provide bounds for the output rate at each class. Similarly, for bandwidth sharing networks, Egorova, Borst, and Zwart [20] give a partial characterization of the overloaded system's behavior by providing a fixed-point equation for the asymptotic growth rates of the queue lengths. For an overloaded switched network, Shah and Wischik analyze a fluid model in [37], and show that the system converges to an invariant state in [38]. Finally, Georgiadis and Tassiulas [22] study the effects of overload in a single commodity network that models information flow.

The use of fluid limits in queueing systems is by now a standard technique. It is known that there is a correspondence between stability of the fluid model and stability of the queueing network in a class of networks considered by Dai [15]. It is also known that if the fluid model is unstable in a strong sense, then the queueing network is unstable in the sense that the total number of customers in the queueing network diverges (Dai [16]). However, additional analysis is required to address how the network becomes unstable.

The organization of this paper is as follows. Section 2 describes our queueing network model and assumptions. In Section 3, we construct a linear program (LP) that is used to

identify the optimal allocation of servers to classes, as well as the resulting throughput, and provide a uniqueness result for the sets of stable or unstable classes. Section 4 introduces two server allocation policies that can achieve any throughput less than the optimal value (with proofs in Appendix A). In Section 5, the concepts of "saturation" input and maximum output are introduced, as well as modified linear programs to calculate those quantities. Section 6 gives numerical results that show how the assignments are determined for a specific network, and provides information about the sensitivity of the optimal assignment to the demand, as well as some simulation results. Finally, Section 7 summarizes our findings.

2 Queueing Network Model

We consider a multiclass network similar to that of Kelly and Laws [30]. It is a classical queueing network generalized to have flexible servers, and is broad enough to cover a wide range of application fields, including manufacturing systems, service systems, and computer systems. Our model facilitates the translation of our stability results (obtained via fluid approximation) into implementable policies for the original system. As a result, it is somewhat less general than the stochastic processing networks considered by Harrison [25, 27, 28].

More specifically, we consider a network of M servers and K classes, with a buffer of infinite size for each class. The class of a customer represents its current processing stage and customers can change class after each stage. The classes may all be at separate physical stations or there may be several classes served at a particular station. The network is supplied by an exogenous arrival process with independent and identically distributed (i.i.d.) increments $u(n)$ for the n^{th} customer with $E(u(1)) = 1/\lambda$. An external arrival is routed to class k with probability $p_{0,k}$, for $k = 1, \ldots, K$. Let the resulting interarrival time of the nth customer at class k be denoted by $u_k(n)$. We allow $p_{0,k} = 0$ for some k, meaning that the external arrival process for customers to class k is null. The arrival rate to class k is denoted by $\lambda_k = \lambda p_{0,k}$. Our results in Sections 3 and 4 are easily extended to the case where some classes have independent arrival streams with i.i.d. interarrival times and rates λ_k at class k.

Upon completion of service, a class i customer becomes one of class k with probability $p_{i,k}$, and leaves the system with probability $p_{i,0} = 1 - \sum_{k=1}^{K} p_{i,k}$ for $i, k = 1, \ldots, K$. Let the routing matrix P have (i, k) entry $p_{i,k}$ for $i, k = 1, \ldots, K$. We assume that the n-step transition matrix P^n satisfies $P^n \to 0$ as $n \to \infty$, which implies that the network is open and $(I - P)^{-1}$ exists and is nonnegative.

The servers are assumed to be flexible, with each server being capable of serving a set of classes. If server j is capable of serving class k and service is in parallel and nonpreemptive, then the n^{th} customer served by server j at class k has a service time given by $v_{j,k}(n)$. Hence

the service rate at a class can depend on both the server and the class being served. We assume that the sequence $\{v_{j,k}(n)\}$ is i.i.d. for each $j = 1, \ldots, M$ and $k = 1, \ldots, K$. The mean service time is given by $m_{j,k} = E(v_{j,k}(1))$ for server j at class k, with corresponding service rate $\mu_{j,k} = 1/m_{j,k}$. If server j is not capable of serving class k, we set $v_{j,k}(n) = \infty$ and $\mu_{j,k} = 0$. Within a class, service is First Come First Served (FCFS). If there are multiple servers available for service at a given class, they can be simultaneously working on different customers, and any one of them can be assigned to a particular customer. Moving server j from class i to class k the nth time incurs a switching time $\xi^j_{i,k}(n)$, $i, k = 1, \ldots, K$, $j = 1, \ldots, M$. We assume that the sequence $\{\xi^j_{i,k}(n)\}$ is i.i.d. for each $i, k = 1, \ldots, K$, $j = 1, \ldots, M$ with mean $s^j_{i,k} = E(\xi^j_{i,k}(1))$. The interarrival, service, and switchover times (i.e., $u(n)$, $v_{j,k}(n)$, and $\xi^j_{i,k}(n)$ for all i, j, k, and n) are assumed to be mutually independent.

Next we define cumulative processes. The total number of exogenous arrivals at time t is given by $E_0(t)$. The processes $A = \{A(t), t \geq 0\}$, $E = \{E(t), t \geq 0\}$, and $D = \{D(t), t \geq 0\}$ are K-dimensional column vectors with $A_k(t)$ denoting the cumulative number of class k customers that arrive in $(0, t]$, $E_k(t)$ being the number of exogenous arrivals to class k in $(0, t]$, and $D_k(t)$ being the number of departures from class k in $(0, t]$. The variable $\Phi_{i,k}(n) = \sum_{l=1}^{n} \phi_{i,k}(l)$, $i = 1, \ldots, K$, $k = 0, \ldots, K$ is the number of customers that arrive to class k from class i among the first n customers passing through class i (the k=0 case corresponds to departures from the system), and $\phi_{i,k}(n)$ are multi-Bernoulli random variables; i.e., the vectors $(\phi_{i,0}(n), \phi_{i,1}(n), \ldots, \phi_{i,k}(n))$ are independent, and for each i, n, exactly one $\phi_{i,k}(n)$ is equal to 1 with probability $p_{i,k}$, for $k = 0, \ldots, K$, and the remainder are zero (meaning that the n^{th} customer from class i is routed to class k). Moreover, $V_{j,k}(t)$ is the residual service time for class k by server j at time t (set to infinity if $\mu_{j,k} = 0$) and $U(t)$, $U_k(t)$ are the residual exogenous interarrival time at time t to the system and to class k, respectively. Let $T_{j,k}(t)$ be the total amount of time that server j spends serving class k customers in $(0, t]$ and $S_{j,k}(t)$ be the potential number of service completions by server j at class k if server j devotes all its time to class k in $(0, t]$. Finally, let $W^j_{i,k}(n)$ denote the total time spent by server j on switching from class i to class k up to and including the nth switch.

Expressing the cumulative processes in terms of the interarrival, service, and switching times $u_k(n)$, $v_{j,k}(n)$, and $\xi^j_{i,k}(n)$, we have

$$S_{j,k}(t) = \max\{n : V_{j,k}(0) + v_{j,k}(1) + v_{j,k}(2) + \cdots + v_{j,k}(n-1) \leq t\}; \quad (1)$$

$$E_0(t) = \max\{n : U(0) + u(1) + u(2) + \cdots + u(n-1) \leq t\}; \quad (2)$$

$$E_k(t) = \max\{n : U_k(0) + u_k(1) + u_k(2) + \cdots + u_k(n-1) \leq t\}; \quad (3)$$

$$W^j_{i,k}(n) = \sum_{m=1}^{n} \xi^j_{i,k}(m). \quad (4)$$

By the Strong Law of Large Numbers (SLLN), we have,

$$\lim_{t\to\infty}\frac{E_k(t)}{t}=\lambda_k,\ \lim_{t\to\infty}\frac{S_{j,k}(t)}{t}=\mu_{j,k},\text{ and }\lim_{n\to\infty}\frac{W^j_{i,k}(n)}{n}=s^j_{i,k},\quad\text{for } j=1,\ldots,M,\text{ and } i,k=1,\ldots,K. \tag{5}$$

Finally, we assume that the interarrival times are unbounded and spread out. That is, there exists some integer l, and some function $q(x)\geq 0$ on $\mathbb{R}_+$ with $\int_0^\infty q(x)dx>0$, such that

$$P(u(1)\geq x) \;>\; 0,\ \text{for any } x>0, \tag{6}$$

$$P(a\leq u(1)+\ldots+u(l)\leq b) \;\geq\; \int_a^b q(x)dx,\ \text{for any } 0\leq a<b. \tag{7}$$

This assumption is required for Theorem 4.2 in Dai [15], which we will use in Appendix A.

Let the queue length at class k at time t be denoted by $Q_k(t)$. For a given server assignment policy (i.e., the functions $T_{j,k}(t)$ are given for all j and k), the cumulative variables satisfy the following queueing network equations

$$A_k(t) \;=\; E_k(t)+\sum_{i=1}^{K}\Phi_{i,k}(D_i(t)),\ k=1,\ldots,K; \tag{8}$$

$$D_k(t) \;=\; \sum_{j=1}^{M}S_{j,k}(T_{j,k}(t)),\ k=1,\ldots,K; \tag{9}$$

$$Q_k(t) \;=\; Q_k(0)+A_k(t)-D_k(t),\ k=1,\ldots,K; \tag{10}$$

and $0\leq\sum_{k=1}^{K}T_{j,k}(t)\leq t,\ j=1,\ldots,M$. Finally, let $D(t)=\sum_{k=1}^{K}\Phi_{k,0}(D_k(t))$ be the number of departures from the system by time t. Then the throughput of the system is given by $\limsup_{t\to\infty}D(t)/t$. Note that the switching times do not appear explicitly in equations (8) through (10), but they are incorporated implicitly through the functions $\{T_{j,k}(t)\}$.

3 Deterministic Analysis

Deterministic analysis (such as linear programs based on a fluid approximation) has been a very useful tool for addressing production, resource, and input planning problems in stochastic processing networks. Constructing a policy to achieve a given objective based on a simple LP is also known as the "static planning problem," introduced by Harrison in a series of papers [25, 27, 28]. This type of problem has received a lot of attention (see for example Nazarathy and Weiss [33, 34] and Dai and Lin [18] for various problem formulations).

When we allow instability in the system, the calculation of the flow rates is not obvious. In particular, the usual traffic equation for the flow rate at class k (i.e., $r_k=\lambda p_{0,k}+\sum_{i=1}^{K}p_{i,k}r_i$,

where r_k is the effective inflow rate to class k) is not valid, because in our case the input rate to a class does not necessarily equal the output rate from that class. In this section, given the offered demand λ to the system, we construct an optimization problem whose solution provides both the optimal allocation of servers to classes and also the corresponding input and output rates at each class. The allocation of the servers is such that the maximum capacity for the network is achieved for λ, while satisfying network constraints.

The outline of this section is as follows. In Section 3.1 the LP that is used to determine the allocation of servers, is constructed. Section 3.2 introduces a uniqueness result for the effective inflow and outflow from each node in the network given the allocation parameters. Finally, in Section 3.3, we identify the stable and unstable classes based on the allocation LP, and also consider the special case when we have a Jackson network.

3.1 The Allocation LP

In this section, we introduce the allocation LP that will be used for solving the static planning problem. We start by defining the flows within the network. The effective inflow rate a_k to class k consists of inflow from the outside plus the inflow from the other classes within the network. Similarly, d_k is the effective outflow rate from class k. Let $\delta_{j,k}$ be the fraction of time that server j devotes for class k customers. For all $k = 1, \ldots, K$, we have

$$a_k = \lambda p_{0,k} + \sum_{i=1}^{K} d_i p_{i,k}, \tag{11}$$

$$d_k = \min\Big(\sum_{j=1}^{M} \mu_{j,k}\delta_{j,k}, a_k\Big). \tag{12}$$

Next, we maximize the throughput over the decision variables $d_k \geq 0$ and $\delta_{j,k} \geq 0$, for $j = 1, \ldots, M$ and $k = 1, \ldots, K$, using the following allocation LP:

$$\max \sum_{k=1}^{K} d_k p_{k,0} \text{ such that} \tag{13}$$

$$d_k \leq \sum_{j=1}^{M} \mu_{j,k}\delta_{j,k},\ k = 1, \ldots, K; \tag{14}$$

$$d_k \leq \lambda p_{0,k} + \sum_{i=1}^{K} d_i p_{i,k},\ k = 1, \ldots, K; \tag{15}$$

$$\sum_{k=1}^{K} \delta_{j,k} \leq 1,\ j = 1, \ldots, M; \tag{16}$$

$$d_k \geq 0, \delta_{j,k} \geq 0,\ j = 1, \ldots, M,\ k = 1, \ldots, K. \tag{17}$$

Our objective in this LP is to allocate the servers to the classes so that the output from the system is maximized. The right-hand side of the first constraint (14) is the total amount of service effort allocated to class k and the left-hand side is the long-run departure rate from class k. So (14) simply means that the departure rate from a class k cannot exceed the service allocation to that class. Similarly, the right-hand side of constraint (15) is the long-run arrival rate to class k. So this constraint means that the long-run departure rate from a class can not exceed the long-run arrival rate to that class. The constraint (16) prevents us from overallocating a server, and (17) prevents negative allocations.

Let $\delta^*_{j,k} \geq 0$ and $d^*_k \geq 0$ for all j, k be an optimal solution to the above LP for the demand λ. Let $\mu^*(\lambda) = \sum_{k=1}^{K} d^*_k p_{k,0}$ be the optimal value of the LP corresponding to λ. Clearly, $(d^*_1, \ldots, d^*_K)$ is an optimal solution to the above LP if and only if $(d^*_1, \ldots, d^*_K)$ satisfy equations $(11) - (12)$ with $\delta_{j,k} = \delta^*_{j,k}$, for all j, k. Consequently, one can obtain a solution to $(11) - (12)$ under the optimal allocation $\delta^*_{j,k}$, for all j, k, by solving the LP. The solution to the allocation LP provides an upper bound on the maximum achievable throughput, and we will see that we can get arbitrarily close to this value. The following theorem states this fact; a proof will be given in Appendix A. Policies that achieve throughput arbitrarily close to the optimum value of the allocation LP will be described in Sections 4.1 and 4.2.

