Compensating for Failures with Flexible Servers

Sigrún Andradóttir and Hayriye Ayhan
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 21, 2006

Abstract

We consider the problem of maximizing capacity in a queueing network with flexible servers, where the classes and servers are subject to failure. We assume that the interarrival and service times are independent and identically distributed, that routing is probabilistic, and that the failure state of the system can be described by a Markov process that is independent of the other system dynamics. We find that the maximal capacity is tightly bounded by the solution of a linear programming problem and that the solution of this problem may be used to construct timed generalized round robin policies that approach the maximal capacity arbitrarily closely. We then give a series of structural results for our policies, including identifying when server flexibility can completely compensate for failures and when the implementation of our policies may be simplified. We conclude with a numerical example that illustrates some of the developed insights.

1 Introduction

The study of queueing systems with flexible servers has attracted significant interest in the research community. Such systems are typically described by a queueing network in which the *class* of a customer is used to track its progress through the network, the customer potentially switching classes many times before leaving the network. The servers are flexible in that they may be capable of serving different classes in the network.

In this work, we identify a tight upper bound on the system capacity in the face of server and/or class failures (a particular service policy π has capacity λ^π if the system is stable for all values of the arrival rate $\lambda < \lambda^\pi$). We then provide an algorithm to construct server assignment policies with performance arbitrarily close to this bound. We assume that the service times and interarrival times are independent and identically distributed (i.i.d.). Customer routing is random and the network is open (all customers eventually leave the network). In the case where there are only server failures, we allow i.i.d. server switching

times. Finally, the dynamics of the failure/repair process are assumed to be described by a Markov process that is independent of the remainder of the system dynamics.

The notion of server failures should be a familiar one. We introduce the notion of class failures to model failures of resources that the servers employ and also to model correlations between different failures (of the servers and resources). The failure model is given in detail in Section 2.3, but at this point we motivate the notion of class failures with an example. Consider a model consisting of several workers in parallel, each manning a workstation. Each worker processes two different kinds of requests, each involving a different database. The workstations all have identical software suites. Requests that use the first database can be considered one class of customers, while those using the second database are another class. If one of the databases becomes corrupted, this would correspond to a class failure because none of the workers could work on that database until it is repaired. They all could continue working on requests that use the other database, assuming that it is operational. If the databases were housed on the same server, then both classes would simultaneously fail if the server went down.

In an earlier work in this research project, Andradóttir, Ayhan, and Down [7] found the maximal capacity for a similar model without failures. The techniques used in that work are adapted here. In particular, we follow the same steps in constructing server assignment policies, i.e., we use the following two step procedure:

1. Solve a linear program (the allocation LP) that yields the maximal capacity and an optimal allocation of each server's effort;

2. Calculate parameters for a timed generalized round robin policy.

We find in this work that the policies developed are significantly more complicated than those in [7], because the presence of failures forces us to consider policies that depend on the current failure state of the network. As in [7], the analysis will proceed by characterizing a formal limiting fluid model for the system. The use of fluid limits to characterize stability of a queueing network is a now standard technique, its genesis being in the work of Rybko and Stolyar [35] and Dai [16]. The identification of the maximal capacity in the presence of server and class failures and how it may be achieved is the first contribution of this paper, and can be seen as an extension of [7].

The work in this paper allows us to compute the maximal capacity in a system with failures and of course the corresponding system without failures. It would therefore be of interest to develop conditions under which the gap between these two capacities can be eliminated, i.e., how one may be able to construct a flexibility structure to eliminate the effects of failures. Also, from the implementation side, it would be of interest to find the

simplest possible policies that still can guarantee reasonable performance. These issues are both studied in the latter half of the paper. This is the second contribution of our work, which we see as entirely new because it gives insight into when and how flexibility can compensate for failures.

The design of server assignment policies to maximize throughput has been studied in several papers. Tassiulas and Ephremides [40] and Tassiulas and Bhattacharya [39] examine a similar problem to that in [7]. The work in [39], which generalizes that in [40], is concerned with a queueing network having the same topology as ours, however the arrivals follow a Poisson process and the server flexibility structure is less general. Andradóttir, Ayhan, and Down [6] and Andradóttir and Ayhan [5] examine tandem lines with limited buffers under manufacturing blocking. Bischak [13], Zavadlav, McClain, and Thomas [42], and Ostolaza, McClain, and Thomas [33] study the throughput of systems where flexibility occurs across different zones, showing that high utilization is possible even with small buffer sizes. Ahn, Duenyas, and Zhang [2] examine the problem of minimizing the expected holding cost in a system with no exogenous arrivals, two parallel queues, one flexible server, and one fixed server. The same authors in [3] study a similar problem where the two queues are in tandem rather than in parallel, and recently, Ahn, Duenyas, and Lewis [1] study this same tandem system with Poisson arrivals. For more general tandem systems, Bartholdi and Eisenstein [9] introduce bucket brigade policies for deterministic systems, with a stochastic version of the same model being the subject of Bartholdi, Eisenstein, and Foley [10]. Finally, the use of specific flexibility structures on a set of existing servers to enhance the system's performance has been the subject of several papers, including Gurumurthi and Benjaafar [22], Hopp, Tekin, and van Oyen [26], and Sheikhzadeh, Benjaafar, and Gupta [36] (see also Jordan and Graves [28] for related work). A more extensive overview of the literature, including discussions of performance objectives other than throughput, is provided in both [6] and, more recently, in Hopp and van Oyen [27].

The discussion in the previous paragraph shows that queueing systems with flexible servers have been an active research area for some time. The important topic of using server flexibility to compensate for uncertainty in the environment in which the system operates is a topic that appears to have received little attention. Recently, Bambos and Michialidis [8] examined the problem of allocating a single resource (bandwidth) to maximize throughput in a system of parallel queues operating in a random environment (see also the references in [8] for related work on such models). Rather than considering a single server model, we examine a very general network structure, and by dealing specifically with failures, provide extensive structural results.

Here, we use fluid limits to design server assignment policies, as in [7]. In addition to [7],

a recent work by Dai and Lin [18] shows that for a class of stochastic processing networks (which include an instance of the class studied in [7]), maximum pressure policies may be used to maximize network capacity. A little less closely related, but still in the spirit of our approach, is a body of work that uses diffusion approximations to design policies that minimize the long-run discounted holding cost in flexible server systems (Williams and co-authors [11, 12, 14], Harrison and co-authors [23, 24, 25], Laws [30], and Mandelbaum and Stolyar [31]). In these works, the notion of heavy traffic is defined through an LP similar to that considered here. Their main focus is on the Brownian control problem that results and hence the network topologies are fairly simple. Squillante et al. [38] study a similar model, but base their observations on results of simulations. However, none of the references in this paragraph deal with the impact of failures.

There is an extensive literature on evaluating the performance of systems of tandem queues with unreliable servers (that cannot move between classes), most of which is concerned with the impact of finite buffers. Excellent overviews can be found in Chapter 6 of Buzacott and Shanthikumar [15] and the survey paper of Dallery and Gershwin [20]. There are also results that use LP bounds to design policies that minimize holding costs in systems where a server with a controllable processing rate is subject to failures, see for example Akella and Kumar [4], Perkins and Srikant [34], and the references therein. However, these results are only for single-server systems, so the impact of flexibility is not considered. In contrast to these references, because we focus on capacity rather than holding cost, we are able to look at an arbitrarily large number of classes and servers and a general failure model. In addition, we do not control the processing rate when a server is at a class (it is fixed).

In Section 4.4 of Dai and Meyn [19], a fluid model is developed for a single-class, single-server system subject to failures. We examine a much more complex failure model, but the spirit of the derivation of the fluid model for our system is similar to that in [19]. Also, as the work in [19] is for a single server, the focus is on incorporating failure models into the fluid limit methodology, rather than on how server flexibility can compensate for failures, which is the focus of our work. While this work was being completed, we became aware of an unpublished paper by Wu et al. [41]. They look at a model with two classes, a general number of dedicated servers at each class, and a general number of flexible servers. The dedicated servers may fail (but not the classes or the flexible servers) and there are no arrivals. The objective is to minimize holding cost. However, in [41], the focus does not appear to be on how server flexibility can compensate for failures.

The organization of this paper is as follows. Section 2 gives a detailed model description. Section 3 provides the means to calculate the maximal capacity, via the allocation LP, with the main result being contained in Theorem 1. An algorithm is also presented to construct

server scheduling policies that are guaranteed to have capacity arbitrarily close to the maximal capacity. Section 4 identifies situations in which the gap between maximal capacities for systems with and without failures may be eliminated by choosing an appropriate flexibility structure. Section 5 suggests that the complexity of the policies may be decreased if there are only server failures or if the failures and repairs occur on a much slower time scale than the cycle time of a server. We also show that the server assignment policies proposed are not very sensitive to changes in the failure state of the system. Section 6 considers a specific example to illustrate some of the insights developed earlier. Finally, Section 7 provides concluding remarks.

2 Queueing network model

2.1 Routing structure

Customers arrive to the system according to a renewal process with i.i.d. interarrival times $\{\xi(n)\}$. The associated arrival rate is $\lambda = 1/E[\xi(1)]$. There are K customer classes, with an infinite buffer for each class. An external arrival is routed to class k with probability $p_{0,k}$, where $\sum_{k=1}^{K} p_{0,k} = 1$.

Upon completion of service at class i, a customer becomes one of class k with probability $p_{i,k}$ or leaves the system with probability $1 - \sum_{k=1}^{K} p_{i,k}$. Let the routing matrix P be a $K \times K$ matrix with entries $p_{i,k}$ and I be the $K \times K$ identity matrix. We assume that all customers eventually leave the system, which is equivalent to the existence of $(I - P')^{-1}$, where $'$ denotes matrix transpose.