Theorem 3.1. *(a) Any throughput less than $\mu^*(\lambda)$ can be achieved, where $\mu^*(\lambda)$ is the optimal value of the allocation LP $(13) - (17)$ for the offered demand λ. That is, for any given λ and $0 < \epsilon < 1$, there exists a policy π with throughput μ^π such that $\mu^\pi \geq (1 - \epsilon)\mu^*(\lambda)$.*

(b) A throughput larger than $\mu^*(\lambda)$ cannot be achieved by any policy.

We also have a result on the behavior of the optimal objective function value $\mu^*(\lambda)$ as a function of λ. This result is a corollary of Theorem 5.1 in Bertsimas and Tsitsiklis [11].

Lemma 3.1. *The optimal objective function value $\mu^*(\lambda)$ obtained from the allocation LP (13)-(17) is a continuous, non-decreasing, piece-wise linear, and concave function of λ.*

Proof. The fact that $\mu^*(\lambda)$ is non-decreasing as we increase λ is obvious, since by increasing λ, we are increasing the feasible set. The concavity and linearity of $\mu^*(\lambda)$ follows from Theorem 5.1 in [11]. Finally, the continuity follows from the concavity. □

3.2 Uniqueness

In this section, we discuss the uniqueness of the solution (a^*_k, d^*_k), where $k = 1, \ldots, K$, of equations $(11) - (12)$ given allocations $\delta^*_{j,k},\ j = 1, \ldots, M,\ k = 1, \ldots, K$. Lemma 3.2 of Chen

and Mandelbaum [13] shows that the a_k^*, d_k^* are unique. Tekin [41] discusses how this result follows from converting $(11)-(12)$ to a linear complementarity problem.

However, non-unique allocations may lead to non-unique (a_k^*, d_k^*) values. For instance, consider a network with three classes, external input only to class 1, and each customer equally likely to go to class 2 or class 3 from class 1, after which they exit the system. We have two servers with $\mu_{j,k}$, $j = 1, 2,\ k = 1, 2, 3$, values given by the (j, k) entry in the matrix

$$H = \begin{pmatrix} 5 & 0 & 0 \\ 0 & 2 & 2 \end{pmatrix}.$$

Let $\lambda = 6$. Then, based on the solution of the allocation LP, $\mu^*(\lambda)$ is 2 and can be achieved through different assignments, each resulting in different (a_k^*, d_k^*) values. For instance, let the $M \times K$ matrix T^* have (j, k) entry $\delta_{j,k}^*$, for all j, k. Consider the following two assignments:

$$T_1^* = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix},\ T_2^* = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

For both assignments, $\mu^*(\lambda)$ is 2. Then, for the first assignments we have $a_2^* = a_3^* = 2.5$ and $d_2^* = 2$, $d_3^* = 0$; however for T_2^*, we have $a_2^* = a_3^* = 2.5$ and $d_2^* = 0$, $d_3^* = 2$.

3.3 Classification of the Nodes

In this section, we introduce stable and unstable sets of nodes based on the solution of the LP. This classification will be used later to construct server allocation policies. In particular, we separate the nodes into two sets as follows:

$$S = \{k : a_k^* = d_k^*\}, \tag{18}$$

$$U = \{k : a_k^* > d_k^*\}. \tag{19}$$

Since there is a unique solution for $(11)-(12)$, see Section 3.2, the sets S and U are uniquely determined given the allocations $\{\delta_{j,k}^*\}$. The sets S and U specify the classes that are stable and unstable, respectively, in the solution of the allocation LP, where a class is defined to be stable if the departure rate from the class equals the arrival rate. Note that the unstable classes U cannot simply be determined by comparing the solution of the regular balance equations $\{r_k\}$ with the effective processing rates at each station; i.e., U is in general different from $\{k : r_k \geq \mu_k^*\}$, where $r_k = \lambda p_{0,k} + \sum_{i=1}^K r_i p_{i,k}$ and $\mu_k^* = \sum_{j=1}^M \mu_{j,k}\delta_{j,k}^*$ for all k.

For example, consider the network shown in Figure 1, where all customers arrive to class 1 and each customer is equally likely to either depart or be routed to the other class from

each class 1, 2; see also the routing matrix P. Suppose that we have three servers and that the service rates for each class are given in the matrix H, where the (j,k) entry is $\mu_{j,k}$:

$$P = \begin{pmatrix} 0 & 0.5 \\ 0.5 & 0 \end{pmatrix}, H = \begin{pmatrix} 6 & 2 \\ 5 & 1 \\ 4 & 0 \end{pmatrix}. \tag{20}$$

Looking at the service rates $\mu_{j,k}$ in H, the best assignment of the servers to the classes is not obvious. Since the effective arrival and departure rates at the classes depend on these allocations, identifying the unstable classes from the matrix H by inspection is also not obvious. So we resort to the allocation LP $(13)-(17)$. When $\lambda = 6$, the optimum objective function value $(d_1^*/2 + d_2^*/2)$ is given by $\mu^*(6) \simeq 4.7727$ and the assignments are as follows

$$T^* \simeq \begin{pmatrix} 0 & 1 \\ 0.6364 & 0.3636 \\ 1 & 0 \end{pmatrix}. \tag{21}$$

According to these results, we see that the effective processing capacities, departure, and arrival rates at each class $k = 1, \ldots, K$ are given by

$$\mu^* \simeq [7.1818, 2.3636]',\ d^* \simeq [7.1818, 2.3636]',\ a^* \simeq [7.1818, 3.5909]',$$

where $\mu^* = [\mu_1^*, \ldots, \mu_K^*]'$, $d^* = [d_1^*, \ldots, d_K^*]'$ and $a^* = [a_1^*, \ldots, a_K^*]'$. If we solve the regular balance equations, we obtain $r_1 = 8$ and $r_2 = 4$, so that $\{k : r_k \geq \mu_k^*\} = \{1, 2\}$. However, according to our algorithm, the only unstable class in the solution of the allocation LP is class 2, because we have $a_1^* = d_1^*$ and $a_2^* > d_2^*$, so that $U = \{2\}$ and $S = \{1\}$.

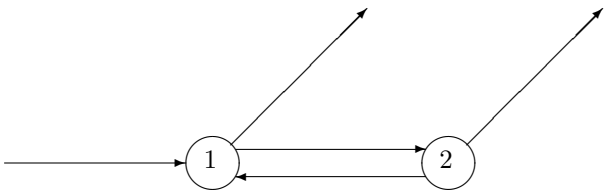


Figure 1: A two-class network

As shown before, we cannot simply determine stable and unstable nodes by inspection, and Jackson networks are no exception. Goodman and Massey [23] identify the maximal subnetwork that achieves steady state in a non-ergodic Jackson network. However, the allocation LP can also determine the stable and unstable sets of nodes for Jackson networks with the servers constrained to choose only one class to serve (i.e., $\delta_{j,k} \in \{0, 1\}$, for all j

and k). Unlike the algorithm suggested by Goodman and Massey [23] to find the stable and unstable sets, our LP not only provides a classification of the nodes but also suggests an optimal server allocation plan that maximizes throughput. If the server allocation is predetermined (i.e., the $\delta^*_{j,k}$ are given), then the stable and unstable sets defined as $S' = \{k : a^*_k < \mu^*_k\}$ and $U' = \{k : a^*_k \geq \mu^*_k\}$ coincide with those determined in Goodman and Massey [23]. The difference between S, U and S', U' reflects the different definition of stability considered in [23]. In particular, as we are considering rate stability, nodes with $a^*_k = \mu^*_k = d^*_k$ are considered stable in our case. Moreover, an invariant distribution exists for the stable set of classes as shown by Goodman and Massey [23].

Note that a policy π whose throughput μ^π is arbitrarily close to the optimum throughput $\mu^*(\lambda)$ (i.e., $\mu^\pi \geq \mu^*(\lambda) - \epsilon$, where $\epsilon > 0$ is small) does not necessarily have the same sets of stable and unstable classes as determined by the allocation LP. For instance, consider a network with two classes and demand $\lambda = 1$, where each job is equally likely to go to class 1 or 2, from which they exit the system. We have one flexible server with $(\mu_{1,1}, \mu_{1,2}) = (1, 0.5)$. Then, the unique optimal allocations are given by $\delta^*_{1,1} = 1/2$ and $\delta^*_{1,2} = 1/2$ with $\mu^*(\lambda) = 0.75$. Hence the sets S and U are uniquely determined by $\{1\}$ and $\{2\}$, respectively.

Next we consider three near-optimal allocations that yield different stable and unstable sets. First, for any $0 < \epsilon < 1$, let $(\delta^{(1)}_{1,1}, \delta^{(1)}_{1,2}) = ((1-\epsilon)/2, 1/2)$, so that $S^{(1)} = \emptyset$, $U^{(1)} = \{1, 2\}$, and $\mu^{(1)} = \mu^*(\lambda) - \epsilon/2 > \mu^*(\lambda) - \epsilon$. Secondly, for any $0 < \epsilon < 1$, let $(\delta^{(2)}_{1,1}, \delta^{(2)}_{1,2}) = (1/2, (1 - \epsilon)/2)$, so that $S^{(2)} = \{1\}$, $U^{(2)} = \{2\}$, and $\mu^{(2)} = \mu^*(\lambda) - \epsilon/4 > \mu^*(\lambda) - \epsilon$. Finally, consider the assignment $(\delta^{(3)}_{1,1}, \delta^{(3)}_{1,2}) = (0, 1)$, then we have $\mu^{(3)} = 0.5 \geq \mu^*(\lambda) - \epsilon$ for $\epsilon \geq 0.25$, and $S^{(3)} = \{2\}$, $U^{(3)} = \{1\}$. We observe that even if the allocation LP has unique set classifications, we can construct policies based on ϵ with different stable and unstable sets. Hence, we conclude that stability of a class according to the LP does not imply its stability for a near-optimal policy, and vice versa. Returning to the example, if we want to get arbitrarily close to $\mu^*(\lambda)$ with a small enough ϵ, then only policies 1 and 2 are valid, because the last one violates this requirement. This suggests that as we get closer to the optimum allocations (i.e., $\epsilon \to 0$), the set of unstable classes under a near-optimal policy will contain the set of unstable classes of the allocation LP.

There appear to be connections between the unstable set of classes obtained as a result of a given policy and the infinite virtual queues discussed in Kopzon, Nazarathy, and Weiss [31], Nazarathy, and Weiss [33, 34], and Weiss [43]. The nodes with infinite virtual queues have an infinite supply of work, and are comparable to the unstable nodes in our case. However, the infinite virtual queues are considered stable, because they are always nonempty with a finite number of customers. Also, in our case the set of unstable nodes depends on the selected policy, whereas the infinite virtual queues in [31, 33, 34, 43] are predetermined.

4 Optimum Server Allocation

In this section, we develop two server allocation algorithms that achieve throughput that is arbitrarily close to the optimum value of the allocation LP (13) − (17). The analysis is complicated by the observation in the previous section that for a policy π, the sets of stable and unstable classes may not correspond to those given by the LP. This makes it difficult to determine the proportion of time spent by a server at each class under π. To ensure that these proportions are sufficiently close to the allocations obtained by the solution of the allocation LP, we propose two approaches. The first, described in Section 4.1, involves admission control and controlled routing. The second approach, described in Section 4.2, involves forced idling of servers at certain classes. In Appendix A we prove that the algorithms provided in Sections 4.1 and 4.2 can be used to obtain throughput that is arbitrarily close to the maximum output $\mu^*(\lambda)$ given the available demand λ.

4.1 Server Allocation Policy with Admission and Routing Control

In this section, an algorithm for assigning servers to classes is presented based on the allocation LP introduced in Section 3.1. In particular, suppose that we are given a certain λ (level of offered demand to the system) and are asked to maximize the throughput without regard to stability. Let $\{\delta^*_{j,k}\}$ be the optimal assignment fractions given by the solution to the allocation LP (13) − (17), and let $\mu^*(\lambda) = \sum_{k=1}^{K} d^*_k p_{k,0}$ be the resulting optimum throughput. Our aim is to assign servers to classes based on the fractions $\{\delta^*_{j,k}\}$ to achieve throughput as close to $\mu^*(\lambda)$ as desired. For this, a generalized round robin server assignment policy is considered, together with admission control and controlled routing. More specifically, we reject arrivals to the system with a small probability, and also modify the routing probabilities $p_{i,k}$, for all i, k, so that the arrival rate to the classes $k \in U$ is reduced to d^*_k and excess input is rerouted to an imaginary class $K+1$ served by an imaginary server $M+1$. In practice, the customers routed to class $K+1$ would be scrapped, but the addition of this imaginary class facilitates differentiation between successful completions and scrapped customers. This approach not only guarantees a target throughput, but also stabilizes the classes in the network by scrapping just enough customers at certain classes in the network.

The following result is Proposition 3 of Andradóttir, Ayhan, and Down [6]. We use it to show that for any allocation of servers to classes, a generalized round robin server assignment policy exists that gets arbitrarily close to that allocation.

Proposition 4.1. *Let κ be a finite set, and for each $k \in \kappa$, suppose that m_k and δ_k satisfy $0 < m_k < \infty$, $\delta_k \geq 0$, and $0 \leq \sum_{k\in\kappa} \delta_k \leq 1$. Suppose furthermore that $0 \leq s < \infty$. Then for*

any $0 < \epsilon \leq 1$*, there exists a set of non-negative integers* $\{l_k\}$*, where* $k \in \kappa$*, such that*

$$\frac{l_k m_k}{s + \sum_{i \in \kappa} l_i m_i} \geq \delta_k(1-\epsilon) \text{ for all } k \in \kappa. \tag{22}$$

Let $1\{\cdot\}$ denote the indicator function. Then one possible choice for l_k is

$$l_k = \left\lceil \frac{(1-\epsilon)(s + \sum_{i \in \kappa} m_i 1\{\delta_i > 0\})\delta_k}{\epsilon m_k} \right\rceil. \tag{23}$$

Consider a specific generalized round robin server assignment policy π that has each server j serving a fixed list V_j^π of classes in a cyclic order. For each class $k \in V_j^\pi$, server j serves a maximum of $l_{j,k}^\pi$ customers and then moves to the next class on the list for service, but if the queue for class k empties before $l_{j,k}^\pi$ service completions, the server moves on to the next class on its list. If there are no more customers in any of the classes on the list, then the server idles until an arrival to any class on the list. We now state how to choose the parameters V_j^π and $l_{j,k}^\pi$ of our generalized round robin server assignment policy π, assuming that the offered demand to the system is λ. Note that the choice of $l_{j,k}^\pi$ determines the lot sizes that server j should process at each visit to class k, and hence impacts the efficiency of the policy. These values need to be chosen sufficiently large to mitigate the effects of switching times (see Section 3.2 of Andradóttir, Ayhan, and Down [6] for various strategies in choosing $\{l_{j,k}^\pi\}$). In the following algorithm, we are primarily interested in the behavior of the network when $\lambda > \lambda^*$ (i.e., when $U \neq \emptyset$), where λ^* is the maximum offered demand such that the system can be stabilized for $\lambda < \lambda^*$. The case $\lambda < \lambda^*$ is already covered in [6], where it is shown that λ^* can be computed by solving an appropriate LP.

1. Solve the allocation LP (13) − (17).

2. Choose $0 < \epsilon < 1$.

3. Admission Control: Thin the arrival process by rejecting arrivals with probability ϵ and accepting them with probability $1 - \epsilon$, so that the arrival rate reduces to $\lambda' = \lambda(1-\epsilon)$.

   Controlled Routing: Introduce an imaginary scrapping class $K+1$ with an associated dedicated server $M+1$ such that $\mu_{M+1,K+1} = \lambda$ and $\delta^*_{M+1,K+1} = 1$. Replace the routing probabilities $p_{i,k}$, where $0 \leq i, k \leq K$ by the following routing probabilities $\bar{p}_{i,k}$, $0 \leq i, k \leq K$. For $0 \leq i \leq K$, let $\bar{p}_{i,k} = p_{i,k}$ for $k \in S$; $\bar{p}_{i,k} = p_{i,k}\epsilon_k$ for $k \in U$, where $\epsilon_k = d_k^*/a_k^*$; $\bar{p}_{i,K+1} = \sum_{k \in U} p_{i,k}(1-\epsilon_k)$; and $\bar{p}_{K+1,0} = 1$, $\bar{p}_{K+1,K+1} = 0$. For $1 \leq k \leq K$, $\bar{p}_{k,0} = p_{k,0}$ and $\bar{p}_{K+1,k} = 0$.

4. For each server j, specify the ordered list V_j^π using all of the classes k with $\mu_{j,k}\delta_{j,k}^* > 0$. Define the ith element of each list V_j^π as $v_{j,i}$ and let $|\cdot|$ denote cardinality of a set.

5. For each server j with $|V_j^\pi| > 1$, let s_j^π be the expected switching time in a cycle of visiting the states in V_j^π in order, so that

$$s_j^\pi = \sum_{i=1}^{|V_j^\pi|-1} s^j_{v_{j,i},v_{j,i+1}} + s^j_{v_{j,|V_j^\pi|},v_{j,1}}.$$

6. For each server j with $|V_j^\pi| > 1$ and each class $k \in V_j^\pi$, calculate parameters $l_{j,k}^\pi$ satisfying $l_{j,k}^\pi m_{j,k}/(s_j^\pi + \sum_{i\in V_j^\pi} l_{j,i}^\pi m_{j,i}) \geq \delta_{j,k}^*(1-\epsilon')$, where $\epsilon' = \epsilon/(2-\epsilon)$, see Proposition 4.1 and equation (23).

7. For each server j with $|V_j^\pi| = 1$, set $s_j^\pi = 0$ and $l_{j,k}^\pi = 1$ for $k \in V_j^\pi$.

8. For each server j and all classes $k \notin V_j^\pi$, let $l_{j,k}^\pi = 0$.

As a result of ignoring stability in the allocation LP $(13)-(17)$, it is possible to have queue lengths $\{Q_k(t)\}$ at certain classes k diverge as $t \to \infty$, without the controlled routing. The following theorem shows that the above generalized round robin server assignment policy π with admission control and controlled routing yields throughput μ^π that comes arbitrarily close to achieving the desired throughput level of $\mu^*(\lambda)$, and also stabilizes the original queueing network. The proof of Theorem 4.1 is postponed until Appendix A.

Theorem 4.1. *A policy constructed using the above algorithm achieves throughput* $\mu^\pi = (1-\epsilon)\mu^*(\lambda)$. *Moreover, the distribution of the queue length process* $\{Q(t)\}$ *converges to a steady state distribution as* $t \to \infty$.

It immediately follows from Theorem 4.1 that the choice $\epsilon = 1 - \mu/\mu^*(\lambda)$ will guarantee that we achieve a target throughput $\mu < \mu^*(\lambda)$ (i.e., $\mu^\pi = \mu$).

4.2 Server Allocation Policy with Forced Server Idling

In this section, we introduce an alternative generalized round robin policy without admission control or controlled routing. Since we allow instability, each server j will eventually always find more than the required number of customers $l_{j,k}$ at unstable classes k, and hence spend the maximum amount of time allowed during each of its cycles at such classes in its list. However, this could result in problems, because although the fractions of time servers spend at unstable classes are guaranteed to achieve certain minimums (see Proposition 4.1), we do not control how big they can be. Since there are always customers to process at unstable

classes, it becomes possible for a server assigned to an unstable class to spend more time than required there, resulting in the flows of customers between stations in the network not being sufficiently close to the optimal flows identified by the allocation LP $(13)-(17)$. To prevent this, we force the servers to spend the required amount of time at each of the classes in their lists, even if it means idling them. Unlike the approach in Section 4.1 where servers complete a fixed number of customers before switching, we will construct a timed round robin policy where servers spend fixed amounts of time at the classes on their lists. Consequently, it is possible that a server will leave a customer whose service is in progress. The residual work may be completed by another server, and hence we assume that the service time distributions are independent of the server and that no service effort is lost. We will represent the service requirement of customer n at class k by $v_k(n)$, and server j reduces this requirement at a rate $\mu_{j,k}$ when assigned to class k. This model is appropriate when service is preemptive and/or cooperative.

Consider a specific policy π that has each server j serving a fixed list V_j^π of classes in a cyclic order as in Section 4.1. For each class $k \in V_j^\pi$, server j spends a fixed amount of time $h_{j,k}^\pi$ at class k, even if the queue for class k empties before that time, and then server j moves to the next class on its list. We make use of Proposition 4.1 to determine $h_{j,k}^\pi$, for all j, k. Although the following algorithm works for any value of λ, we are primarily interested in the behavior of the network when $\lambda > \lambda^*$. Next, we state how to choose the parameters V_j^π and $h_{j,k}^\pi$ of our generalized round robin server assignment policy π, assuming that the offered demand to the system is λ. In particular, we will use the eight-step policy of Section 4.1, except that steps 3, 6, and 8 of that policy are replaced by the steps below:

3. For all the servers $j = 1, \ldots, M$, let
$$\kappa_j = 1 - \sum_{k=1}^{K} \delta_{j,k}^* 1\{\mu_{j,k} > 0\}.$$

6. For each server j with $|V_j^\pi| > 1$ and each class $k \in V_j^\pi$, set $\bar{\delta}_{j,k}^* = \delta_{j,k}^* + \kappa_j/|V_j^\pi|$, for all $k \in V_j^\pi$, and calculate parameters $l_{j,k}^\pi$ satisfying $l_{j,k}^\pi m_{j,k}/(s_j^\pi + \sum_{i \in V_j^\pi} l_{j,i}^\pi m_{j,i}) \geq \bar{\delta}_{j,k}^*(1-\epsilon)$, see Proposition 4.1 and (23).

8. For each server j, set $h_{j,k}^\pi = l_{j,k}^\pi m_{j,k}$, for $k \in V_j^\pi$, and $h_{j,k}^\pi = 0$, for $k \notin V_j^\pi$.

Theorem 4.2. *A policy constructed using the above algorithm achieves the throughput* $\mu^\pi \geq (1-\epsilon)\mu^*(\lambda)$.

It immediately follows from Theorem 4.2 that $\epsilon = 1 - \mu/\mu^*(\lambda)$ guarantees that we achieve a target throughput $\mu < \mu^*(\lambda)$.

5 The Saturation Input and Maximum Output

Even if we allow some classes in the network to be unstable, the output from the network does not necessarily increase with the demand λ. We refer to the point $\bar{\lambda}$ where increasing the demand has no effect on the best possible output as the "saturation" input to the system, and we let $\bar{\mu}$ denote the corresponding maximum output. In this section, we discuss how to identify $\bar{\lambda}$ and $\bar{\mu}$. This information determines the limitations for our system. We also show how to determine the minimum demand required for a target output level of $\mu \leq \bar{\mu}$.

To determine $\bar{\mu}$, we use the allocation LP (13) – (17) with $\lambda = \infty$:

$$\max \sum_{k=1}^{K} d_k p_{k,0} \text{ such that}$$

$$\begin{aligned}
d_k &\leq \sum_{j=1}^{M} \mu_{j,k}\delta_{j,k}, \ k = 1, \ldots, K; && (24)\\
d_k &\leq \sum_{i=1}^{K} d_i p_{i,k}, \ \forall k : p_{0,k} = 0; && (25)\\
\sum_{k=1}^{K} \delta_{j,k} &\leq 1, \ j = 1, \ldots, M;\\
d_k &\geq 0, \ \delta_{j,k} \geq 0, \ j = 1, \ldots, M, \ k = 1, \ldots, K. && (26)
\end{aligned}$$

The following theorem shows that the solution of this LP over $\delta_{j,k} \geq 0$ and $d_k \geq 0$ for all j, k allows us to identify the maximum output $\bar{\mu}$ and an upper bound on the saturation input $\bar{\lambda}$.

Theorem 5.1. *(a) Let $\bar{\mu} = \sum_{k=1}^{K} d_k^* p_{k,0}$ be the optimal value for the allocation LP (24) – (26) and*

$$\hat{\lambda} = \max_{k:p_{0,k}>0} \left\{ \frac{d_k^* - \sum_{i=1}^{K} d_i^* p_{i,k}}{p_{0,k}} \right\}. \tag{27}$$

Then we have $\bar{\lambda} \leq \hat{\lambda}$ and $\mu^*(\lambda) = \bar{\mu}$, for all $\lambda \geq \hat{\lambda}$. That is, even if the arrival rate to the original queueing network is increased beyond $\hat{\lambda}$, any capacity larger than $\bar{\mu}$ can not be achieved.

(b) The optimal value $\bar{\mu}$ of the allocation LP (24) – (26) is a tight upper bound on the maximum achievable throughput. That is, any capacity larger than $\bar{\mu}$ cannot be achieved in the original queueing network. Moreover, given a demand $\lambda \geq \hat{\lambda}$, there exists a specific round robin policy π with parameters given by the solution of the LP (24)–(26) and constructed as in Section 4.1 or Section 4.2 with $\mu^\pi \geq \bar{\mu}(1-\epsilon)$, where $0 < \epsilon < 1$.

Proof. The optimum value $\bar{\mu}$ of the allocation LP $(24)-(26)$ is finite, since (24) implies that $d_k \leq \sum_{j=1}^{M} \mu_{j,k}$, $k = 1, \ldots, K$, and $\sum_{k=1}^{K} d_k p_{k,0} \leq \sum_{k=1}^{K} d_k$. Also note that $\delta_{j,k}^*$, d_k^* from the solution of the above LP also satisfy the allocation LP $(13)-(17)$ for any $\lambda \geq \hat{\lambda}$ with an optimum value $\mu^*(\lambda) = \bar{\mu}$, since (15) is automatically satisfied by definition of $\hat{\lambda}$. Together with part (b) of Theorem 3.1, this proves part (a) of the theorem.

By Theorem 3.1, we know that $\bar{\mu}$ is a tight upper bound on the achievable throughput. Moreover, Theorems 4.1 and 4.2 show that a policy π constructed as in Section 4.1 or 4.2 will achieve $\mu^\pi \geq \bar{\mu}(1-\epsilon)$, and part (b) of the theorem follows. □

Next our aim is to show how to determine a policy based on a target throughput and also to show how to find the saturation input $\bar{\lambda}$. Because of the non-uniqueness of optimal solutions, $\hat{\lambda}$ can be different from $\bar{\lambda}$. For instance, consider a network with two stations in tandem, each having exactly one dedicated server with processing rates μ_1 and μ_2, respectively. Suppose furthermore that $\lambda > \mu_1 > \mu_2$. Then $d_1^* = \mu_1$, $d_2^* = \mu_2$ is an optimal solution with $\hat{\lambda} = \mu_1$, but $\bar{\lambda} = \mu_2$. We need this tighter saturation input bound to gain insight into the limitations of our network. For instance, if the actual offered demand to the system is less than the saturation level (i.e., $\lambda < \bar{\lambda}$), then our capacity is underutilized. On the other hand, when $\lambda \geq \bar{\lambda}$, we know that we have excess offered demand. The second benefit is the fact that for $\lambda \geq \bar{\lambda}$, optimal allocations become insensitive to the offered demand λ, so that we do not need to worry about fluctuations in the input process as long as $\lambda \geq \bar{\lambda}$.

Let $\mu \leq \bar{\mu}$ be the target output. We determine the minimum demand $\lambda' \geq \mu$ required so that the target output of μ is feasible. For this, consider the following allocation LP:

$$\min\ \lambda \text{ such that} \tag{28}$$

$$\sum_{k=1}^{K} d_k p_{k,0} \geq \mu; \tag{29}$$

$$d_k \leq \sum_{j=1}^{M} \mu_{j,k}\delta_{j,k},\ k = 1, \ldots, K; \tag{30}$$

$$d_k \leq \lambda p_{0,k} + \sum_{i=1}^{K} d_i p_{i,k},\ k = 1, \ldots, K; \tag{31}$$

$$\sum_{k=1}^{K} \delta_{j,k} \leq 1,\ j = 1, \ldots, M; \tag{32}$$

$$d_k \geq 0,\ \delta_{j,k} \geq 0,\ j = 1, \ldots, M,\ k = 1, \ldots, K. \tag{33}$$

This time our objective is to allocate the servers such that the minimum demand is required while maintaining the desired output. Our decision variables (that we minimize over) are

λ, $d_k \geq 0$, and $\delta_{j,k} \geq 0$ for $j = 1, \ldots, M$, $k = 1, \ldots, K$. The right-hand side of the first constraint (29) is the total amount of output required μ and the left-hand side is the long-run departure rate from the system. So (29) simply means the throughput of the system should be at least μ. All the other constraints in this LP appear in the previous LP $(13) - (17)$ and have the same interpretations. Note that there is no nonnegativity constraint for λ, because in the optimal solution its value will always be nonnegative. To see this, we proceed by contradiction. Suppose $\lambda < 0$ in the optimal solution. Now summing over k in (31) yields $\sum_k d_k \leq \lambda \sum_k p_{0,k} + \sum_i d_i \sum_k p_{i,k} = \lambda + \sum_i d_i(1 - p_{i,0})$, so that $\sum_k d_k < \sum_k d_k(1 - p_{k,0}) \leq \sum_k d_k$, a contradiction. Thus the constraints cannot be satisfied for a negative λ. Let $\delta^*_{j,k} \geq 0$, $d^*_k \geq 0$ and $\lambda^*(\mu)$ for all j, k be an optimal solution to the above LP.