We could generalize our results to more general routing (i.e., non-random), at the cost of a more complicated Markov process description, a weaker notion of stability, or stronger assumptions. Our structural insights do not depend on the routing structure (other than through the proportion of customers routed from one class to another) and consequently we continue with the probabilistic routing assumptions for simplicity.

2.2 Service mechanism

There are a total of M servers, each capable of working on at least one class. Service within a class is First Come, First Served (FCFS). At any point in time if there are multiple servers at a class, then we assume that they either combine their efforts on a single customer or work in parallel on different customers. Some more details on each variation now follow.

Parallel servers. Servers work independently (in parallel) at each class. We assume that a preempted service may be resumed by any server, with the service requirement depending only on the class. In other words, the nth customer served at class k has service requirement $\eta_k(n)$. The sequence $\{\eta_k(n)\}$ is assumed to be i.i.d. for each class k, where without loss of generality, we assume $E[\eta_k(1)] = 1$. When server j is processing class k customers, the remaining service requirement is reduced at rate $\mu_{j,k} < \infty$, with $\mu_{j,k} = 0$ if server j is not capable of working at class k. To this rate we associate a mean service time $m_{j,k} = 1/\mu_{j,k}$. Without loss of generality, we assume that for $j = 1, \ldots, M$, there exists $k \in \{1, \ldots, K\}$ such that $\mu_{j,k} > 0$ (otherwise, server j may be removed from the set of available servers).

Cooperating servers. Here, if there are multiple servers at a class, they pool their efforts on one customer. We make the same stochastic assumptions as in the parallel server case, and also assume that the preempted service may be resumed by any set of servers. The servers may leave or join a class independent of the location of the remaining servers and we assume that the service rates are additive when there are multiple servers at a class.

There is now enough of a framework to give a partial description of the system dynamics. For the remainder of the paper, we will concentrate on the cooperating servers case. The results are exactly the same for the parallel servers case, differing only slightly in the technical details of the proofs. Let $E(t)$ be the number of arrivals from outside of the system in the time interval $(0, t]$, $S_k(t)$ be the potential number of service completions by a server working at rate one at class k in $(0, t]$, $\Phi_{i,k}(n)$ be the number of customers that have arrived to class k from class i amongst the first n customers passing through i, $A(t)$ be the residual interarrival time at time t, $Y_k(t)$ be the residual service requirement for class k at time t, and $\phi_{i,k}(n)$ be independent random variables that have value one with probability $p_{i,k}$ and are equal to zero otherwise. We can write

$$\begin{aligned} E(t) &= \max\{n \geq 1 : A(0) + \xi(1) + \cdots + \xi(n-1) \leq t\}, \\ S_k(t) &= \max\{n \geq 1 : Y_k(0) + \eta_k(1) + \cdots + \eta_k(n-1) \leq t\}, \\ \Phi_{i,k}(n) &= \sum_{\ell=1}^{n} \phi_{i,k}(\ell), \end{aligned}$$

where the maximum of an empty set and sum over an empty set are both defined to be zero. The queue length dynamics are defined in terms of the above three processes and $T_{j,k}(t)$, the cumulative time that server j has spent on class k customers in $(0, t]$. In particular, the queue length $Q_k(t)$ of class k at time t (including the customer in service, if any) for

$k = 1, \ldots, K$ and $t \geq 0$ evolves as

$$Q_k(t) = Q_k(0) + \Phi_{0,k}(E(t)) + \sum_{i=1}^{K} \Phi_{i,k} \left(S_i \left(\sum_{j=1}^{M} \mu_{j,i} T_{j,i}(t) \right) \right) - S_k \left(\sum_{j=1}^{M} \mu_{j,k} T_{j,k}(t) \right). \quad (1)$$

The functions $T_{j,k}(t)$ are determined by the server assignment policy, so the above is not an explicit description of the system dynamics. The main focus of the paper is finding appropriate server assignment policies to maximize system capacity.

2.3 Failure Model

We track the availability of resources in the system through $W_1(t) = (u(t), v(t))$, where $u(t) \subseteq \{1, \ldots, M\}$ and $v(t) \subseteq \{1, \ldots, K\}$ give the sets of servers and classes that are functioning at time t, respectively. We assume that there exists a Markov process $\{W(t)\}$ whose first component is $W_1(t)$. The other components of $W(t)$ allow additional generality and could include terms for residual failure and repair times, dependencies amongst components, etc. We also assume that the process $\{W(t)\}$ is independent of the remainder of the system behavior (the arrival, service, and routing processes, the queue length processes, and the server locations) and satisfies the strong Markov property. The set of possible states for $W_1(t)$ is given by $U \times V$, where U is the set of all subsets of $\{1, \ldots, M\}$ and V is the set of all subsets of $\{1, \ldots, K\}$. Let $\mathbf{1}\{\cdot\}$ be the indicator function. It is convenient to introduce the notation

$$B_{u,v}(t) = \int_0^t \mathbf{1}\{W_1(s) = (u, v)\} ds \quad (2)$$

to denote the total time spent in failure state (u, v) up to time t. We assume

$$a_{u,v} = \lim_{t \to \infty} \frac{B_{u,v}(t)}{t} \quad (3)$$

exists with probability one for all $u \in U$, $v \in V$, where $a_{u,v}$ can be interpreted as the long-run proportion of time spent in state (u, v).

It may be useful to note at this point that our failure/repair model is very general. In particular, correlation between failures and repairs may occur, for example there may be events that simultaneously bring down a set of servers and classes. It is only crucial that (3) holds, which involves only the proportions of time a server or class is in the working state. Also, note that $W_1(t)$ is the only part that will be directly required for the stability analysis. The remaining components of $W(t)$ are only needed to ensure that the overall system is Markovian.

We have given one example of class failures in the Introduction. To end this section, we give an additional example to illustrate our choice of model. Suppose that we have a

manufacturing system with four processing steps on three machines. The first and fourth steps are on the first machine, the second step is on the second machine, and the third step is on the third machine. There are two workers, the first is able to work at the first two machines, the second at the last two machines. This would correspond in our notation to a network with $K = 4$ classes and $M = 2$ servers. We would also have $\mu_{1,3} = \mu_{2,1} = \mu_{2,4} = 0$, with all other values of $\mu_{j,k}$ being non-zero.

Now, suppose that the machines fail independently of each other and the workers fail (i.e., are unavailable to do any processing) independently of each other and of the machines. Times to failure and times to repair are i.i.d. and mutually independent. This would be a case of correlated failures, because classes one and four are either simultaneously functioning or not. In this case

$$W(t) = (W_1(t), R_{1,m}(t), R_{2,m}(t), R_{3,m}(t), R_{1,w}(t), R_{2,w}(t)),$$

where $R_{i,m}(t)$ is the residual failure or repair time (as appropriate) for machine $i \in \{1, 2, 3\}$, and $R_{j,w}(t)$ is the residual failure or repair time for worker $j \in \{1, 2\}$ (the residual failure and repair times are only required when these times are not exponentially distributed). Note that if we were to assume that the first and fourth steps of the operation required different sets of resources, then class one could fail while class four is still operational.

2.4 Stochastic assumptions

To complete the model description, we require two technical assumptions for the arrival process: there exists some non-negative function $q(x)$ on $\mathbb{R}_+$ and some integer ℓ such that

$$\mathrm{P}(\xi(1) \geq x) > 0 \quad \text{for all} \quad x > 0,$$

$$\mathrm{P}(\xi(1) + \cdots + \xi(\ell) \in dx) \geq q(x)dx \quad \text{and} \quad \int_0^\infty q(x)dx > 0.$$

As discussed in [7], these two assumptions are made for technical reasons, allowing the origin to be reachable for the Markov process that describes the network dynamics (it is required in Theorem 4.2 of Dai [16], which we use in the proof of Theorem 1). These two assumptions may be relaxed; all that is necessary is that a reachable compact set exists for the Markov process (see Meyn and Down [32] for this argument and Sigman [37] for examples where these assumptions may be relaxed).

3 Main Results

3.1 The allocation LP

We first solve the following set of *traffic equations* for $k = 1, \ldots, K$:

$$\lambda_k = p_{0,k}\lambda + \sum_{i=1}^{K} p_{i,k}\lambda_i.$$

If the system is stable, then λ_k is the long-run arrival rate to class k. As $(I - P')$ is invertible, the traffic equations have a unique solution. When $\lambda = 1$, we will denote this unique solution by $\alpha_1, \ldots, \alpha_K$. Without loss of generality, we assume $\alpha_k > 0$ for $k = 1, \ldots, K$. As a result, $\lambda_k = \lambda \times \alpha_k$, for $k = 1, \ldots, K$, is the unique solution to the traffic equations for any value of λ.

We now wish to use the solution of the traffic equations to solve for the maximal capacity. We first need two definitions. The set $U(j) = \{u \in U : j \in u\}$ gives the set of all u such that server j is functioning and similarly $V(k) = \{v \in V : k \in v\}$ gives the set of all v such that class k is functioning. Consider the following allocation LP, where the decision variables are λ and $\delta_{j,k,u,v}$ for $j = 1, \ldots, M$, $k = 1, \ldots, K$, $u \in U(j)$, $v \in V(k)$ (recall that the $\mu_{j,k}$ are the given service rates for the network and may be zero). The variables $\delta_{j,k,u,v}$ are to be interpreted as the long-run average fraction of time server j is assigned to class k, while the set of servers u and the set of classes v are working.

$$\begin{aligned}
&\max \lambda \\
\text{s.t.} \quad &\sum_{j=1}^{M} \sum_{u \in U(j)} \sum_{v \in V(k)} \delta_{j,k,u,v}\mu_{j,k} \geq \lambda\alpha_k, \text{ for all } k = 1, \ldots, K, && (4)\\
&\sum_{k \in v} \delta_{j,k,u,v} \leq a_{u,v}, \text{ for all } j = 1, \ldots, M,\ u \in U(j),\ v \in V, && (5)\\
&\delta_{j,k,u,v} \geq 0, \text{ for all } k = 1, \ldots, K,\ j = 1, \ldots, M,\ u \in U(j),\ v \in V(k). && (6)
\end{aligned}$$

The constraint (4) says that the service allocation must be at least as large as the arrival rate at each class. The constraints (5) and (6) prevent us from over and under allocating a server, respectively. We will assume that $\delta_{j,k,u,v} = 0$ when $u \notin U(j)$ or $v \notin V(k)$. Later, it will be useful to refer to the above problem as LP(1).