Theorem 5.2. *(a) A policy π constructed as in Section 4.1 or Section 4.2, based on the offered demand $\lambda \geq \lambda^*(\mu)$ and allocations $\delta^*_{j,k}$, for all j, k, obtained from the solution of the allocation LP $(28) - (33)$ comes arbitrarily close to the target throughput μ. That is, the throughput μ^π of π satisfies $\mu^\pi \geq \mu(1 - \epsilon)$, where $0 < \epsilon < 1$.*

(b) We have $\bar{\lambda} = \lambda^*(\bar{\mu})$.

Proof. To simplify the notation, let $\tilde{\lambda} = \lambda^*(\mu)$. Let a policy π be designed as in Section 4.1 or Section 4.2 corresponding to $\tilde{\lambda}$ and ϵ. Then we have by Theorem 4.1 or 4.2 that $\mu^\pi \geq \mu^*(\tilde{\lambda})(1 - \epsilon)$, where $\mu^*(\tilde{\lambda})$ is the solution to the allocation LP $(13) - (17)$. Note that d^*_k and $\{\delta^*_{j,k}\}$ from the LP $(28) - (33)$ also satisfy $(13) - (17)$. The constraint (29) implies that $\mu^*(\tilde{\lambda}) \geq \mu$, and hence that $\mu^\pi \geq \mu(1 - \epsilon)$ as required. Together with Lemma 3.1, this proves part (a) of the theorem, and part (b) follows by the definition of $\bar{\lambda}$ and Theorem 5.1. □

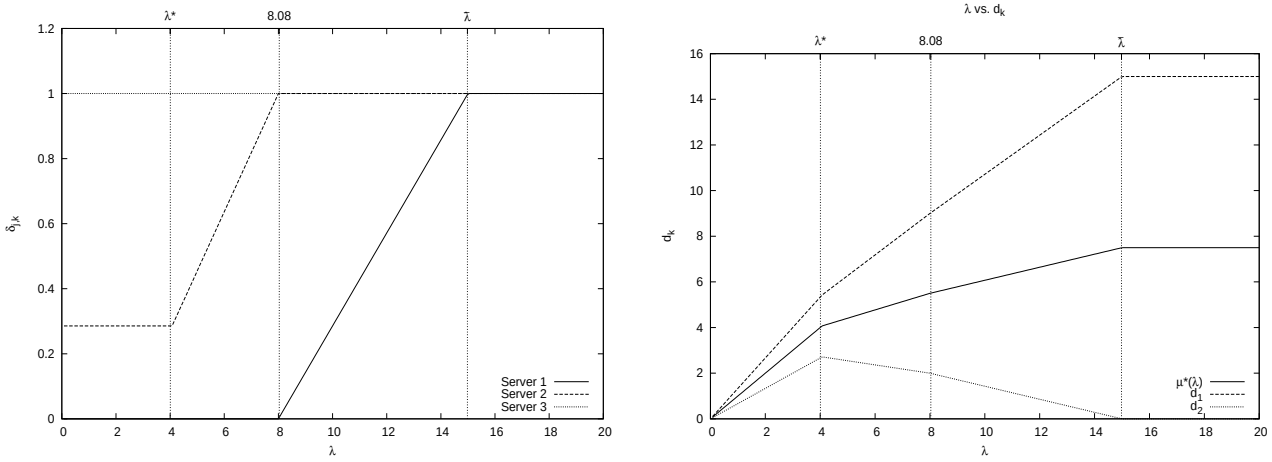
Note that by Lemma 3.1 and the definitions of λ^* and $\bar{\lambda}$, we have $d\mu^*(\lambda)/d\lambda = 1$ for $\lambda < \lambda^*$ and $d\mu^*(\lambda)/d\lambda = 0$ for $\lambda > \bar{\lambda}$. Also, our policies depend on the offered demand λ. An optimal assignment for a given λ may not be the best choice when the actual demand varies. In Section 6, we look at the sensitivity of the throughput to varying demand.

6 A Numerical Example

In this section, we provide in-depth analysis of an example from Section 3.3. Section 6.1 demonstrates how the optimal allocations vary as the demand to the system changes. Section 6.2 investigates the sensitivity of the optimal allocation for a given offered demand to the actual demand. Lastly, Section 6.3 simulates the same example for a given demand level.

6.1 Optimal Server Allocations Under Varying Offered Demand

In this section, we use an example to illustrate the effects on the maximum throughput of increasing the demand λ to the system. We will investigate the system considered earlier in Section 3.3 (see Figure 1 and equation (20)). Instead of looking at a single offered demand $\lambda = 6$, we consider $\lambda \in [0, 20]$ by dividing this range into 500 equal intervals and solving the allocation LP for each value of λ incrementally (i.e., $\lambda = 0.04, 0.08, \ldots, 20$). Note that for this system, we have $\lambda^* \simeq 4.0714$, $\bar{\lambda} = 15$, and $\bar{\mu} = 7.5$. Figure 2(a) gives the optimal assignments to class 1 for each server corresponding to different λ. Figure 2(b) shows d_1^*, d_2^*, and $\mu^*(\lambda)$ as a function of the offered demand λ. Note that optimal allocations for a given λ may not be unique. To avoid fluctuations in the allocations and better see the effects of instability, we consider two specific basic allocations and use them whenever they are feasible and optimal. The first specific basic allocation is obtained by solving the allocation LP given by Andradóttir, Ayhan, and Down [6]. The second specific basic solution is obtained by solving the allocation LP (13) − (17) for $\lambda = \bar{\lambda}$. Then for $\lambda \leq \lambda^*$ and $\lambda \geq \bar{\lambda}$, the optimal allocations are constant and equal to the first and second specific basic solutions, respectively. When $\lambda^* < \lambda < \bar{\lambda}$, neither of the specific basic solutions is optimal, and the allocations obtained from the solution of the allocation LP (13) − (17) are used.



(a) Server Assignments at Class 1

(b) λ vs. d_k

Figure 2: Optimal server assignments at class 1 and corresponding departure rates at each class as a function of λ

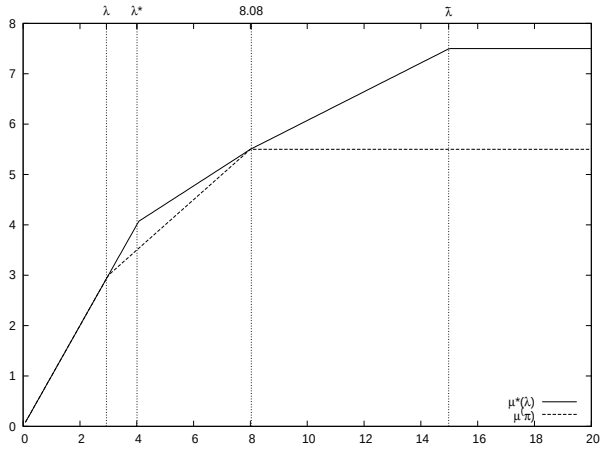
As we can see from Figure 2(a), servers 1 and 2 switch from the second class to the first class as the demand λ increases. Consequently, the servers prefer class 1 as long as there are customers there to process (because a customer leaving class 2 requires more service effort than one leaving class 1). But any excess capacity is devoted to class 2 since it also has

an effect on the throughput. If all the servers work at class 1, then the total processing rate is 15. Hence until $\bar{\lambda} = 15$, some excess capacity is available to allocate to class 2 customers. For $\lambda \leq 8.08$, servers 2 and 3 are able to handle all of the input to class 1, with server 2 helping with class 1, increasingly with λ. After all the efforts of servers 2 and 3 are devoted to class 1 at $\lambda = 8.08$, server 1 starts to help until all of its effort is switched to class 1 as well. Figure 2(b) also shows that as expected by Lemma 3.1, the throughput is a piecewise-linear concave function of the offered demand level. Moreover, we observe that by allowing instability in the queueing network, it is possible for the production output to increase significantly compared to the stable throughput (in this case by a factor of almost two) given sufficient input. However, the optimal departure rates from each class d_1^* and d_2^* display different reactions to the increasing demand λ. They both increase until server 2 starts to spend more time on the first class, so that d_2^* starts to decrease.

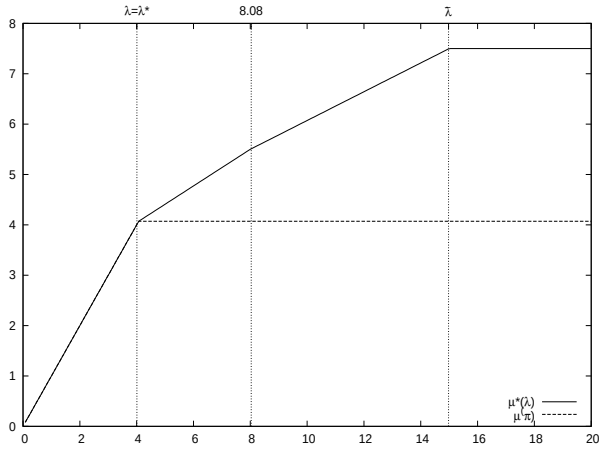
6.2 System Throughput Under Varying Offered Demand

In this section, we look at the performance of the optimal policy developed for one offered demand as a function of the actual demand. For this, we develop a policy based on a fixed λ, and then investigate the system performance when the actual demand λ' is different from λ. Figure 3 depicts the cases where the policy π is designed for $\lambda \in \{3, \lambda^*, 6, 9, 12, \bar{\lambda}\}$, respectively, and provides the optimal throughput $\mu^*(\lambda')$ and actual throughput $\mu_\lambda^\pi(\lambda')$ for different λ'. To obtain $\mu_\lambda^\pi(\lambda')$, we use the optimal fractions obtained for λ in the allocation LP $(13) - (17)$, and solve for d_k^*, for all k, see Section 3.2. The actual throughput of the system differs from the optimal because the policy is designed based on the offered demand λ, and hence the assignments may no longer be optimal for another demand λ'. Note that in Figure 3(a), we have used the allocations obtained as a result of solving the LP $(13) - (17)$ for $\lambda = 3$, and not the ones obtained for the point λ^*. As a result, we observe that the throughput becomes sensitive to the offered demand even for $\lambda \leq \lambda' \leq \lambda^*$. Substituting the allocations obtained at λ^* for $\lambda = 3$, Figure 3(a) would be the same as Figure 3(b).

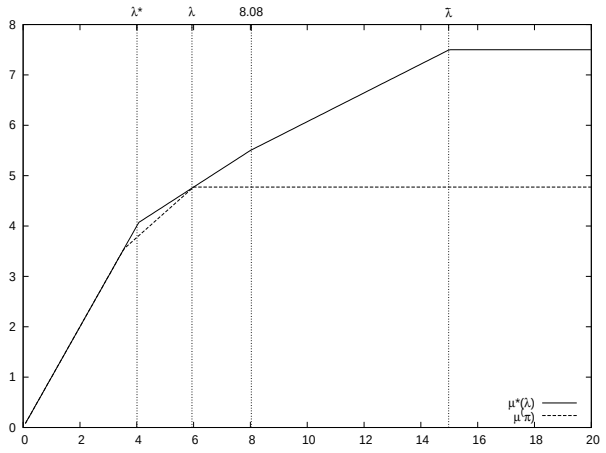
As can be seen in Figure 3, the system performance is sensitive to the actual demand level. Note that λ^* is a critical point in all of the figures. Moreover, we notice that $\mu_\lambda^\pi(\lambda')$ equals $\mu^*(\lambda')$ until some point t_1, then deviates from $\mu^*(\lambda')$, intersecting it only at a second point t_2 (if $\lambda \neq \lambda^*$), and finally becoming constant after the second intersection. For those two points t_1 and t_2, we have $0 \leq t_1 \leq \min\{\lambda, \lambda^*\}$ and $\lambda^* \leq t_2 \leq \bar{\lambda}$. Also, $\mu_\lambda^\pi(\lambda)$ is always equal to $\mu^*(\lambda)$, and in particular $t_1 = \lambda$ when $\lambda \leq \lambda^*$, and $t_2 = \min\{\lambda, \bar{\lambda}\}$ when $\lambda \geq \lambda^*$. We have two special cases, namely when $\lambda = \lambda^*$, where $t_1 = t_2 = \lambda$, and when $\lambda \geq \bar{\lambda}$, where $t_1 = 0$ and $t_2 = \bar{\lambda}$. Also, a comparison of parts (a) and (b) of Figure 3 shows that solving



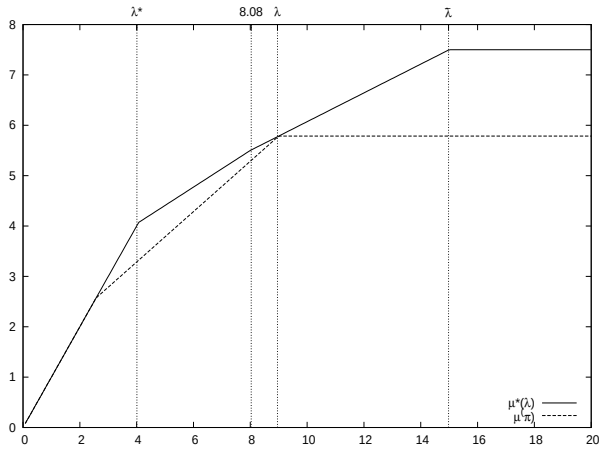
(a) λ=3



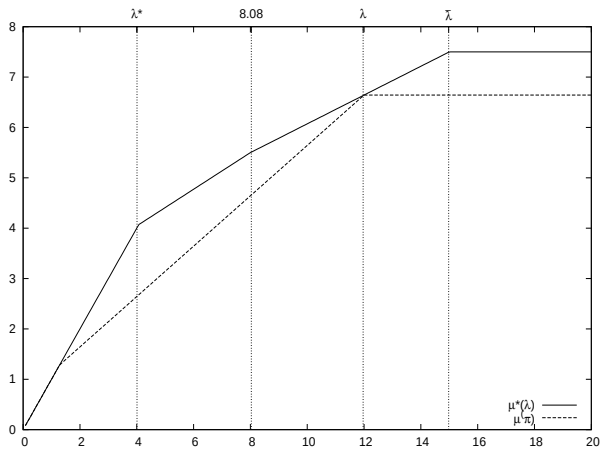
(b) $\lambda = \lambda^*$



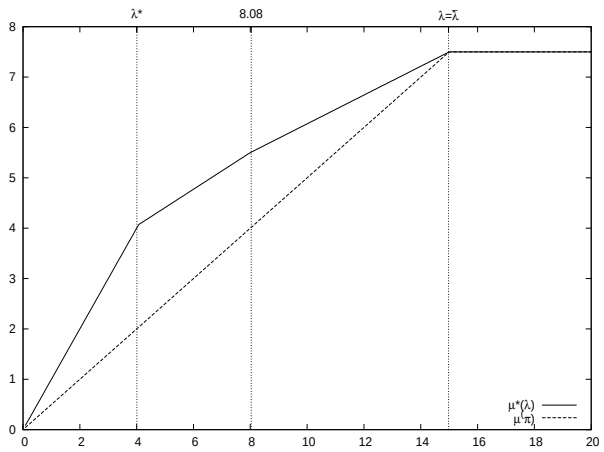
(c) λ=6



(d) λ=9



(e) λ=12



(f) $\lambda = \bar{\lambda}$

Figure 3: Sensitivity analysis when actual offered demand differs from the one designed for.

the allocation LP for $\lambda = 3$, rather than $\lambda = \lambda^*$, achieves higher output for large λ'. This is because the optimal solution for $\lambda = 3$ turns out to be similar to the optimal solution for $\lambda \simeq 8$. Finally, note that the assignments are constant for $\lambda \geq \bar{\lambda}$ (see Figure 2(a)), and hence sensitivity analysis for $\lambda \geq \bar{\lambda}$ will be exactly the same as for $\lambda = \bar{\lambda}$.

If the offered demand to the system is not known beforehand, then there is no single best λ to design for, since solving for λ is not necessarily good for other λ' regardless of whether $\lambda' < \lambda$ or $\lambda' > \lambda$. However, we can still make some generalizations, since system capacity is not lost when $\lambda' < \lambda \leq \lambda^*$ and when $\lambda, \lambda' \geq \bar{\lambda}$. In particular, if the expected offered demand is less than λ^*, then it is best to design for λ^* so that no throughput is lost (see Theorem 1 in [6]). Similarly, if the expected offered demand is greater than $\bar{\lambda}$, then we design for $\bar{\lambda}$ without any loss of throughput. However, we cannot say the same when $\lambda^* < \lambda < \bar{\lambda}$. So, if the expected offered demand is between λ^* and $\bar{\lambda}$, and we design for λ, then the actual throughput cannot exceed $\mu^\pi(\lambda)$. However, we could find a value of λ that minimizes our maximum loss, which in our case corresponds to some $\lambda \in [9, 12]$, where the losses at λ^* and $\bar{\lambda}$ are equal. We could find this point using the Bisection-Extreme Point Search Algorithm (BEPSA), starting with $(\lambda^* + \bar{\lambda})/2$, then moving towards the middle point between the current solution and the extreme point (i.e., λ^* or $\bar{\lambda}$) where the difference is greater. For our case, it turns out that designing a policy for $\lambda = 11$ minimizes our loss at the extreme points. Another approach to deal with the sensitivity of the given policies to the offered demand level is to consider state-dependent policies, as in [20, 29].

6.3 Simulation Results

In this section, we give simulation results for the system analyzed in the previous subsections under Poisson arrivals with rate $\lambda = 6$. We assume the service requirements are exponentially distributed with mean 1 and that servers switch instantaneously, so that no switching times occur. Then, from the allocation LP (13) – (17), we have $\mu^*(6) \simeq 4.7727$ and the optimum assignments are given in (21). Our aim is to observe how our allocation policy with admission and routing control (see Section 4.1) performs in terms of achieving the theoretical throughput value, and also to see if the sets S and U predicted by the allocation LP coincide with the ones actually observed without admission control or controlled routing.

Next we choose $\epsilon = 2/11$ in the server assignment algorithm of Section 4.1, so that $\epsilon' = 0.1$. Then server 1(3) is dedicated to class 2(1), see (21). Moreover, we have that $l_{2,1} = 35$ and $l_{2,2} = 4$, obtained from (23), satisfy step 6 of the assignment algorithm. We simulate this system for one million time units with a warm-up period of length 50,000. We divide the runtime into 40 batches for constructing a 95 percent confidence interval on the throughput.

We expect the throughput to approach $\mu^*(6)(1-\epsilon) \simeq 3.9049$ (see Theorem 4.1) and all nodes to be stable. Figure 4 shows the throughput rate $D^\pi(t)/t$ as a function of time. We observe that the throughput approaches its limiting value from above. The resulting 95 percent confidence interval for the throughput is (3.9007, 3.9101) with an average of 3.9054. We have also prepared plots of the queue lengths over time at classes 1 and 2. These plots are omitted here to conserve space, but can be found in Tekin [41]. As expected given the results of Section 4.1, the queue length at both classes displays stable behavior.

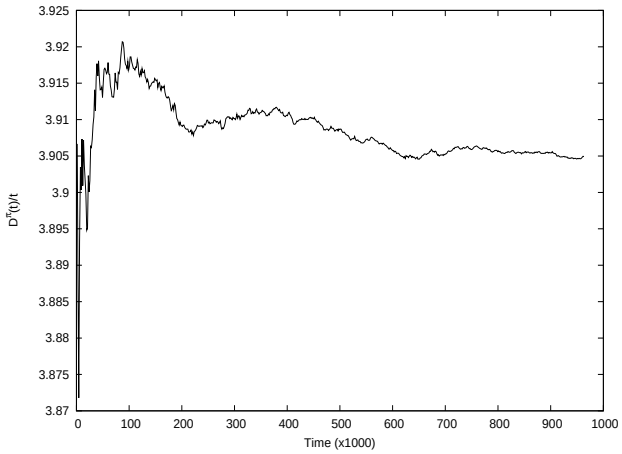


Figure 4: Average throughput with admission and routing control.

Finally, we observe the system under the policy of Section 4.1 without admission and routing controls. For this, we follow the same steps as in Section 4.1, but omit steps 2 and 3 and choose $\epsilon' = 0.1$ in step 6. Then we expect the throughput to be no smaller than $\mu^*(6)(1-\epsilon') \simeq 4.2954$. As before, we have $l_{2,1} = 35$ and $l_{2,2} = 4$, obtained from (23), satisfy step 6 of the assignment algorithm. We simulate this system for eight million time units with a warm-up period of length 300,000, and divide the run time into 40 batches. A longer run length is chosen for this version of the system to observe the queue length process of class 1 (which is expected to be stable, see Section 3.3) for a longer period of time. Figure 5 shows the throughput as a function of time. The resulting 95 percent confidence interval for the throughput is (4.7708, 4.7736) with an average of 4.7722. We have also plotted the queue lengths over time at classes 1 and 2 (see Tekin [41]). In accordance with the results of Section 3.3, the queue length at class 1 displays stable behavior, whereas the queue length at class 2 increases over time. Thus the stable and unstable sets in the original queueing system operating under this policy appear to coincide with the stable and unstable sets S and U defined in (18) and (19) for the allocation LP (13) – (17). As we observe, dropping steps 2 and 3 of the policy of Section 4.1 results in significantly increased throughput at a cost of having an unstable system. This is the case because we do not reject any incoming

demand (i.e., no admission control) and keep the second class busy at all times (i.e., no routing control).

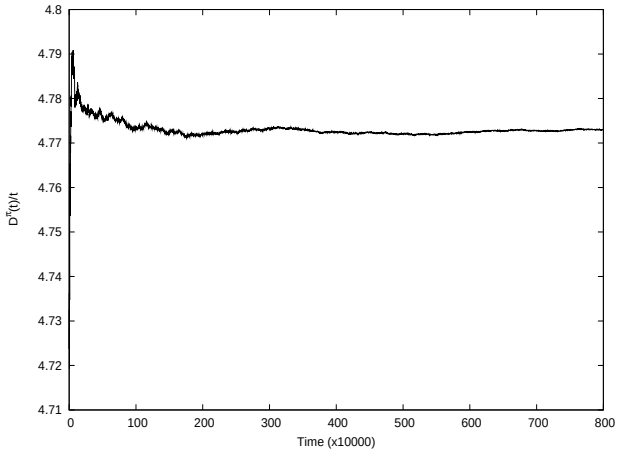


Figure 5: Average throughput without admission or routing control.

7 Conclusions

We have developed generalized round robin server assignment policies for a possibly unstable queueing network with flexible servers, i.i.d. interarrival, service, and switching times, and probabilistic routing. These policies are shown to achieve any throughput less than the maximum value computed using a simple LP. In fact, allowing instability can increase the production throughput significantly given sufficient demand, resulting in higher revenues. In another paper [42], we demonstrate that this is indeed the case for a serial manufacturing process with inspection and repair stations. We have also shown how to determine the saturation input and the corresponding maximum output, and provided means to check the feasibility of a desired output given the available offered demand.

One drawback for a fixed server assignment policy is the sensitivity of the throughput to fluctuations in the offered demand. We have shown that this sensitivity is eliminated and our policies are robust when the system is stable or the demand is above the saturation level. We have also discussed how to choose what demand level a policy should be designed for to minimize the maximum loss in the presence of demand uncertainty. In actual production systems, demand often changes over time. In that case, we can simply modify our policies by letting the server allocations adjust with time according to the forecasted demand.

Another performance measure of interest is the total number of items processed during each server visit to a given class (i.e., the lot sizes). In general low switching rates are effective

with respect to throughput, but they can result in the production of large lots, which in turn implies longer lead times and higher inventories. Hence, in future work it would be interesting to design policies that simultaneously consider throughput and lot sizes.

Acknowledgments

This research was supported by the National Science Foundation under Grant CMMI-0856600. In addition, the research of the third author was supported by the Natural Sciences and Engineering Research Council of Canada. The authors thank three anonymous referees and associate editor for their careful reviews and valuable comments.

A Proofs of Theorems 3.1, 4.1, and 4.2

In this section we give formal proofs to Theorems 3.1, 4.1, and 4.2. We start by constructing the underlying fluid model for the original queueing network described in Section 2, and then describe a Markov process model for the same queueing network. Using these results we prove that the algorithms provided in Sections 4.1 and 4.2 can be used to obtain throughput that is arbitrarily close to the maximum output $\mu^*(\lambda)$ given the available demand λ. More specifically, we start with part (b) of Theorem 3.1. Next, we prove Theorem 4.1, then part (a) of Theorem 3.1 follows. Finally we show that the algorithm provided in Section 4.2 also achieves the target throughput, as stated in Theorem 4.2.

The fluid models involve smoothing out discrete processes, using the SLLN. We now develop a fluid model for the original queueing network described in Section 2 under a server assignment policy π. Let $q = \sum_{k=1}^{K} Q_k(0)$. Suppose that the function $(\bar{Q}_k(\cdot), \bar{T}_{j,k}(\cdot), \forall j, k)$ is a limit point of $(Q_k(qt)/q, T_{j,k}(qt)/q, \forall j, k)$ when $q \to \infty$. Then $(\bar{Q}_k(\cdot), \bar{T}_{j,k}(\cdot) : k = 1, \ldots, K)$ is a fluid limit of the system. Each component of a fluid limit is absolutely continuous (and thus differentiable almost everywhere in $[0, \infty)$ with respect to the Lebesgue measure, see Dai [17], page 20). If we require the derivative of a quantity, we will assume it is taken at a time point t such that the derivative exists (such a point is known as a regular point); we will not require derivatives involving class k at a moment when $\bar{Q}_k(t)$ hits zero for any k. For each class k, let $\bar{A}_k(t) = \lim_{q\to\infty} A_k(qt)/q$ and $\bar{D}_k(t) = \lim_{q\to\infty} D_k(qt)/q$ (almost surely uniformly on compact intervals) be the fluid limits for the arrival and departure processes $A_k(t)$ and $D_k(t)$, respectively. Then the deterministic analogs $\bar{A}$, $\bar{D}$, and $\bar{Q}$ of the queueing

network processes A, D, and Q satisfy the following equations (see Theorem 4.1 of Dai [15]):

$$\bar{A}_k(t) = \lambda p_{0,k} t + \sum_{i=1}^{K} \sum_{j=1}^{M} p_{i,k} \mu_{j,i} \bar{T}_{j,i}(t), \ k = 1, \ldots, K; \tag{34}$$

$$\bar{D}_k(t) = \sum_{j=1}^{M} \mu_{j,k} \bar{T}_{j,k}(t), \ k = 1, \ldots, K; \tag{35}$$

$$\bar{Q}_k(t) = \bar{Q}_k(0) + \lambda p_{0,k} t + \sum_{i=1}^{K} \sum_{j=1}^{M} p_{i,k} \mu_{j,i} \bar{T}_{j,i}(t) - \sum_{j=1}^{M} \mu_{j,k} \bar{T}_{j,k}(t), \ k = 1, \ldots, K. \tag{36}$$

Equations $(34) - (36)$ are obtained from $(8) - (10)$ by replacing $S_{j,k}(t)$, $E_k(t)$, and $\Phi_{i,k}(n)$ by their asymptotic means. The dependence of $(34) - (36)$ on π is determined through the functions $\{\bar{T}_{j,k}(t)\}$ and Theorem 4.1 of [15] applies because there are only a finite number of servers, each working on at most one customer (see the remark on page 58 of [15]).