At this point, it is useful to note that if we allowed the service rates to depend on (u, v) (i.e., assume service rates $\mu_{j,k,u,v}$), we would still have an LP to solve to determine the optimal allocations. This could be useful to model such phenomenon as a server failure resulting in the other servers speeding up. However, for ease of notation, we continue with $\mu_{j,k}$, i.e., the service rates are independent of $W_1(t)$. Our results would still hold in the more general setting.

We could also allow for multiple arrival streams, in which case we would have one set of traffic equations for each stream and in (4) we would need to have on the right-hand side the total arrival rate (over all arrival streams). If we were to define the capacity as a weighted average of the arrival rates, then the problem would be the same (the maximal capacity would be a scalar). In general, we would have a capacity *region*, rather than an interval of possible capacity values. We are currently examining this latter situation.

Also, note that as (u, v) is an element of the set $U \times V$ (see Section 2.3), the LP quickly grows in size as the number of classes and/or servers increases. However, it is still an LP, so it is not problematic to solve large instances. Nevertheless, it would be useful to look for special structure to aid in the solution.

Let a solution of this LP be given by λ^*, $\{\delta^*_{j,k,u,v}\}$, where $j = 1, \ldots, M$, $k = 1, \ldots, K$, $u \in U(j)$, and $v \in V(k)$. These values can be interpreted as the maximal capacity and a proportional allocation of server j to class k while in state (u, v) that achieves λ^*, respectively. It is worthwhile to note that the only elements of the failure process that enter the calculation of the maximal capacity are the proportions of time $a_{u,v}$ that the servers in u and classes in v are functioning. In particular, correlations between the up and down times do not appear in the calculation. However, the correlations between the servers and classes that are up or down at any given instant does affect the values $a_{u,v}$ and hence the maximal capacity of the system.

For our candidate policies, we consider the set of timed generalized round robin policies. A timed generalized round robin policy π is given by a set of parameters $\{d^\pi_{j,k,u,v}\}$ and an ordered list of classes $V^\pi_{j,u,v}$. While the system is in state (u, v), server j serves classes in $V^\pi_{j,u,v}$ in cyclic order, with server j spending $d^\pi_{j,k,u,v}$ time units at each class k in $V^\pi_{j,u,v}$ (unless server j idles, in which case the server moves to the next class in $V^\pi_{j,u,v}$). If the classes in $V^\pi_{j,u,v}$ are all empty, the server moves to the next class and idles until one of the classes in $V^\pi_{j,u,v}$ becomes nonempty.

The following proposition will be used to construct a timed generalized round robin policy that comes arbitrarily close to the assignments required by the allocation LP.

Proposition 1 *Let $\mathcal{K}$ be a finite set, and for each $k \in \mathcal{K}$, suppose that s, a, and δ_k satisfy $s \geq 0$, $0 < a \leq 1$, $\delta_k \geq 0$, and $0 < \sum_{k\in\mathcal{K}} \delta_k \leq a$. Then, for any $0 < \varepsilon < 1$, there exists a set of non-negative real numbers $\{d_k\}$, where $k \in \mathcal{K}$, such that*

$$\frac{ad_k}{s + \sum_{i\in\mathcal{K}} d_i} \geq \delta_k(1 - \varepsilon) \quad \textit{for all } k \in \mathcal{K}. \tag{7}$$

Proof. The choice

$$d_k = \frac{(1 - \varepsilon)(s + \beta \mathbf{1}\{s = 0\})\delta_k}{\varepsilon a}$$

for arbitrary $\beta > 0$ and for all $k \in \mathcal{K}$ (so that $d_k = 0$ when $\delta_k = 0$) is one valid choice. $\diamondsuit$

There is a discussion in Section 3.2 of [7] that gives other alternatives for choosing the policy parameters. These ideas may be readily adapted to the model considered in this paper. If $s = 0$, then $d_k = c\delta_k$ for all k satisfies (7) for all $c > 0$.

3.2 Server assignment algorithm

At this point, we assume that the switching time for servers to travel between classes is zero. This will be relaxed when we look at a less general model, that of server failures only, in Section 5.1.

The work in the previous section suggests the following algorithm for dynamic server assignment.

1. Solve the allocation LP.

2. For each server j and each $(u, v) \in U(j) \times V$, specify the list $V^{\pi}_{j,u,v}$ using all of the classes $k \in v$ with $\mu_{j,k}\delta^*_{j,k,u,v} > 0$. Let $\#(V^{\pi}_{j,u,v})$ be the cardinality of this list.

3. For each server j with $\#(V^{\pi}_{j,u,v}) = 1$, set $d^{\pi}_{j,k,u,v} = 1$ for $k \in V^{\pi}_{j,u,v}$.

4. Choose the desired capacity, $\lambda < \lambda^*$, and compute timed generalized round robin policy parameters $d^{\pi}_{j,k,u,v}$ for each j, k, $(u, v) \in U(j) \times V(k)$ satisfying Proposition 1 with $p = \lambda/\lambda^*$, $\varepsilon = (1-p)/(1+p)$, $a = a_{u,v}$, $s = 0$, $\delta_k = \delta^*_{j,k,u,v}$, and $\mathcal{K} = \{k : k \in V^{\pi}_{j,u,v}\}$. Call the resulting timed generalized round robin policy $\pi_{u,v}$.

5. While $W_1(t) = (u, v)$, use $\pi_{u,v}$ in a preemptive-resume manner. In other words, when switching from (u, v) to (u', v'), use $\pi_{u',v'}$ initialized to its status when $\{W_1(t)\}$ last left (u', v'). If this is the first time in (u', v'), $\pi_{u',v'}$ may be initialized arbitrarily.

We are now in position to give a theorem that identifies the maximal capacity of the network. Let the queue length vector $Q(t)$ have kth entry $Q_k(t)$ (defined in Section 2.2).

Theorem 1 (i) *Any capacity less than λ^* may be achieved. More specifically, for an arrival process with rate $\lambda < \lambda^*$, a timed generalized round robin policy constructed with the above algorithm ensures that the distribution of the queue length process $\{Q(t)\}$ converges to a steady-state distribution φ as $t \to \infty$.*

(ii) *A capacity larger than λ^* cannot be achieved. More specifically, for an arrival process with rate $\lambda > \lambda^*$, as $t \to \infty$, $\mathrm{P}(|Q(t)| \to \infty) = 1$.*

Proof. We first must construct a Markov process $\{X(t)\}$ for the system. The following process is an appropriate choice:

$$X(t) := (Q_k(t), A(t), Y_k(t), L_{j,u,v}(t), D_{j,u,v}(t), W(t) : \ k = 1, \ldots, K, \ j = 1, \ldots, M, \ (u,v) \in U \times V), \tag{8}$$

where $A(t)$ and $Y_k(t)$ are defined in Section 2.2 and $W(t)$ is the Markov process describing the failure status of the system, defined in Section 2.3. As a reminder, our assumption is that $\{W(t)\}$ is independent of the remaining components of $\{X(t)\}$. While $W_1(t) = (u, v)$, $L_{j,u,v}(t)$ is the location of server j at time t and $D_{j,u,v}(t)$ is the time elapsed in the current visit by server j to its location $L_{j,u,v}(t)$ at time t. The quantities L and D need the subscripts u and v due to the fact that we must use the policy $\pi_{u,v}$ in a preemptive-resume fashion when $W_1(t)$ enters each state (u, v) (note that when $(u', v') \neq W_1(t)$, the corresponding values of these variables are those when $W_1(t)$ last left (u', v'); when $W_1(t)$ moves to (u', v'), the appropriate values are used to "initialize" the system). The residual times $A(t)$ and $Y_k(t)$ are taken to be right continuous and, as in Davis [21], the process $\{X(t)\}$ may be shown to have the strong Markov property.

We use a fluid limit methodology to prove this result. For a general reference see Dai [17], and for more details on how this methodology applies to a flexible server model, see Andradóttir, Ayhan, and Down [7]. To adapt the work in [17] to our failure model, one can think of introducing an additional class to model the failed state for each server, thus remaining in the multiclass framework of [17]. To handle class failures, we simply examine the fluid model during class up times, as made precise below. If we let $q = \sum_{k=1}^{K} Q_k(0)$, and suppose that the function $(\bar{Q}_k(\cdot), \bar{T}_{j,k}(\cdot) : k = 1, \ldots, K, \ j = 1, \ldots, M)$ is a limit point of the functions

$$(q^{-1}Q_k(q\cdot), q^{-1}T_{j,k}(q\cdot) : k = 1, \ldots, K, \ j = 1, \ldots, M) \tag{9}$$

when $q \to \infty$, then we call $(\bar{Q}_k(\cdot), \bar{T}_{j,k}(\cdot) : k = 1, \ldots, K, \ j = 1, \ldots, M)$ a *fluid limit* of the system. Our assumptions on the arrival, service, routing, and failure processes imply that their corresponding fluid limits converge almost surely uniformly on compact sets (a.s. u.o.c.). Then, as in Dai [16], the equations (1) that describe the dynamics of the queueing network can be used to show that the functions in (9) converge a.s. u.o.c. to a fluid limit that satisfies the fluid model equations below.