The next step is to define a Markov process $X = \{X(t), t > 0\}$ which describes the dynamics of the queueing network described in Section 2 with K classes and M servers operating under a generalized round robin server assignment policy π, where each server j cycles among all the classes k on its list V_j^π, serving a maximum of $l_{j,k}^\pi$ customers at class k before moving to the next class. Let $U(t)$ and $V_{j,k}(t)$, $j = 1, \ldots, M$, $k = 1, \ldots, K$, be the residual interarrival and service times defined in Section 2 and $W_j(t)$ be the residual switching time at time t for server j. Also, let $L_j(t)$ be the location of server j at time t (set to the destination class if the server is switching at time t and to the current class when the server idles), $I_j(t)$ be the status of server j (0 if the server is idle or switching, 1 if busy), and $N_j(t)$ be the number of customers finished by server j at the current location (reset to zero each time server j idles or makes a switch). Note that since we have non-preemptive service, the residual service time can be only at the current location $L_j(t)$ at time t, so let $V_j(t)$ be the residual service time for server j. The piecewise-continuous variables $\{U(t), V_j(t), W_j(t)\}$ are taken to be right continuous. Then the process $X(t)$ defined by

$$X(t) = (U(t), V_j(t), W_j(t), Q_k(t), L_j(t), I_j(t), N_j(t); j = 1, \ldots, M, k = 1, \ldots, K)$$

can be shown to have the strong Markov property as in Section 4 of Davis [19], with elements

$$x \in \mathbb{R}_+ \times \mathbb{R}_+^M \times \mathbb{R}_+^M \times \mathbb{Z}_+^K \times \{1, \ldots, K\}^M \times \{0, 1\}^M \times \{0, 1, \ldots, \max l_{j,k} - 1\}^M.$$

Next, we need to make minor modifications for the allocation policy described in Section 4.1 as it results in a slightly modified network. A similar Markov process exists for the modified network under admission control and controlled routing as for the original network,

with the only difference that the dimension of the state is increased by the additional class and server, so that the Markov process evolves on

$$x \in \mathbb{R}_+ \times \mathbb{R}_+^{M+1} \times \mathbb{R}_+^M \times \mathbb{Z}_+^{K+1} \times \{1, \ldots, K\}^M \times \{0,1\}^M \times \{0, 1, \ldots, \max_{j,k} l_{j,k} - 1\}^M.$$

Note that $V_{M+1}(t)$ is the only information needed for the $(M+1)^{th}$ server, because it is dedicated to class $K+1$. In the case of the allocation policy of Section 4.2, the residual service time $V_j(t)$ for each server j needs to be replaced by the residual service time $V_k(t)$ at each class k. However, since we will not be proving the stability of the queueing network under the policy of Section 4.2, we do not describe the resulting Markov process in detail.

Proof of Theorem 3.1(b). To prove the theorem we proceed by contradiction. Assume that there exists a policy π and a subset A of the sample space Ω with $P(A) > 0$, such that

$$\limsup_{t\to\infty} \frac{D^\pi(t,\omega)}{t} > \mu^*(\lambda), \ \forall \omega \in A, \tag{37}$$

where $D^\pi(t,\omega)$ is the total number of departures from the system under the policy π in $(0,t]$ for the sample path ω. By the i.i.d. assumption on the primitive processes, there exists a set A' with $P(A') = P(A)$ such that for all $\omega \in A'$, and any $\epsilon, \epsilon_1 > 0$, there exists $T_1(\omega)$ and $N(\omega)$ such that for all $t \geq T_1(\omega), n \geq N(\omega)$, $i = 0, \ldots, K$, $k = 1, \ldots, K$, and $j = 1, \ldots, M$,

$$\left| \frac{E_k(t,\omega)}{t} - \lambda p_{0,k} \right| \leq \epsilon_1, \ \left| \frac{\Phi_{k,i}(n,\omega)}{n} - p_{k,i} \right| \leq \epsilon_1, \ \left| \frac{S_{j,k}(t,\omega)}{t} - \mu_{j,k} \right| \leq \epsilon.$$

Next we obtain bounds on the cumulative processes, starting with the departure processes. We have $D_k(t) = \sum_{j=1}^M S_{j,k}(T_{j,k}(t)), \ k = 1, \ldots, K$. On the sample path $\omega \in A$, some servers may spend a finite amount of time at given classes, resulting in two cases:

- For pairs j,k such that $\lim_{t\to\infty} T_{j,k}(t,\omega) < \infty$, we have $S_{j,k}(T_{j,k}(t),\omega)/t \to 0$, since $S_{j,k}(t,\omega) < \infty$, for all t, by assumption (5).

- For pairs j,k such that $\lim_{t\to\infty} T_{j,k}(t,\omega) = \infty$, we can find $T_2(\omega)$ such that for all $t \geq T_2(\omega)$, we have $T_{j,k}(t,\omega) \geq T_1(\omega)$ implying

$$\left| \frac{S_{j,k}(T_{j,k}(t,\omega))}{T_{j,k}(t,\omega)} - \mu_{j,k} \right| \leq \epsilon.$$

Let $M_k(\omega) = \{j : \mu_{j,k} > 0 \text{ and } \lim_{t\to\infty} T_{j,k}(t,\omega) = \infty\}$. We have

$$\frac{D_k(t,\omega)}{t} = \sum_{j \in M_k(\omega)} \frac{S_{j,k}(T_{j,k}(t,\omega))}{T_{j,k}(t,\omega)} \times \frac{T_{j,k}(t,\omega)}{t} + \sum_{j \notin M_k(\omega)} \frac{S_{j,k}(T_{j,k}(t,\omega))}{t}, \ k = 1, \ldots, K.$$

Let $\delta_{j,k}(t,\omega) = T_{j,k}(t,\omega)/t$. For any $\epsilon_2 > 0$, there exists $T_3(\omega)$ such that for all $t \geq T_3(\omega)$, we have $\sum_{j \notin M_k(\omega)} S_{j,k}(T_{j,k}(t,\omega))/t \leq \epsilon_2$, for $k = 1, \ldots, K$. Then for $t \geq \max\{T_2(\omega), T_3(\omega)\}$,

$$\frac{D_k(t,\omega)}{t} \leq \sum_{j \in M_k(\omega)} (\mu_{j,k} + \epsilon)\delta_{j,k}(t,\omega) + \epsilon_2,\ k = 1, \ldots, K.$$

Let $\epsilon_3 = \epsilon M + \epsilon_2$, which implies that $\epsilon_3 \geq \max_k\{\epsilon \sum_{j \in M_k(\omega)} \delta_{j,k}(t,\omega) + \epsilon_2\}$, and thus for $t \geq \max\{T_2(\omega), T_3(\omega)\}$, we obtain

$$\frac{D_k(t,\omega)}{t} - \epsilon_3 \leq \sum_{j \in M_k(\omega)} \mu_{j,k}\delta_{j,k}(t,\omega),\ k = 1, \ldots, K. \tag{38}$$

Next, we bound the arrival process to each class. Let $\mathcal{K} = \{1, \ldots, K\}$ denote the set of all classes in the network, and define

$$\mathcal{K} \setminus \bar{\mathcal{K}}(\omega) = \{k : \lim_{t\to\infty} T_{j,k}(t,\omega) < \infty, \forall j \text{ with } \mu_{j,k} > 0\} = \{k : M_k(\omega) = \emptyset\}.$$

Note that all servers capable of working at the classes in $\mathcal{K} \setminus \bar{\mathcal{K}}(\omega)$ spend only a finite amount of time at those classes, so that the number of departures is bounded. We have

$$A_k(t,\omega) = E_k(t,\omega) + \sum_{i=1}^{K} \Phi_{i,k}(D_i(t,\omega)),\ k = 1, \ldots, K.$$

For $i \in \bar{\mathcal{K}}(\omega)$, $\lim_{t\to\infty} D_i(t,\omega) = \infty$, and hence there exists $T_4(\omega)$ such that for all $t \geq T_4(\omega)$, we have $D_i(t,\omega) > N(\omega)$, implying

$$\left| \frac{\Phi_{i,k}(D_i(t,\omega))}{D_i(t,\omega)} - p_{i,k} \right| \leq \epsilon_1,\ k = 0, 1, \ldots, K.$$

For $i \in \mathcal{K} \setminus \bar{\mathcal{K}}(\omega)$, $\lim_{t\to\infty} D_i(t,\omega) < \infty$, and $\lim_{t\to\infty} \Phi_{i,k}(D_i(t,\omega))/t = 0$. Hence, for any $\epsilon_4 > 0$, there exists $T_5(\omega)$ such that for all $t \geq T_5(\omega)$, we have

$$\sum_{i \in \mathcal{K} \setminus \bar{\mathcal{K}}(\omega)} \frac{\Phi_{i,k}(D_i(t,\omega))}{t} \leq \epsilon_4,\ k = 0, 1, \ldots, K.$$

Then, for the arrival process, we have for $t \geq \max\{T_1(\omega), T_4(\omega), T_5(\omega)\}$,

$$\frac{A_k(t,\omega)}{t} \leq \lambda p_{0,k} + \epsilon_1 + \sum_{i \in \bar{\mathcal{K}}(\omega)} (p_{i,k} + \epsilon_1)\frac{D_i(t,\omega)}{t} + \epsilon_4,\ k = 1, \ldots, K.$$

Substituting in (38), we get, for $t \geq \max\{T_1(\omega), T_2(\omega), T_3(\omega), T_4(\omega), T_5(\omega)\}$,

$$\frac{A_k(t,\omega)}{t} \leq \lambda p_{0,k} + \sum_{i \in \bar{\mathcal{K}}(\omega)} p_{i,k}\frac{D_i(t,\omega)}{t} + \epsilon_1 \sum_{i \in \bar{\mathcal{K}}(\omega)} \sum_{j \in M_i(\omega)} \delta_{j,i}(t,\omega)\mu_{j,i} + \epsilon_1 \sum_{i \in \bar{\mathcal{K}}(\omega)} \epsilon_3 + \epsilon_4 + \epsilon_1,\ k = 1, \ldots, K.$$

We also have $D_k(t,\omega) \leq A_k(t,\omega)+Q_k(0)$ for all $t \geq 0$. Let $\epsilon_5 = \epsilon_1 KM\mu + K\epsilon_1\epsilon_3 + \epsilon_4 + 2\epsilon_1$, where $\mu = \max\{\mu_{j,i}, j = 1,\ldots,M, i = 1,\ldots,K\}$, so that

$$\epsilon_5 \geq \max_k \Big\{\epsilon_1 \sum_{i\in\bar{\mathcal{K}}(\omega)} \sum_{j\in M_i(\omega)} \delta_{j,i}(t,\omega)\mu_{j,i} + \epsilon_1 \sum_{i\in\bar{\mathcal{K}}(\omega)} \epsilon_3 + \epsilon_4 + 2\epsilon_1\Big\}.$$

Then for $t \geq \max\{T_1(\omega), T_2(\omega), T_3(\omega), T_4(\omega), T_5(\omega), Q_k(0)/\epsilon_1\}$ we have

$$\frac{D_k(t,\omega)}{t} - \epsilon_5 \leq \lambda p_{0,k} + \sum_{i\in\bar{\mathcal{K}}(\omega)} p_{i,k}\frac{D_i(t,\omega)}{t},\ k = 1,\ldots,K. \tag{39}$$

Finally, we bound the departure process from the system, $D(t,\omega) = \sum_{i=1}^K \Phi_{i,0}(D_i(t,\omega))$. For $t \geq \max\{T_4(\omega), T_5(\omega)\}$, we have

$$\frac{D(t,\omega)}{t} \leq \sum_{i\in\bar{\mathcal{K}}(\omega)} (p_{i,0}+\epsilon_1) \times \frac{D_i(t,\omega)}{t} + \epsilon_4.$$

Let $\epsilon_6 = \epsilon_1 K(\epsilon_3 + M\mu) + \epsilon_4$, so that (38) implies that $\epsilon_6 \geq \epsilon_1 \sum_{i\in\bar{\mathcal{K}}(\omega)} D_i(t,\omega)/t + \epsilon_4$. Then we get for $t \geq \max\{T_4(\omega), T_5(\omega)\}$

$$\frac{D(t,\omega)}{t} - \epsilon_6 \leq \sum_{i\in\bar{\mathcal{K}}(\omega)} p_{i,0}\frac{D_i(t,\omega)}{t}. \tag{40}$$

By assumption, under policy π, the departure process satisfies (37). Let $\limsup_{t\to\infty} D^\pi(t,\omega)/t = l > \mu^*(\lambda)$. Then for any $\epsilon_7 > 0$, $D^\pi(t,\omega)/t \geq l - \epsilon_7$ infinitely often. Then we can choose a time $t_0 \geq \max\{T_1(\omega), T_2(\omega), T_3(\omega), T_4(\omega), T_5(\omega), Q_k(0)/\epsilon_1\}$ with an ϵ_7 small enough so that $D^\pi(t_0,\omega)/t_0 > \mu^*(\lambda)$ and also the bounds in (38), (39) and (40) are satisfied at t_0. Rewriting (38) – (40) for the cumulative processes at time t_0, we get

$$\frac{D_k(t_0,\omega)}{t_0} - \epsilon_3 \leq \sum_{j\in M_k(\omega)} \mu_{j,k}\delta_{j,k}(t_0,\omega),\ k = 1,\ldots,K, \tag{41}$$

$$\frac{D_k(t_0,\omega)}{t_0} - \epsilon_5 \leq \lambda p_{0,k} + \sum_{i\in\bar{\mathcal{K}}(\omega)} p_{i,k}\frac{D_i(t_0,\omega)}{t_0},\ k = 1,\ldots,K, \tag{42}$$

$$\frac{D(t_0,\omega)}{t_0} - \epsilon_6 \leq \sum_{i\in\bar{\mathcal{K}}(\omega)} p_{i,0}\frac{D_i(t_0,\omega)}{t_0}. \tag{43}$$

Next, our aim is to show that given the above bounds on the cumulative processes, there exists a solution to the LP (13) – (17) with an objective value greater than $\mu^*(\lambda)$, which will yield the desired contradiction. To see this, define

$$d_k = \begin{cases} 0 & \text{if } k \in K \setminus \bar{\mathcal{K}}(\omega), \\ \frac{D_k(t_0,\omega)}{t_0} & \text{if } k \in \bar{\mathcal{K}}(\omega); \end{cases}$$

$$\delta_{j,k} = \begin{cases} 0 & \text{if } j \notin M_k(\omega), \\ \delta_{j,k}(t_0, \omega) & \text{if } j \in M_k(\omega). \end{cases}$$

Substituting these in (41), (42) and (43) and noting that the bounds in (41) – (43) hold for arbitrarily small ϵ_3, ϵ_5, and ϵ_6, respectively, we get after a little manipulation

$$\sum_{i=1}^{K} p_{i,0} d_i > \mu^*(\lambda), \tag{44}$$

$$d_k \leq \sum_{j=1}^{M} \mu_{j,k} \delta_{j,k},\ k = 1, \ldots, K, \tag{45}$$

$$d_k \leq \lambda p_{0,k} + \sum_{i=1}^{K} p_{i,k} d_i,\ k = 1, \ldots, K. \tag{46}$$

By definition, we have $d_k \geq 0$ and $\delta_{j,k} \geq 0$. Moreover, $\sum_{k=1}^{K} T_{j,k}(t_0, \omega) \leq t_0$ implies that $\sum_{k=1}^{K} \delta_{j,k} \leq 1$ for $j = 1, \ldots, M$. Then we see that along this sample path ω under the policy π, we can construct a solution to the LP (13) – (17) with an objective value greater than $\mu^*(\lambda)$, a contradiction. □

Remark A.1. *The proof of Theorem 3.1(b) is complicated by the fact that the instantaneous output rate from the system is not bounded above by $\mu^*(\lambda)$, since we are not placing any restrictions on the stability of the system. The size of the fluid queues must be taken into account, so we cannot discount the situation where the instantaneous throughput may rise above $\mu^*(\lambda)$ infinitely often, due to positive queue lengths (in the fluid limit) at appropriate queues. Thus traditional techniques can be used to show that $\liminf_{t\to\infty} D(t)/t \leq \mu^*(\lambda)$; the challange is to show that $\limsup_{t\to\infty} D(t)/t \leq \mu^*(\lambda)$.*

Proof of Theorem 4.1 and Theorem 3.1(a). We will refer to the network obtained as a result of the controlled routing in step 3 of the policy described in Section 4.1 as the "modified" queueing network. Hence step 3 of this policy results in a modified network under admission control. Let $\bar{P}$ be the routing matrix for the modified network (so that $\bar{P}$ has (i, k) entry $\bar{p}_{i,k}$ for $i, k = 1, \ldots, K+1$). Then we have that $(I - \bar{P})$ is invertible and $\bar{P}^n \to 0$ since the modified network is open (see, e.g., Lawler [32], page 27).

To prove Theorem 4.1, we need to develop the queueing network equations and the corresponding fluid model for the modified network. We can obtain these in the same way

as we did in Section 2 and above for the original network. So we have

$$\begin{aligned} A_k(t) &= E_k(t) + \sum_{i=1}^{K+1} \bar{\Phi}_{i,k}(D_i(t)),\ k = 1, \ldots, K+1; \\ D_k(t) &= \sum_{j=1}^{M+1} S_{j,k}(T_{j,k}(t)),\ k = 1, \ldots, K+1; \\ Q_k(t) &= Q_k(0) + A_k(t) - D_k(t),\ k = 1, \ldots, K+1; \end{aligned}$$

and $0 \leq \sum_{k=1}^{K+1} T_{j,k}(t) \leq t,\ j = 1, \ldots, M+1$, where $\bar{\Phi}_{i,k}(n) = \sum_{l=1}^{n} \bar{\phi}_{i,k}(l)$ and the random variables $\bar{\phi}_{i,k}(l)$ are independent and multi-Bernoulli, so that for each i, l, exactly one $\bar{\phi}_{i,k}(l)$ is equal to 1 with probability $\bar{p}_{i,k}$, for $k = 0, \ldots, K+1$, and the remainder are zero (meaning that the l^{th} customer from class i is routed to class k). Similarly, fluid limits $\bar{A}_k(t)$, $\bar{D}_k(t)$, and $\bar{Q}_k(t)$ for the modified network under admission control are defined in the same manner as for the original network, for $k = 1, \ldots, K+1$, and satisfy the equations

$$\begin{aligned} \bar{A}_k(t) &= \lambda' \bar{p}_{0,k} t + \sum_{i=1}^{K+1} \sum_{j=1}^{M+1} \bar{p}_{i,k} \mu_{j,i} \bar{T}_{j,i}(t),\ k = 1, \ldots, K+1; \\ \bar{D}_k(t) &= \sum_{j=1}^{M+1} \mu_{j,k} \bar{T}_{j,k}(t),\ k = 1, \ldots, K+1; \\ \bar{Q}_k(t) &= \bar{Q}_k(0) + \lambda' \bar{p}_{0,k} t + \sum_{i=1}^{K+1} \sum_{j=1}^{M+1} \bar{p}_{i,k} \mu_{j,i} \bar{T}_{j,i}(t) - \sum_{j=1}^{M+1} \mu_{j,k} \bar{T}_{j,k}(t), \\ &\qquad\qquad k = 1, \ldots, K+1; \end{aligned} \tag{47}$$

subject to the conditions

$$0 \leq \sum_{k=1}^{K+1} \bar{T}_{j,k}(t) \leq t,\ j = 1, \ldots, M+1;$$

$$\bar{T}_{j,k}(0) = 0 \text{ and } \bar{T}_{j,k}(\cdot) \text{ is non-decreasing for } j = 1, \ldots, M+1, k = 1, \ldots, K+1;$$

$$\bar{Q}_k(t) \geq 0, \qquad k = 1, \ldots, K+1;$$

$$\begin{aligned} \frac{l^{\pi}_{j,k} m_{j,k}}{s^{\pi}_j + \sum_{i \in V^{\pi}_j} l^{\pi}_{j,i} m_{j,i}} \leq \frac{d\bar{T}_{j,k}(t)}{dt} \leq 1, &\qquad j = 1, \ldots, M+1, \\ &\qquad k = 1, \ldots, K+1, \text{ whenever } \bar{Q}_k(t) > 0. \end{aligned} \tag{48}$$

The lower bound in (48) can be derived as in Andradóttir, Ayhan, and Down [6].

Let $r_k = d^*_k$, for $k = 1, \ldots, K$, and $r_{K+1} = \lambda \bar{p}_{0,K+1} + \sum_{i=1}^{K} d^*_i \bar{p}_{i,K+1}$. Then $r_1, \ldots, r_{K+1}$ satisfy the traffic equations for the modified queueing network under the demand λ, so that

$$r_k = \lambda \bar{p}_{0,k} + \sum_{i=1}^{K+1} r_i \bar{p}_{i,k},\ k = 1, \ldots, K+1. \tag{49}$$

To see this, recall that $a_k^* = \lambda p_{0,k} + \sum_{i=1}^K d_i^* p_{i,k}$, for all k. First consider $k \in U$. Then,

$$\begin{aligned}\lambda \bar{p}_{0,k} + \sum_{i=1}^{K+1} r_i \bar{p}_{i,k} &= \lambda p_{0,k}\epsilon_k + \sum_{i=1}^{K} d_i^* p_{i,k}\epsilon_k \\ &= \epsilon_k(\lambda p_{0,k} + \sum_{i=1}^{K} d_i^* p_{i,k}) = \frac{d_k^*}{a_k^*} a_k^* = d_k^* = r_k,\ k \in U,\end{aligned}$$

as required. Next consider the classes $k \in S$. Then $d_k^* = a_k^*$, and we get

$$\lambda \bar{p}_{0,k} + \sum_{i=1}^{K+1} r_i \bar{p}_{i,k} = \lambda p_{0,k} + \sum_{i=1}^{K} d_i^* p_{i,k} = a_k^* = d_k^* = r_k,\ k \in S.$$

Finally, $r_{K+1} = \lambda \bar{p}_{0,K+1} + \sum_{i=1}^{K+1} r_i \bar{p}_{i,K+1}$ follows from the definition of r_{K+1} and the fact that $\bar{p}_{K+1,K+1} = 0$. We have shown that $r_1, \ldots, r_{K+1}$ satisfy (49), and since $I - \bar{P}$ is invertible, the solution is unique.

Let α_k, $k = 1, \ldots, K+1$, be the unique solution of the system of equations (49) when $\lambda = 1$. Then we have $\alpha_k = d_k^*/\lambda$, $k = 1, \ldots, K$, and $\alpha_{K+1} = \bar{p}_{0,K+1} + (\sum_{i=1}^K d_i^* \bar{p}_{i,K+1})/\lambda$. Moreover, by constraint (14) in the allocation LP, we have

$$r_k = d_k^* \ \leq \ \sum_{j=1}^{M} \mu_{j,k} \delta_{j,k}^* = \sum_{j=1}^{M+1} \mu_{j,k} \delta_{j,k}^*,\ k = 1, \ldots, K; \tag{50}$$

and

$$r_{K+1} \ \leq \ \lambda = \sum_{j=1}^{M+1} \mu_{j,K+1} \delta_{j,K+1}^* \tag{51}$$

follows from the facts that $p_{K+1,0} = 1$, r_{K+1} is the flow through node $K+1$ in a stable queueing network with offered demand λ and routing matrix $\bar{P}$, and if $r_{K+1} > \lambda$, the system would have more output than input. Then, by the server allocation policy π, and (50) – (51), we have, for all $k = 1, \ldots, K+1$,

$$\sum_{j=1}^{M+1} \frac{l_{j,k}^\pi}{s_j^\pi + \sum_{i \in V_j^\pi} l_{j,i}^\pi m_{j,i}} \geq \sum_{j=1}^{M+1} \mu_{j,k} \delta_{j,k}^* (1 - \epsilon') \geq r_k (1 - \epsilon'). \tag{52}$$

Let $\lambda' = \lambda(1 - \epsilon)$ be the thinned offered demand, and $r_k' = \lambda' \alpha_k$, $k = 1, \ldots, K+1$, be the solution to the traffic equations for the modified network corresponding to the offered demand λ'. Since $(1 - \epsilon')/(1 - \epsilon) = 1 + \epsilon'$, we have

$$r_k(1 - \epsilon') = \lambda \alpha_k (1 - \epsilon') = \lambda' \alpha_k \frac{1 - \epsilon'}{1 - \epsilon} = r_k'(1 + \epsilon'),\ k = 1, \ldots, K+1. \tag{53}$$

Substituting (53) in (52), we get

$$\sum_{j=1}^{M+1} \frac{l_{j,k}^{\pi}}{s_j^{\pi} + \sum_{i \in V_j^{\pi}} l_{j,i}^{\pi} m_{j,i}} \geq r_k'(1+\epsilon'), \ k = 1, \ldots, K+1. \tag{54}$$

Equations (48) and (54) for the modified network imply that when $\bar{Q}_k(t) > 0$, $\sum_{j=1}^{M+1} \frac{d\bar{T}_{j,k}(t)}{dt} \geq r_k'(1+\epsilon')$. By Theorem 2.4.9 of Dai [17], this means that there is a finite time t_0 such that the system is empty and the fluid model for the modified network is stable under the offered demand λ'. Then by Theorem 4.2 of Dai [15], the Markov chain describing the dynamics of the modified network is positive Harris recurrent. Hence, the modified queueing network is stable for the offered demand λ', and the distribution of the queue length process $\{Q_k(t)\}$, $k = 1, \ldots, K+1$, converges to a steady state limit as $t \to \infty$.

Finally, it remains to find the throughput μ^{π} for the modified network with offered demand λ' under the server assignment policy π with admission control and controlled routing, i.e., without the customers serviced at class $K+1$. For this, consider the fluid scale queue length differential equation obtained from (47) for the modified network under admission control. Given the queueing network is stable, there exists some time t_0 such that $\sum_{k=1}^{K+1} \bar{Q}_k(t) = 0$ for $t \geq t_0$. Then, for any $t > t_0$, we have

$$0 = \lambda' \bar{p}_{0,k} + \sum_{i=1}^{K+1} \bar{p}_{i,k} \sum_{j=1}^{M+1} \mu_{j,i} \frac{d\bar{T}_{j,i}(t)}{dt} - \sum_{j=1}^{M+1} \mu_{j,k} \frac{d\bar{T}_{j,k}(t)}{dt}, \ k = 1, \ldots, K+1. \tag{55}$$

Let $\bar{d}_k(t) = d\bar{D}_k(t)/dt = \sum_{j=1}^{M+1} \mu_{j,k} d\bar{T}_{j,k}(t)/dt$ be the fluid level departure rate from class k, for $k = 1, \ldots, K+1$, in the above equation (55). Then we see that solving the set of equations (55) for $\bar{d}_k(t)$, $k = 1, \ldots, K+1$, gives the same solution as for the traffic equations in (49) when the offered demand is λ'. Hence, $\bar{d}_k(t)$, $k = 1, \ldots, K+1$, are uniquely given by $\bar{d}_k(t) = r_k'$, $k = 1, \ldots, K+1$, and the fluid level total throughput rate $\bar{d}(t) = d\bar{D}(t)/dt$ from classes $k = 1, \ldots, K$ is

$$\bar{d}(t) \ = \ \sum_{k=1}^{K} \bar{p}_{k,0} \sum_{j=1}^{M+1} \mu_{j,k} \frac{d\bar{T}_{j,k}(t)}{dt} = \sum_{k=1}^{K} r_k' \bar{p}_{k,0} = \sum_{k=1}^{K} (1-\epsilon) d_k^* p_{k,0} = \mu^*(\lambda)(1-\epsilon),$$

and hence

$$\bar{D}(t) - \bar{D}(t_0) = \mu^*(\lambda)(1-\epsilon)(t - t_0). \tag{56}$$

Connecting back to the queueing network, recall that $\bar{D}(t)$ is a limit point of $D^{\pi}(qt)/q$ as $q \to \infty$, where $D^{\pi}(t) = \sum_{k=1}^{K} \bar{\Phi}_{k,0}(D_k(t))$ is the total number of departures from the modified network until time t from classes $k = 1, \ldots, K$ with offered demand λ' under the

policy π. Assume $l = \limsup_{t\to\infty} D^\pi(t)/t \neq \mu^*(\lambda)(1-\epsilon)$. Then, there exists a sequence $\{t_k\}$ such that $\lim_{k\to\infty} D^\pi(t_k)/t_k = l$. Hence, there exists a fluid limit $\bar{D}(\cdot)$ such that $\bar{D}(t) = \lim_{q\to\infty} tD^\pi(qt)/qt = tl$, contradicting (56). So we have

$$\limsup_{t\to\infty} \frac{D^\pi(t)}{t} = \mu^*(\lambda)(1-\epsilon).$$

This completes the proof for Theorem 4.1, and part (a) of Theorem 3.1 follows. □

Proof of Theorem 4.2. By Steps 3 and 6 of the generalized round robin policy π in Section 4.2, we obtain an alternative optimal solution to the LP (13) − (17). By inflating some $\delta^*_{j,k}$, we are relaxing some of the bounds in (14) and (15) and also making each of the constraints in (16) tight. Let $\bar{d}^*_k$ be the corresponding departure rates with allocation $\bar{\delta}^*_{j,k}$, see (12). Then we see that $\bar{d}^*_k \geq d^*_k$, hence this feasible solution also achieves the optimal. From now on, we will refer to the alternative LP solution $\bar{d}^*_k$, $\bar{\delta}^*_{j,k}$ as d^*_k, $\delta^*_{j,k}$.