Let π be a policy chosen according to the server assignment algorithm and let $T_{j,k,u,v}(t)$ be the time that server j has spent on class k when the system is in state (u, v) prior to time t. Clearly we have

$$T_{j,k}(t) = \sum_{u \in U(j)} \sum_{v \in V(k)} T_{j,k,u,v}(t).$$

Define $\bar{T}_{j,k,u,v}(t) = \lim_{q\to\infty} q^{-1} T_{j,k,u,v}(qt)$. Whenever these limits exist, we have

$$\bar{T}_{j,k}(t) = \sum_{u\in U(j)} \sum_{v\in V(k)} \bar{T}_{j,k,u,v}(t).$$

As a reminder, $B_{u,v}(t)$ is the time that the system has been in state (u,v) up to time t (see (2)). If we define $\bar{B}_{u,v}(t) = \lim_{q\to\infty} q^{-1} B_{u,v}(qt)$, we have by (3) that $\bar{B}_{u,v}(t) = t a_{u,v}$ and hence

$$\frac{d}{dt}\bar{B}_{u,v}(t) = a_{u,v}. \tag{10}$$

Then, following the analysis of Section 4 of [7] and using the property that $\pi_{u,v}$ is used in a preemptive-resume manner, we have

$$1 \geq \frac{d\bar{T}_{j,k,u,v}(t)}{d\bar{B}_{u,v}(t)} \geq \frac{d^{\pi}_{j,k,u,v}}{\sum_{i\in V^{\pi}_{j,u,v}} d^{\pi}_{j,i,u,v}}, \tag{11}$$

whenever $\bar{Q}_k(t) > 0$ and the derivative exists. Combining (10) and (11) using the chain rule results in

$$a_{u,v} \geq \frac{d}{dt}\bar{T}_{j,k,u,v}(t) \geq a_{u,v}\frac{d^{\pi}_{j,k,u,v}}{\sum_{i\in V^{\pi}_{j,u,v}} d^{\pi}_{j,i,u,v}}, \tag{12}$$

whenever $\bar{Q}_k(t) > 0$ and the derivative exists.

This result, together with (1), implies that each fluid limit is a solution of the following set of conditions, known as the *fluid model*:

$$\begin{aligned}
\bar{Q}_k(t) &= \bar{Q}_k(0) + p_{0,k}\lambda t + \sum_{i=1}^{K}\sum_{j=1}^{M} p_{i,k}\mu_{j,i}\bar{T}_{j,i}(t) - \sum_{j=1}^{M}\mu_{j,k}\bar{T}_{j,k}(t), \quad 1 \leq k \leq K;\\
\bar{Q}_k(t) &\geq 0, \quad 1 \leq k \leq K;\\
\bar{T}_{j,k,u,v}(0) &= 0 \quad \text{for } j = 1,\ldots,M,\ k = 1,\ldots,K,\ u \in U(j),\ v \in V(k);\\
\bar{T}_{j,k,u,v}(\cdot) &\quad \text{is non-decreasing for } j = 1,\ldots,M,\ k = 1,\ldots,K,\ u \in U(j),\ v \in V(k);
\end{aligned}$$

$$a_{u,v} \geq \frac{d}{dt}\bar{T}_{j,k,u,v}(t) \geq \frac{a_{u,v} d^{\pi}_{j,k,u,v}}{\sum_{i\in V^{\pi}_{j,u,v}} d^{\pi}_{j,i,u,v}}, \text{ whenever } \bar{Q}_k(t) > 0 \text{ and the derivative exists.} \tag{13}$$

For each j, u, and v, using Proposition 1 with $\{\delta^*_{j,k,u,v}\}$, then summing (13) over all j, u and v, and finally using (4) yields

$$\sum_{j=1}^{M}\sum_{u\in U(j)}\sum_{v\in V(k)} \frac{d}{dt}\bar{T}_{j,k,u,v}(t)\mu_{j,k} \geq \lambda_k(1+\varepsilon)$$

whenever $\bar{Q}_k(t) > 0$ and the derivative exists, where as in Section 4.1 of [7], $(1-\varepsilon)/(1+\varepsilon) = \lambda/\lambda^*$ and λ is the desired capacity. That the system empties in a finite time follows from

Theorem 2.4.9 of [17] and thus part (i) of the theorem follows upon applying Theorem 4.2 of [16] which connects the stability of a fluid limit model with that of the original Markov process.

The proof of part (ii) is exactly the same as that of part (ii) of Theorem 1 in [7]. $\diamondsuit$

Finally, we may be interested in calculating the capacity of an arbitrary timed generalized round robin policy π (not necessarily generated by the suggested algorithm). We can bound its capacity as follows. The proof is similar to that of Theorem 3 in [7] and is thus omitted.

Theorem 2 *For a timed generalized round robin policy π with parameters $V^{\pi}_{j,u,v}$ and $d^{\pi}_{j,k,u,v}$, where $j = 1, \ldots, M$, $k = 1, \ldots, K$, $u \in U(j)$, and $v \in V(k)$, the capacity λ^{π} of the system is bounded below and above by*

$$\min_{1 \le k \le K} \frac{1}{\alpha_k} \sum_{j=1}^{M} \sum_{u \in U(j)} \sum_{k \in V(k)} \frac{a_{u,v} d^{\pi}_{j,k,u,v} \mu_{j,k}}{\sum_{i \in V^{\pi}_{j,u,v}} d^{\pi}_{j,i,u,v}} \le \lambda^{\pi} \le \lambda^{*}.$$

3.3 Optimal policy properties

The concept of a bottleneck set of classes was introduced in [7]. Similar to Proposition 1 in [7], if we let Γ^* be a set of classes with the minimum number of tight constraints in (4) and Γ^{**} be a set of (j, u, v) with the minimum number of tight constraints in (5), then we have the following results that we state without proof.

Proposition 2 *1. For all j such that $\mu_{j,k} > 0$ for at least one $k \in \Gamma^*$,*

$$\sum_{k \in v \cap \Gamma^* \cap \{k : \mu_{j,k} > 0\}} \delta^{*}_{j,k,u,v} = a_{u,v}, \quad \text{for all } u \in U(j),\ v \text{ such that } v \cap \Gamma^* \cap \{k : \mu_{j,k} > 0\} \neq \emptyset.$$

2. For all j with $\mu_{j,k} = 0$ for all $k \in \Gamma^*$, $\{\delta^{*}_{j,k,u,v}\}$ can be chosen such that

$$\sum_{k \in v} \delta^{*}_{j,k,u,v} < a_{u,v}, \quad \text{for all } u \in U(j),\ v \in V \text{ such that } a_{u,v} > 0.$$

Proposition 3 *1. For all k with $\delta^{*}_{j,k,u,v} > 0$ for at least one $(j, u, v) \in \Gamma^{**}$,*

$$\sum_{j=1}^{M} \mu_{j,k} \sum_{u' \in U(j)} \sum_{v' \in V(k)} \delta^{*}_{j,k,u',v'} = \lambda^{*} \alpha_k.$$

Moreover, for all j with $(j, u, v) \in \Gamma^{**}$ for at least one (u, v), $\mu_{j,k} = 0$ implies $\delta^{*}_{j,k,u,v} = 0$ for all $(j, u, v) \in \Gamma^{**}$.

2. If $\lambda^* > 0$, then for all k with $\delta^*_{j,k,u,v} = 0$, for all $(j,u,v) \in \Gamma^{**}$, $\{\delta^*_{j,k,u,v}\}$ can be chosen such that

$$\sum_{j=1}^{M} \mu_{j,k} \sum_{u \in U(j)} \sum_{v \in V(k)} \delta^*_{j,k,u,v} > \lambda^* \alpha_k.$$

Part 1 of Proposition 2 states that any server who can serve a class in the bottleneck set of classes Γ^* will be busy serving the bottleneck classes at all times when the state (u,v) of the system is such that this is possible, while part 2 of Proposition 2 states that any server who cannot serve any of the bottleneck classes, can idle some of the time regardless of the state of the system. Similarly, part 1 of Proposition 3 states that bottleneck servers j will spend their time productively in bottleneck states (u,v) and that any class that is served by a bottleneck server in a bottleneck state does not have any extra capacity, while part 2 of Proposition 3 states that any class that is not served by any bottleneck server in any bottleneck state has some excess capacity.

4 Maintaining optimal capacity

We would like to determine when server flexibility can completely compensate for the presence of server and class failures; i.e., the maximal capacity is not decreased. This may guide system designers who have some freedom to choose the flexibility structure of their systems.

We denote the allocation LP without any failures by LP(0), where an optimal solution is given by $\lambda^*(0)$, $\{\delta^*_{j,k}(0)\}$ (note that we may drop u and v in the subscripts):

$$\begin{aligned} & \max \lambda \\ \text{s.t.} \quad & \sum_{j=1}^{M} \delta_{j,k} \mu_{j,k} \geq \lambda \alpha_k, \text{ for all } k = 1, \ldots, K, \\ & \sum_{k=1}^{K} \delta_{j,k} \leq 1, \text{ for all } j = 1, \ldots, M, \\ & \delta_{j,k} \geq 0, \text{ for all } k = 1, \ldots, K, \; j = 1, \ldots, M. \end{aligned} \tag{14}$$

Similarly, we denote the allocation LP with both class and server failures given in Section 3.1 by LP(1). The following proposition specifies under what conditions it is possible to achieve the same capacity in a system with failures as in a system without failures.

Proposition 4 *$\lambda^*(0) = \lambda^*$ if and only if there exists an optimal solution to LP(0) such that for all $j = 1, \ldots, M$ and $\mathcal{K} \subseteq \{1, \ldots, K\}$,*

$$\sum_{k \in \mathcal{K}} \delta^*_{j,k}(0) \leq \sum_{u \in U(j)} \sum_{v \in V : v \cap \mathcal{K} \neq \emptyset} a_{u,v}. \tag{15}$$

Proof. First suppose that $\lambda^* = \lambda^*(0)$. For an arbitrary j, define

$$\delta_{j,k} = \sum_{u \in U(j)} \sum_{v \in V(k)} \delta^*_{j,k,u,v}, \text{ for all } k = 1, \ldots, K.$$

Suppose that $\mathcal{K} \subseteq \{1, \ldots, K\}$. Then we have

$$\begin{aligned} \sum_{k \in \mathcal{K}} \delta_{j,k} &= \sum_{u \in U(j)} \sum_{v \neq \emptyset} \sum_{k \in v} \delta^*_{j,k,u,v} \mathbf{1}\{k \in \mathcal{K}\} \leq \sum_{u \in U(j)} \sum_{v \in V : v \cap \mathcal{K} \neq \emptyset} \sum_{k \in v} \delta^*_{j,k,u,v} \\ &\leq \sum_{u \in U(j)} \sum_{v \in V : v \cap \mathcal{K} \neq \emptyset} a_{u,v}, \end{aligned}$$

where the last inequality follows from (5). Hence (15) holds for j and $\mathcal{K}$. It remains to show that $\{\delta_{j,k}\}$ is an optimal solution to LP(0) using $\lambda^* = \lambda^*(0)$. We have

$$\sum_{j=1}^{M} \delta_{j,k} \mu_{j,k} = \sum_{j=1}^{M} \sum_{u \in U(j)} \sum_{v \in V(k)} \delta^*_{j,k,u,v} \mu_{j,k} \geq \lambda^* \alpha_k = \lambda^*(0) \alpha_k, \tag{16}$$

for all $k = 1, \ldots, K$, where we have used (4), and

$$\begin{aligned} \sum_{k=1}^{K} \delta_{j,k} &= \sum_{k=1}^{K} \sum_{u \in U(j)} \sum_{v \in V(k)} \delta^*_{j,k,u,v} = \sum_{u \in U(j)} \sum_{v \neq \emptyset} \sum_{k \in v} \delta^*_{j,k,u,v} \\ &\leq \sum_{u \in U(j)} \sum_{v \neq \emptyset} a_{u,v} \leq 1, \end{aligned} \tag{17}$$

where we have used (3) and (5). Hence, combining (16) and (17) with (6) implies that $\{\delta_{j,k}\}$ solves LP(0).

To show the other direction, it clearly suffices to show that there exists a non-negative solution $\{\delta_{j,k,u,v}\}$ to the following sets of inequalities:

$$\sum_{u \in U(j)} \sum_{v \in V(k)} \delta_{j,k,u,v} \geq \delta^*_{j,k}(0), \text{ for all } j = 1, \ldots, M,\ k = 1, \ldots, K, \tag{18}$$

$$\sum_{k \in v} \delta_{j,k,u,v} \leq a_{u,v}, \text{ for all } j = 1, \ldots, M,\ u \in U(j),\ v \in V, \tag{19}$$

see equations (4) and (5). (Note that $\lambda^* \leq \lambda^*(0)$ follows from (16) and (17).)

For each $j = 1, \ldots, M$, consider the following network with upper and lower bounds on the flow in each arc. There is a central node c. Corresponding to each class, there is a node labeled k, $k = 1, \ldots, K$. There are arcs from node c, to each node k, with lower bound $\delta^*_{j,k}(0)$ and upper bound ∞. There is also a node corresponding to each possible (u, v) other than the states in which all classes are down ($v = \emptyset$). Label these nodes $s_1, \ldots, s_n$, where n is the cardinality of $U(j) \times (V \backslash \{\emptyset\})$. There is an arc from node $s_i = (u, v)$ to the central

node c, with upper bound $a_{u,v}$ and lower bound 0. Finally there is an arc from node k to node s_i if and only if $k \in v$ (here $u \in U(j)$). These arcs all have lower bound 0 and upper bound ∞.

The system of inequalities (18) and (19) has a feasible solution for j if and only if there is a feasible circulation of flow in this network. By Hoffman's Circulation Theorem (see, e.g., Lawler [29]) this is true if and only if $L(\mathcal{S},\mathcal{T}) \leq U(\mathcal{S},\mathcal{T})$ for all cutsets $\mathcal{S},\mathcal{T}$, where $L(\mathcal{S},\mathcal{T}) = \sum_{i\in\mathcal{T},j\in\mathcal{S}} \ell_{i,j}$ and $U(\mathcal{S},\mathcal{T}) = \sum_{i\in\mathcal{S},j\in\mathcal{T}} c_{i,j}$. The quantities $\ell_{i,j}$ and $c_{i,j}$ are the lower and upper bounds, respectively, on the flows from node i to node j. We consider both possible cases:

1. $\mathcal{T} = \{c\} \cup \mathcal{K} \cup \mathcal{W}$, $\mathcal{K} \subseteq \{1,\ldots,K\}$, $\mathcal{W} \subseteq U(j) \times (V\backslash\{\emptyset\})$. Here

$$L(\mathcal{S},\mathcal{T}) = \sum_{k\in\mathcal{K}^c} \delta^*_{j,k}(0), \quad \text{and}$$

$$U(\mathcal{S},\mathcal{T}) = \begin{cases} \infty & \text{if there exists } k \in \mathcal{K}^c,\ (u,v) \in \mathcal{S} \text{ such that } k \in v, \\ \sum_{(u,v)\in\mathcal{S}^c} a_{u,v} & \text{otherwise,} \end{cases}$$

   where $\mathcal{K}^c$ and $\mathcal{W}^c$ are such that $\mathcal{K} \cup \mathcal{K}^c = \{1,\ldots,K\}$, $\mathcal{W} \cup \mathcal{W}^c = U(j) \times (V\backslash\{\emptyset\})$, $\mathcal{K} \cap \mathcal{K}^c = \emptyset$, and $\mathcal{W} \cap \mathcal{W}^c = \emptyset$. This case is satisfied by (15) (note that if $\forall k \in \mathcal{K}^c$, $(u,v) \in \mathcal{W}$ we have $k \notin v$, then $U(j) \times \{v : v \cap \mathcal{K}^c \neq \emptyset\} \subseteq \mathcal{W}^c$, for otherwise there exists $(u,v) \in \mathcal{W}$ with $v \cap \mathcal{K}^c \neq \emptyset$, a contradiction).

2. $\mathcal{T} \subseteq \{1,\ldots,K\} \cup [U(j) \times (V\backslash\{\emptyset\})]$. Here $L(\mathcal{S},\mathcal{T}) = 0 < \infty = U(\mathcal{S},\mathcal{T})$. $\diamondsuit$

Note that the left side of (15) may be interpreted as the fraction of time server j spends working in $\mathcal{K}$ in the optimal solution $\{\delta^*_{j,k}(0)\}$ to LP(0), while the right side is the fraction of time that server j could spend working in $\mathcal{K}$ in LP(1). For (15) to hold, we must have

$$\sum_{u\in U(j)} \sum_{v\neq\emptyset} a_{u,v} = 1 \tag{20}$$

for any server j such that $\sum_{k=1}^K \delta^*_{j,k}(0) = 1$. Assuming that for all k, there exists a j such that $\mu_{j,k} > 0$, the fact that there exists at least one such server follows for example from Proposition 1 in [7]. If any server j satisfying this condition is subject to failure, then a corresponding decrease in the maximal capacity will occur. On the other hand, if a particular class k fails, it is possible that under a given flexibility structure the condition (15) remains satisfied for all fully utilized servers. Hence Proposition 4 implies that server flexibility is more effective in compensating for class failures than for server failures. We will have more to say on this later in the special case when server and class failures occur independently. On the negative side, (15) implies that $a_{\emptyset,\emptyset} = 0$ and that for all servers j such that $\sum_{k=1}^K \delta^*_{j,k}(0) = 1$, $a_{u,\emptyset} = 0$ for all $u \in U(j)$ and $a_{u,v} = 0$ for all $u \notin U(j)$, $v \in V$.

Now consider a solution to LP(0) with the smallest number of tight constraints in (14). If (14) is tight for j, then it is obvious that $\mu_{j,k} = 0$ implies $\delta^*_{j,k}(0) = 0$. Now let $\mathcal{K} = \{k : \mu_{j,k} > 0\}$. Then (15) implies that $a_{u,v} = 0$ when $u \notin U(j)$ or $v \subseteq \mathcal{K}^c$ (because $\sum_{k\in\mathcal{K}} \delta^*_{j,k}(0) = 1$).

In the remainder of this section, we consider the special case when server and class failures occur independently. More specifically, suppose that there are both server and class failures, but the proportion of time that the system is in state (u, v) is $a_{u,v} = a_u b_v$, where $a_u, b_v \geq 0$ for all $u \in U$ and $v \in V$, $\sum_{u\in U} a_u = 1$, and $\sum_{v\in V} b_v = 1$. For $j = 1, \ldots, M$, let $a_j = \sum_{u\in U(j)} a_u$ be the fraction of time that server j is up. The following result follows immediately from Proposition 4.