As a result of the policy π, each server spends exactly the same amount of time at any class during each cycle of visiting the classes in its list. Let $I_{j,k}(t)$ be the cumulative idle time for server j at class k, and $\bar{I}_{j,k}(t)$ the corresponding fluid limit. Then the fluid model for the queueing network under the server allocation policy of Section 4.2 satisfies the equations (34) − (36) subject to the conditions

$$0 \leq \sum_{k=1}^{K} \bar{T}_{j,k}(t) \leq t, \ j = 1, \ldots, M,$$

$$\sum_{k=1}^{K} \bar{T}_{j,k}(t) + \bar{I}_{j,k}(t) = t, \ j = 1, \ldots, M,$$

$\bar{T}_{j,k}(0) = 0$, $\bar{I}_{j,k}(0) = 0$, and $\bar{T}_{j,k}(\cdot)$ and $\bar{I}_{j,k}(\cdot)$ are non-decreasing for $j = 1, \ldots, M, k = 1, \ldots, K$,

$$\bar{Q}_k(t) \geq 0, \text{ and } \bar{Q}_k(t)\frac{d\bar{I}_{j,k}(t)}{dt} = 0, \ k = 1, \ldots, K, \tag{57}$$

$$\frac{d\bar{T}_{j,k}(t)}{dt} + \frac{d\bar{I}_{j,k}(t)}{dt} = \frac{h^\pi_{j,k}}{s^\pi_j + \sum_{i\in V^\pi_j} h^\pi_{j,i}}, \ j = 1, \ldots, M, \ k = 1, \ldots, K, \text{ and}$$

$$\frac{d\bar{T}_{j,k}(t)}{dt} = \frac{h^\pi_{j,k}}{s^\pi_j + \sum_{i\in V^\pi_j} h^\pi_{j,i}}, \ j = 1, \ldots, M, \ k = 1, \ldots, K, \text{ whenever } \bar{Q}_k(t) > 0.$$

The second constraint in (57) means that $\bar{I}_{j,k}(t)$ can only increase when $\bar{Q}_k(t)$ is zero. When the amount of fluid at a given class k is positive, the fluid level is decreased at a constant rate by each server j such that $h^\pi_{j,k} > 0$. Then $\sum_{j:k\in V^\pi_j} (h^\pi_{j,k}\mu_{j,k})/(s^\pi_j + \sum_{i\in V^\pi_j} h^\pi_{j,i})$ is the total rate at which the fluid level at class k is decreased whenever $\bar{Q}_k(t) > 0$ for $k = 1, \ldots, K$.

Next consider a fixed server system with K servers operating under the non-idling FCFS service discipline, external arrival rate λ, and routing probabilities given in the matrix P. Assume that server k is assigned to class k, and set the service rates μ_k^π of the servers as

$$\mu_k^\pi = \sum_{j:k\in V_j^\pi} \frac{h_{j,k}^\pi \mu_{j,k}}{s_j^\pi + \sum_{i\in V_j^\pi} h_{j,i}^\pi}, k = 1, \ldots, K.$$

This fixed server system has fluid limits $\{\bar{A}_k(t), \bar{D}_k(t), \bar{Q}_k(t), \forall k\}$ satisfying the same properties as the multi-class system with M servers operating under the policy of Section 4.2. Since we have the same routing matrix P for both systems, the fluid limits for the throughput $\bar{D}(t) = \sum_{k=1}^K \bar{D}_k(t)p_{k,0}$, and hence the throughput of the original system, are equal.

To analyze the throughput in the fixed server system, we proceed as in Chen and Mandelbaum [13]. Let a_k and d_k be the arrival and departure rates (defined as the inflow and outflow capacities in [13]) at server k, with the corresponding vectors A and D. Let μ be the K-dimensional processing capacity, with k^{th} element μ_k^π. Then A, D, μ, and the external arrival rate vector E with $E_k = \lambda p_{0,k}$, $k = 1, \ldots, K$, satisfy the traffic equations

$$A = E + P'D, \tag{58}$$

$$D = A \wedge \mu, \tag{59}$$

where $\wedge$ denotes the componentwise minimum. We know from Section 3.2 that (58) – (59) has a unique solution for A and D, when μ is given. The throughput of the fixed server system $\tau(\mu)$ as a function of the processing capacity μ is given by

$$\tau(\mu) = \sum_{i=1}^K d_i p_{i,0} = e'(I - P')(A \wedge \mu),$$

where e is the K-dimensional unit vector, see page 426 of Chen and Mandelbaum [13].

Now, $\tau(\mu)$ is a nondecreasing function of the processing capacity μ (see page 427 of Chen and Mandelbaum [13]). Let μ^* be the vector of processing capacities corresponding to the optimal allocations, so that the k^{th} entry of μ^* is $\mu_k^* = \sum_{j=1}^M \mu_{j,k}\delta_{j,k}^*$, $k = 1, \ldots, K$. Then we see that (58)–(59) are satisfied for $a_k = a_k^*$ and $d_k = d_k^*$. Hence the maximum throughput for the fixed server system with processing capacity μ^* is given by $\tau(\mu^*) = \sum_{i=1}^K d_i^* p_{i,0} = \mu^*(\lambda)$. By Proposition 4.1, we have $\mu \geq \mu^*(1-\epsilon)$ so that $\tau(\mu) \geq \tau(\mu^*(1-\epsilon))$. We claim that for the fixed server system with processing capacity $\mu^*(1-\epsilon)$, the throughput $\tau(\mu^*(1-\epsilon))$ is at least $\mu^*(\lambda)(1-\epsilon)$. This follows because we have

$$d_k^*(1-\epsilon) \leq \mu_k^*(1-\epsilon), \tag{60}$$

$$d_k^*(1-\epsilon) \leq \lambda(1-\epsilon)p_{0,k} + (1-\epsilon)\sum_{i=1}^K d_i^* p_{i,k} \leq \lambda p_{0,k} + \sum_{i=1}^K d_i^*(1-\epsilon)p_{i,k}. \tag{61}$$

Inequality (60) follows from (14) and (61) follows from (15). But then $d'_k = d^*_k(1-\epsilon)$ is a feasible solution for the allocation LP (13) − (17) with fixed servers having processing capacity $\mu^*(1-\epsilon)$, and d_k is the optimal solution. Hence, we have $\tau(\mu) \geq \tau(\mu^*(1-\epsilon)) = \sum_{k=1}^{K} d_k p_{k,0} \geq \sum_{k=1}^{K} d'_k p_{k,0} = \mu^*(\lambda)(1-\epsilon)$ as required. □

References

[1] H.-S. Ahn, I. Duenyas, and M. E. Lewis. The optimal control of a two-stage tandem queueing system with flexible servers. *Probability in the Engineering and Informational Sciences*, 16:453–469, 2002.

[2] H.-S. Ahn, I. Duenyas, and R. Zhang. Optimal stochastic scheduling of a two-stage tandem queue with parallel servers. *Advances in Applied Probability*, 16:453–469, 1999.

[3] H.-S. Ahn, I. Duenyas, and R. Zhang. Optimal control of a flexible server. *Advances in Applied Probability*, 36:139–170, 2004.

[4] S. Andradóttir and H. Ayhan. Throughput maximization for tandem lines with two stations and flexible servers. *Operations Research*, 53:516–531, 2005.

[5] S. Andradóttir, H. Ayhan, and D. G. Down. Server assignment policies for maximizing the steady-state throughput of finite queueing systems. *Management Science*, 47:1421–1439, 2001.

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

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

[8] S. Andradóttir, H. Ayhan, and D. G. Down. Dynamic assignment of dedicated and flexible servers in tandem lines. *Probability in the Engineering and Informational Sciences*, 21:497–538, 2007.

[9] S. L. Bell and R. J. Williams. Dynamic scheduling of a system with two parallel servers in heavy traffic with complete resource pooling: Asymptotic optimality of a continuous review threshold policy. *Annals of Applied Probability*, 11:608–649, 2001.

[10] S. L. Bell and R. J. Williams. Dynamic scheduling of a parallel server system in heavy traffic with complete resource pooling: Asymptotic optimality of a threshold policy. *Electronic Journal of Probability*, 10:1044–1115, 2005.

[11] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, Belmont, MA, second edition, 1997.

[12] M. Bramson and R. J. Williams. On dynamic scheduling of stochastic networks in heavy traffic and some new results for the workload process. In *Proceedings of the 39th IEEE Conference on Decision and Control*, pages 516–521, 2000.

[13] H. Chen and A. Mandelbaum. Discrete flow networks: Bottleneck analysis and fluid limit approximations. *Mathematics of Operations Research*, 16:408–445, 1991.

[14] H. Chen and A. Mandelbaum. Stochastic discrete flow networks: Diffusion approximations and bottlenecks. *The Annals of Probability*, 19(4):1463–1519, 1991.

[15] J. G. Dai. On positive Harris recurrance of multiclass queueing networks: A unified approach via fluid limit models. *Annals of Applied Probability*, 5:49–77, 1995.

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

[17] J. G. Dai. *Stability of Fluid and Stochastic Processing Networks*, volume 9 of *MaPhySto Miscellanea Publications*. Centre for Mathematical Physics and Stochastics, Ny Munkegade, Denmark, 1999.

[18] J. G. Dai and W. Lin. Maximum pressure policies in stochastic processing networks. *Operations Research*, 53:197–218, 2005.

[19] M. H. A. Davis. Piecewise deterministic Markov processes: A general class of diffusion stochastic models. *Journal of Royal Statistics Society: Series B*, (46):353–388, 1984.

[20] R. Egorova, S. Borst, and B. Zwart. Bandwidth-sharing networks in overload. *Performance Evaluation*, 64(9-12):978–993, 2007.

[21] T. M. Farrar. Optimal use of an extra server in a two station tandem queueing network. *IEEE Transactions on Automatic Control*, 38:1296–1299, 1993.

[22] L. Georgiadis and L. Tassiulas. Optimal overload response in sensor networks. *IEEE/ACM Transactions on Networking*, 14:2684–2696, 2006.

[23] J. B. Goodman and W. A. Massey. The non-ergodic Jackson network. *Journal of Applied Probability*, 21:860–869, 1984.

[24] B. Hajek. Optimal control of interacting service stations. *IEEE Transactions on Automatic Control*, 29:491–499, 1984.

[25] J. M. Harrison. Stochastic networks and activity analysis. Suhov, Yu. M. (ed.), *Analytic Methods in Applied Probability: In Memory of Fridrikh Karpelevich.* American Mathematical Society, Providence, RI, 2002.

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

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

[28] J.M. Harrison. A broader view of Brownian networks. *The Annals of Applied Probability*, 13:1119–1150, 2001.

[29] M. Jonckheere, R. D. van der Mei, and W. van der Weij. Rate stability and output rates in queueing networks with shared resources. *Performance Evaluation*, 67(1):28–42, 2010.

[30] F. Kelly and C. Laws. Dynamic routing in open queueing networks. *Queueing Systems*, 13:47–86, 1993.

[31] A. Kopzon, Y. Nazarathy, and G. Weiss. A push—pull network with infinite supply of work. *Queueing Systems*, 62(1-2):75–111, 2009.

[32] G. F. Lawler. *Introduction to Stochastic Processes.* CRC Press, second edition, 2006.

[33] Y. Nazarathy and G. Weiss. Near optimal control of queueing networks over a finite time horizon. *Annals of Operations Research*, 170:233–249, 2009.

[34] Y. Nazarathy and G. Weiss. Positive Harris recurrence and diffusion scale analysis of a push-pull queueing network. *Performance Evaluation*, 67(4):201 – 217, 2010.

[35] D. G. Pandelis and D. Teneketzis. Optimal multiserver stochastic scheduling of two interconnected priority queues. *Advances in Applied Probability*, 26:258–279, 1994.

[36] Z. Rosberg, P. P. Varaiya, and J. C. Walrand. Optimal control of service in tandem queues. *IEEE Transactions on Automatic Control*, 27(3):600–609, 1982.

[37] D. Shah and D. Wischik. Fluid models of switched networks in overload. Working Paper, 2011.

[38] D. Shah and D. Wischik. The teleology of scheduling algorithms for switched networks under light load, critical load, and overload. Submitted to *Annals of Applied Probability*.

[39] L. Tassiulas and L. L. Bhattacharya. Allocation of interdependent resources for maximal throughput. *Stochastic Models*, 16:27–48, 2000.

[40] L. Tassiulas and A. Ephrimedes. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37:1936–1948, 1992.

[41] S. Tekin. *Efficient System Design: Stability and Flexibility.* PhD thesis, Georgia Institute of Technology, 2011.

[42] S. Tekin and S. Andradóttir. Inspection location in capacity-constrained serial lines. Working Paper, 2011.

[43] G. Weiss. Jackson networks with unlimited supply of work. *Journal of Applied Probability*, 42(3):879–882, 2005.

[44] R. J. Williams. On dynamic scheduling of a parallel server system with complete resource pooling. *Analysis of Communication Networks: Call Centers, Traffic and Performance*, 28, 2000.

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