Corollary 1 *The presence of independent server and class failures does not decrease the maximal capacity of the system if and only if there exists a solution to LP(0) such that for all $j = 1, \ldots, M$ and for all $\mathcal{K} \subseteq \{1, \ldots, K\}$,*

$$\sum_{k\in\mathcal{K}} \delta^*_{j,k}(0) \leq a_j \sum_{v\in V: v\cap\mathcal{K}\neq\emptyset} b_v.$$

Note that Corollary 1 implies that $a_j = 1$ for any server j such that (14) is tight. Moreover, since there exists a server j such that (14) is tight, Corollary 1 immediately yields that if independent server and class failures do not decrease the maximal capacity of the system, then $b_\emptyset = 0$, where $b_\emptyset$ is the long-run fraction of time that all classes are down. Note that the converse of this result does not hold as shown in the following example. We assume that there is one server and two classes, with $\alpha_1 = \alpha_2 = 1$, $\mu_{1,1} = 2$ and $\mu_{1,2} = 5$. The failure statistics are $a_1 = 1$, $b_{\{1,2\}} = 2/7$, $b_{\{1\}} = 1/7$, $b_{\{2\}} = 4/7$, and $b_\emptyset = 0$. Solving LP(0) yields $\lambda^*(0) = 10/7$. Solving the original allocation LP (LP(1)) gives a maximal capacity of $\lambda^* = 6/7$. Here, class 1 is only up 3/7 of the time and the server works at that class at rate 2, so the best that we can hope for is a maximal capacity of 6/7.

Note that the cases with server failures only or class failures only are special cases of the case with independent server and class failures. In particular, the case with server failures only corresponds to $b_{\{1,\ldots,K\}} = 1$ and the case with class failures only corresponds to $a_{\{1,\ldots,M\}} = 1$. Consequently, Corollary 1 shows that the presence of server failures only does not decrease the maximal capacity of the system if and only if there exists a solution to LP(0) with

$$\sum_{k=1}^{K} \delta^*_{j,k}(0) \leq a_j, \ \forall j = 1, \ldots, M.$$

As before, this seems rather restrictive, because any server j with $\sum_{k=1}^{K} \delta^*_{j,k}(0) = 1$ in the solution to LP(0) with the smallest number of tight constraints in (14) must have $a_j = 1$.

Similarly, Corollary 1 implies that the presence of class failures only does not decrease the maximal capacity of the system if and only if there exists a solution to LP(0) such that

$$\sum_{k\in\mathcal{K}} \delta^*_{j,k}(0) \le \sum_{v\in V: v\cap\mathcal{K}\neq\emptyset} b_v, \ \forall j = 1,\dots,M, \mathcal{K}\subseteq\{1,\dots,K\}. \tag{21}$$

If $a_j \equiv a$, for $j = 1,\dots,M$, then it is a simple matter to compute the maximal capacity λ^* in terms of the maximal capacity $\lambda^*(2)$ of the system with class failures only, and similarly it is easy to express the optimal allocation $\{\delta^*_{j,k,u,v}\}$ in terms of the optimal allocation $\{\delta^*_{j,k,v}(2)\}$ of the system with class failures only. Note that $a_j \equiv a$ for all j means that the servers are homogeneous with respect to the long-run fraction of time that they are failed, but not necessarily with respect to the frequency at which they fail. We have

Proposition 5 *If $a_j \equiv a$, for $j = 1,\dots,M$, then the maximal capacity of the system with class and server failures is $a\lambda^*(2)$, and it is achieved by the allocation $\delta^*_{j,k,u,v} = a_u\delta^*_{j,k,\{1,\dots,M\},v}(2)$ for all j, k, u, and v.*

Proof. Let $\lambda = \lambda^*/a$ and $\delta_{j,k,v} = \sum_{u\in U(j)} \delta^*_{j,k,u,v}/a$ for all j,k, and let LP(2) denote the allocation LP with class failures only. Then it is easy to see that the constraints (4), (5), and (6) of LP(1) imply that the constraints

$$\begin{aligned}
&\sum_{j=1}^{M}\sum_{v\in V(k)} \delta_{j,k,v}\mu_{j,k} \ge \lambda\alpha_k, \text{ for all } k = 1,\dots,K,\\
&\sum_{k\in v} \delta_{j,k,v} \le b_v, \text{ for all } j = 1,\dots,M,\ v\in V,\\
&\delta_{j,k,v} \ge 0, \text{ for all } k = 1,\dots,K,\ j = 1,\dots,M,\ v\in V(k),
\end{aligned}$$

of LP(2) hold with $\lambda = \lambda^*/a$, and consequently that $\lambda^*(2) \ge \lambda^*/a$. Now let $\lambda = a\lambda^*(2)$ and $\delta_{j,k,u,v} = a_u\delta^*_{j,k,v}(2)$ for all j,k,u,v. Then the constraints of LP(2) imply that (4), (5), and (6) hold, and hence that $\lambda^* \ge a\lambda^*(2)$. ♢

Corollary 2 *If $a_j \equiv a$, $j = 1,\dots,M$, then the maximal capacity of the system with server failures only is $a\lambda^*(0)$, and it is achieved by the allocation $\delta^*_{j,k,u,\{1,\dots,K\}} = a_{u,\{1,\dots,K\}}\delta^*_{j,k}(0)$ for all j, k, and u.*

Proposition 5 and Corollary 2 show that when server and class failures occur independently and the servers are homogeneous in that $a_j = a$ for all j, then any server j can be allocated to classes in a manner found to be optimal for the system with class failures only in each state $u \in U(j)$. One might expect that this remains true when server and

class failures occur independently of one another, even if the servers are not homogeneous. In other words, one might expect that $\delta^*_{j,k,u,v} = a_u \delta^*_{j,k,\{1,\ldots,M\},v}(2)$ for all j, k, u, and v, whenever server and class failures occur independently. Unfortunately, this is not true in general as can be seen from the following example. Consider a system with two servers and two classes in tandem ($\alpha_1 = \alpha_2 = 1$), with $\mu_{1,1} = \mu_{1,2} = 2$, $\mu_{2,2} = 1$, and $\mu_{2,1} = 0$. The failure statistics are $a_{\{1,2\}} = p \in [1/2, 1)$, $a_{\{2\}} = 1 - p$, and $b_{\{1,2\}} = 1$, so that $a_1 = p$ and $a_2 = 1$. Since $b_{\{1,2\}} = 1$, it is clear that $\lambda^*(2) = \lambda^*(0)$, and solving the allocation LP with no failures, LP(0), gives $\lambda^*(2) = \lambda^*(0) = 1.5$, $\delta^*_{1,1} = 3/4$, and $\delta^*_{1,2} = 1/4$. However, solving LP(1) yields $\lambda^* = (2p+1)/2$, $\delta^*_{1,1,\{1,2\},\{1,2\}} = (2p+1)/4 \neq 3p/4 = a_1 \delta^*_{1,1}$, and $\delta^*_{1,2,\{1,2\},\{1,2\}} = (2p-1)/4 \neq p/4 = a_1 \delta^*_{1,2}$. Clearly we need to take into account the failure statistics of server 1 when determining its allocation to the two classes.

The above example shows that even when server and class failures occur independently, if the servers are not homogeneous, then it is important to take into account the presence of server failures when allocating the servers to classes. This is obviously also true in the special case where there are only server failures (and no class failures). However, in the next section we will show that the structure of our policies can be simplified when there are only server failures (and no class failures). We will also generalize our model in this case and handle positive server switching times.

Overall, the results in this section demonstrate that server flexibility can compensate for class failures (completely if (15) holds), but that it cannot compensate for failures of key servers (see, e.g., Corollary 2). The intuition is that if a bottleneck server fails, the lost processing time directly decreases the overall capacity (whose calculation assumes all bottleneck servers are busy all of the time), while if a class fails, a working server can potentially go to another class and remain busy. These intuitive effects are quantified in the observations in this section and also in Section 6.

5 Implementation Issues

5.1 Server failures

The fact that we have to keep track of many different policies (i.e., $\pi_{u,v}$, where $u \in U$ and $v \in V$) may be onerous from an implementation standpoint. In the case where only servers fail, the proportion of time spent at a class by server j will be seen to depend only on the server rates, the solution of the traffic equations, and the proportion of time that server j is failed. This will allow us to use only one policy. We may also add switching times in this case. On the nth occurrence of server j switching from class i to class k, a time of

$\zeta^j_{i,k}(n)$ elapses. We assume that the sequence $\{\zeta^j_{i,k}(n)\}$ is i.i.d. for each $j = 1, \ldots, M$, and $i, k = 1, \ldots, K$, and furthermore we assume that $\{\zeta^j_{i,i}(n)\}$ is a sequence of zeros for all i and j. Note that this is the only case where we allow switching times. If there are class failures, a switch may be incurred when a failure or repair occurs, which significantly complicates the analysis.

Recall that a_j is the proportion of time that server j is operational. In this case, the original allocation LP reduces to the following allocation LP, where $\{\delta_{j,k}\}$ is to be interpreted as the long-run average fraction of time server j is assigned to class k.

$$\begin{aligned} & \max \lambda \\ \text{s.t.} \quad & \sum_{j=1}^{M} \delta_{j,k}\mu_{j,k} \geq \lambda\alpha_k, \text{ for all } k = 1, \ldots, K, \\ & \sum_{k=1}^{K} \delta_{j,k} \leq a_j, \text{ for all } j = 1, \ldots, M, \\ & \delta_{j,k} \geq 0, \text{ for all } k = 1, \ldots, K \text{ and } j = 1, \ldots, M. \end{aligned}$$

Let a solution of this LP be given by λ^*, $\{\delta^*_{j,k}\}$, $j = 1, \ldots, M$, $k = 1, \ldots, K$. These values can be interpreted as the maximal capacity and a proportional allocation of server j to class k that achieves λ^*, respectively.

In this case, the server assignment algorithm simplifies to the following.

1. Solve the allocation LP.
2. For each server j, specify the list V^π_j using all of the classes k with $\mu_{j,k}\delta^*_{j,k} > 0$. Define the ith element of each list V^π_j as $v_{j,i}$ and let $\#(V^\pi_j)$ be the cardinality of this list.
3. For each server j with $\#(V^\pi_j) = 1$, set $s^\pi_j = 0$ and $d^\pi_{j,k} = 1$ for $k \in V^\pi_j$.
4. For each server j with $\#(V^\pi_j) > 1$, specify the expected switching time in a cycle of visiting the states in V^π_j in order:
$$s^\pi_j = \sum_{i=1}^{\#(V^\pi_j)-1} E[\zeta^j_{v_{j,i},v_{j,i+1}}(1)] + E[\zeta^j_{v_{j,\#(V^\pi_j)},v_{j,1}}(1)].$$
5. Choose the desired capacity, $\lambda < \lambda^*$, and compute the policy parameters $d^\pi_{j,k}$ using Proposition 1, with $p = \lambda/\lambda^*$, $\varepsilon = (1-p)/(1+p)$, $a = a_j$, $s = s^\pi_j$, $\delta_k = \delta^*_{j,k}$, and $\mathcal{K} = \{k : k \in V^\pi_j\}$.

Here each server j simply uses the timed generalized round robin policy π, resuming from where it left off π when it fails and is subsequently repaired. The following result indicates that this set of policies achieves maximal capacity.

Theorem 3 **(i)** *Any capacity less than λ^* may be achieved. More specifically, for an arrival process with rate $\lambda < \lambda^*$ and a system with server failures only, a timed generalized round robin policy constructed with the above algorithm ensures that the distribution of the queue length process $\{Q(t)\}$ converges to a steady-state distribution φ as $t \to \infty$.*

(ii) *A capacity larger than λ^* cannot be achieved. More specifically, for an arrival process with rate $\lambda > \lambda^*$, as $t \to \infty$, $\mathrm{P}(|Q(t)| \to \infty) = 1$.*

Proof. The proof is very similar to that of Theorem 1, with the following minor changes. First, the Markov process $\{X(t)\}$ can be simplified to:

$$X(t) := (Q_k(t), A(t), Y_k(t), L_j(t), D_j(t), W(t) : \ k = 1, \ldots, K, \ j = 1, \ldots, M),$$

where the changes from $X(t)$ in Theorem 1 are that $L_j(t)$ is the location of server j at time t and $D_j(t)$ is the time elapsed in the current visit by server j to its location at time t (set to zero if the server is idle).

Let π be a policy chosen according to the server assignment algorithm and let $B_j(t)$ be the time that server j has been up in $[0, t)$. If we define $\bar{B}_j(t) = \lim_{q\to\infty} q^{-1} B_j(qt)$, we have in the same manner as (10),

$$\frac{d}{dt}\bar{B}_j(t) = a_j. \tag{22}$$

Following the proof of (12) in Theorem 1, we then conclude

$$a_j \geq \frac{d}{dt}\bar{T}_{j,k}(t) \geq a_j \frac{d^\pi_{j,k}}{s^\pi_j + \sum_{i\in V^\pi_j} d^\pi_{j,i}},$$

whenever $\bar{Q}_k(t) > 0$ and the derivative exists. With this result, we have that the fluid model is:

$$\begin{aligned}
\bar{Q}_k(t) &= \bar{Q}_k(0) + p_{0,k}\lambda t + \sum_{i=1}^{K}\sum_{j=1}^{M} p_{i,k}\mu_{j,i}\bar{T}_{j,i}(t) - \sum_{j=1}^{M}\mu_{j,k}\bar{T}_{j,k}(t), \quad 1 \leq k \leq K; \quad (23)\\
\bar{Q}_k(t) &\geq 0, \quad 1 \leq k \leq K;\\
\bar{T}_{j,k}(0) &= 0 \quad \text{and} \quad \bar{T}_{j,k}(\cdot) \quad \text{is non-decreasing for } j = 1, \ldots, M,\ k = 1, \ldots, K;
\end{aligned}$$

$$a_j \geq \frac{d}{dt}\bar{T}_{j,k}(t) \geq a_j \frac{d^\pi_{j,k}}{s^\pi_j + \sum_{i\in V^\pi_j} d^\pi_{j,i}}, \text{ whenever } \bar{Q}_k(t) > 0 \text{ and the derivative exists.} \tag{24}$$

As we now have an appropriate lower bound on $\frac{d}{dt}\bar{T}_{j,k}(t)$, the remainder of the proof follows as in Theorem 1. ♢

We also have a result analogous to Theorem 2, which we state without proof.

Theorem 4 *For a timed generalized round robin policy π with parameters V_j^π and $d_{j,k}^\pi$ where $j = 1, \ldots, M$ and $k = 1, \ldots, K$, operating in a system with server failures only, the capacity λ^π of the system is bounded below and above by*

$$\min_{1 \le k \le K} \frac{1}{\alpha_k} \sum_{j=1}^{M} \frac{a_j d_{j,k}^\pi \mu_{j,k}}{s_j^\pi + \sum_{i \in V_j} d_{j,i}^\pi} \le \lambda^\pi \le \lambda^*.$$

5.2 Policy insensitivity

In general, we must look at the situation where we track many different policies. It would be useful to demonstrate that the policies are somewhat insensitive to which classes are failed. If we combine this with the asymptotic results of Section 5.3, policies can be designed that minimize server movement upon a failure or repair, while still maintaining high capacity. In particular, in this section we demonstrate that the set of classes that a server visits does not change significantly as the classes fail and are repaired. The following result shows that there is an optimal allocation of servers to classes with the property that if server j is allocated to a class k in state v and some class $k' \neq k$ fails or is repaired, then server j remains allocated to class k.

Proposition 6 *Let λ^*, $\{\delta_{j,k,u,v}^*\}$ be an optimal solution to LP(1) with the smallest number of zeros in the set $\{\delta_{j,k,u,v}^*\}$. Then if u, v, v', j are such that $a_{u,v'} > 0$, $v' \subseteq v$, and $v' \cap \{k : \delta_{j,k,u,v}^* > 0\} \neq \emptyset$, then*

$$\delta_{j,k,u,v'}^* > 0 \Longleftrightarrow \delta_{j,k,u,v}^* > 0 \text{ and } k \in v'. \tag{25}$$

Proof. (1) Assume $\delta_{j,k,u,v'}^* > 0$, implying that $k \in v'$. If $\delta_{j,k,u,v}^* = 0$, then let $\ell \in v' \cap \{k : \delta_{j,k,u,v}^* > 0\}$ and $0 < \delta < \min\{\delta_{j,k,u,v'}^*, \delta_{j,\ell,u,v}^*\}$. Reduce $\delta_{j,k,u,v'}^*$, $\delta_{j,\ell,u,v}^*$ by δ and increase $\delta_{j,k,u,v}^*$, $\delta_{j,\ell,u,v'}^*$ by δ. Then we have an optimal solution with fewer zeros, a contradiction.

(2) Assume $\delta_{j,k,u,v}^* > 0$, $k \in v'$, $\delta_{j,k,u,v'}^* = 0$, and let $\ell \in v' \cap \{k : \delta_{j,k,u,v'}^* > 0\}$ (note that ℓ always exists because $a_{u,v'} > 0$ and we are considering the solution with the smallest number of zeros). Now let $0 < \delta < \min\{\delta_{j,k,u,v}^*, \delta_{j,\ell,u,v'}^*\}$. Reduce $\delta_{j,k,u,v}^*$ and $\delta_{j,\ell,u,v'}^*$ by δ and increase $\delta_{j,k,u,v'}^*$ and $\delta_{j,\ell,u,v}^*$ by δ. We now have another optimal solution with fewer zeros, a contradiction. ♢

It is obvious that the result of the proposition holds when the optimal solution is unique. However, when the optimal solution is not unique, then the solution obtained by Proposition 6 is not basic, and has more non-zero elements of $\delta_{j,k,u,v}^*$ than needed. This suggests that an optimal solution with more zeros but less nice structure may also be of interest. We next pursue this thought.

Note that if we are satisfied with either

$$\{k : \delta^*_{j,k,u,v'} > 0\} \subseteq \{k : \delta^*_{j,k,u,v} > 0\} \cap v' \tag{26}$$

or

$$\{k : \delta^*_{j,k,u,v} > 0\} \cap v' \subseteq \{k : \delta^*_{j,k,u,v'} > 0\}$$

for each j, u, v, and v', then we may be able to retain some zeros. Suppose neither statement is true of any optimal solution, and consider an optimal solution with the minimum number of j, u, v, v', and $k \in v'$ such that either $\delta^*_{j,k,u,v'} > 0$, $\delta^*_{j,k,u,v} = 0$ or $\delta^*_{j,k,u,v} > 0$, $\delta^*_{j,k,u,v'} = 0$. Since neither statement is true, there exist j, u, v, v', and $k, \ell \in v'$ such that

$$\begin{array}{ll} \delta^*_{j,k,u,v'} > 0, & \delta^*_{j,k,u,v} = 0, \\ \delta^*_{j,\ell,u,v} > 0, & \delta^*_{j,\ell,u,v'} = 0. \end{array}$$

Assume that $\delta^*_{j,k,u,v'} \neq \delta^*_{j,\ell,u,v'}$. If we decrease the entries in the left column by $\delta = \min\{\delta^*_{j,k,u,v'}, \delta^*_{j,\ell,u,v}\}$ and increase the entries in the right column by δ, we have an optimal solution with fewer values of j, u, v, v', and $k \in v'$ such that either $\delta^*_{j,k,u,v'} > 0$, $\delta^*_{j,k,u,v} = 0$ or $\delta^*_{j,k,u,v} > 0$, $\delta^*_{j,k,u,v'} = 0$, which is a contradiction.

Also, note that the added positive elements of $\delta^*_{j,k,u,v}$ (in both the argument in the previous paragraph and in the proof of Proposition 6) correspond to adding values for which $\delta^*_{j,k,u,v}$ is positive given $\delta^*_{j,k,u,v_0} > 0$ for at least one v_0. This means that server j is already capable of working at class k, so the additional positive value $\delta^*_{j,k,u,v}$ appears to be at very low cost.

Also, note that (25) does not hold for all optimal solutions, and there may not exist a basic feasible solution such that (25) holds (so the fact we need to increase the number of positive δ's is to some extent necessary). As an example, consider a tandem line of K stations with one server, only station one fails and state $v = 1$ corresponds to station one up and $v = 0$ corresponds to station one down. With three classes ($K = 3$) and one server with $\mu_{1,1} = \mu_{1,2} = \mu_{1,3} = 1$, and $a_{\{1\},\{1,2,3\}} = a_{\{1\},\{2,3\}} = 1/2$, we get $\lambda^* = 1/3$ with one possible solution $\tilde{\delta}^*_{1,1} = 1/3$, $\tilde{\delta}^*_{2,1} = 1/6$, $\tilde{\delta}^*_{3,1} = 0$, $\tilde{\delta}^*_{2,0} = 1/6$, and $\tilde{\delta}^*_{3,0} = 1/3$, where $\tilde{\delta}_{k,v}$ denotes an optimal allocation of the server to class k in state v. Clearly, (25) does not hold. It would if we set $\tilde{\delta}^*_{2,1} = \tilde{\delta}^*_{3,1} = 1/12$ and $\tilde{\delta}^*_{2,0} = \tilde{\delta}^*_{3,0} = 1/4$, but this solution does not have the maximum number of zeros.

5.3 Asymptotic results

Finally, we would like to show when one may remove the preemptive-resume assumption in step 5 of the server assignment algorithm in Section 3.2. This would allow a server to move to an arbitrary class in $V^{\pi}_{j,u',v'}$ when transitioning from (u, v) to (u', v') and remove the requirement of tracking $L_{j,u,v}(t)$ and $D_{j,u,v}(t)$ for each $(u, v) \in U \times V$ (see (8)). An

important special case of this is that if k is in both $V^{\pi}_{j,u,v}$ and $V^{\pi}_{j,u',v'}$, then the server does not have to move (see Proposition 6). In what follows, it is conceptually simple to include server switching times, however for ease of notation, we assume that they are zero.

Suppose that we have a sequence of networks, indexed by n. Only the failure and repair time distributions depend on n, the interarrival and service time distributions are fixed. Any quantity that depends on the nth network will have a superscript (n). Let $a^{(n)}_{u,v}$ be the proportion of time that the servers in u and classes in v are functioning for the nth network. We assume that $a_{u,v} = \lim_{n\to\infty} a^{(n)}_{u,v}$ exists for all $(u,v) \in U \times V$. We employ the server assignment algorithm to construct a server assignment policy π, using the proportions $a_{u,v}$ and desired capacity $\lambda < \lambda^*$ (in particular, π does not depend on n). We let the random variable $\tau^{(n)}_{u,v}$ represent the occupation time in state (u,v). (The variable $\tau^{(n)}_{u,v}$ may depend on other quantities in $X^{(n)}(t)$ (see (8)) but we suppress these dependencies and assume that the bound (27) below is uniform over these additional dependencies.) We then further assume that for any N and ε, there exists an n_0 such that for all $n \geq n_0$,

$$P\left\{\tau^{(n)}_{u,v} > N \max_{j\in u} \sum_{k\in V^{\pi}_{j,u,v}} d^{\pi}_{j,k,u,v}\right\} > 1-\varepsilon, \tag{27}$$

for all (u,v) such that $a_{u,v} = \lim_{n\to\infty} a^{(n)}_{u,v} > 0$. Condition (27) simply says that with high probability, each functioning server will have completed a large number of cycles in the states where (in the limit) a non-zero proportion of time is spent, while the servers in u and the classes in v are working. Without loss of generality, the assumption is that the condition is uniform in $(u,v) \in U \times V$. Let the queue-length vector at time t for network n be denoted by $Q^{(n)}(t)$ and let π^{NP} be a fixed policy that uses parameters $\{d^{\pi}_{j,k,u,v}\}$ chosen according to the allocation LP (see equations (4), (5), and (6)), but when switching from (u,v) to (u',v') may switch to any class according to some rule (so that π^{NP} is not preemptive-resume). We then have the following result:

Theorem 5 *Given an arrival process with rate $\lambda < \lambda^*$, for any policy of the form π^{NP}, there exists an index n^* such that the distribution of the queue length process $\{Q^{(n)}(t)\}$ converges to a steady-state distribution $\varphi^{(n)}$ as $t \to \infty$, for all $n \geq n^*$.*

Proof. The result follows if we can show (13) for π^{NP}. First, choose ε_1 such that for all u, v, and $k \in V^{\pi}_{j,u,v}$

$$\frac{d^{\pi}_{j,k,u,v}}{\sum_{m\in V^{\pi}_{j,u,v}} d^{\pi}_{j,m,u,v}} > \varepsilon_1 > 0.$$

Then choose N large enough such that for any k, u, v

$$\frac{(N-2)d^{\pi}_{j,k,u,v}}{N\sum_{m\in V^{\pi}_{j,u,v}} d^{\pi}_{j,m,u,v}} > \frac{d^{\pi}_{j,k,u,v}}{\sum_{m\in V^{\pi}_{j,u,v}} d^{\pi}_{j,m,u,v}} - \varepsilon_1. \tag{28}$$

The idea here is that if server j performs N cycles and there are a sufficient number of customers at class k, the first and last cycles may be partial and thus the server will complete at least $N-2$ complete busy periods of full length $d^{\pi}_{j,k,u,v}$ at any class k. The left-hand side would be a lower bound on the resulting proportion of time spent at class k. Choose $0 < \varepsilon_2 < 1$ and $\varepsilon_3 > 0$. Now, in each visit to the state (u, v), choose n_0 large enough so that for all $n \geq n_0$,

$$P\left\{\tau^{(n)}_{u,v} > N \max_{j\in u} \sum_{k\in V^{\pi}_{j,u,v}} d^{\pi}_{j,k,u,v}\right\} > 1-\varepsilon_2 \tag{29}$$

and

$$|a^{(n)}_{u,v} - a_{u,v}| < \varepsilon_3 \tag{30}$$

for all $(u, v) \in U \times V$. Let $n \geq n_0$. That there are at least N cycles in $\tau^{(n)}_{u,v}$ is a result of (29), and (28) then implies that if there are a large number of customers at class k, the proportion of time spent on class k by server j during those N cycles with the servers in u and the classes in v functioning is at least as large as the right-hand side of (28). Let $B^{(n)}_{u,v}(t)$ be the time that the nth network has been in state (u, v) up to time t. First, as in (10)

$$\frac{d}{dt}\bar{B}^{(n)}_{u,v}(t) = a^{(n)}_{u,v}. \tag{31}$$

For each n, taking the fluid limit of $T^{(n)}_{j,k,u,v}(t)$ and following the derivation of (11),

$$1 \geq \frac{d\bar{T}^{(n)}_{j,k,u,v}(t)}{d\bar{B}^{(n)}_{u,v}(t)} \geq \left(\frac{(N-2)d^{\pi}_{j,k,u,v}}{N\sum_{m\in V^{\pi}_{j,u,v}} d^{\pi}_{j,m,u,v}}\right) P\left\{\tau^{(n)}_{u,v} > N \sum_{m\in V^{\pi}_{j,u,v}} d^{\pi}_{j,m,u,v}\right\} \tag{32}$$

$$> \left(\frac{d^{\pi}_{j,k,u,v}}{\sum_{m\in V^{\pi}_{j,u,v}} d^{\pi}_{j,m,u,v}} - \varepsilon_1\right)(1-\varepsilon_2), \tag{33}$$

whenever $\bar{Q}^{(n)}_k(t) > 0$ and the derivative exists. The right hand side of (32) has the interpretation that on the event that the occupation time in (u, v) is sufficiently long, the proportion of time spent by server j at class k while in (u, v) (and hence the corresponding derivative of the busy time) can be bounded below by the left hand side of (28). Using the chain rule on (33) and (31), then in turn applying (30), we have

$$a_{u,v} + \varepsilon_3 > \frac{d}{dt}\bar{T}^{(n)}_{j,k,u,v}(t) > (a_{u,v} - \varepsilon_3)\left(\frac{d^{\pi}_{j,k,u,v}}{\sum_{m\in V^{\pi}_{j,u,v}} d^{\pi}_{j,m,u,v}} - \varepsilon_1\right)(1-\varepsilon_2),$$

whenever $\bar{Q}_k^{(n)}(t) > 0$ and the derivative exists. As ε_1, ε_2, and ε_3 are arbitrary, (13) holds and the remainder of the proof is exactly the same as the proof of part (i) of Theorem 1. $\diamondsuit$

There are two important special cases addressed by Theorem 5. If both the failures and repairs are occurring on a much slower time scale than the interarrival and service times, then Theorem 5 justifies the use of nonpreemptive-resume policies that asymptotically approach the maximal capacity as the difference between the two time scales increases. Also, if the failures are happening on a much slower time scale than the remainder of the system dynamics (including repairs), then we may simplify the policy even further. To be precise $a_{u,v}^{(n)} \to 0$ for all (u, v) except $(\{1, \ldots, M\}, \{1, \ldots, K\})$ (i.e., the state where all servers and classes are working), and consequently Theorem 5 implies that using the optimal allocation for a system without failures approaches the maximal capacity.

6 Numerical Example

We consider System 1 of Andradóttir, Ayhan, and Down [7] with no switching times. This is a tandem network with $K = 10$ classes and $M = 3$ servers, see Figure 1. The matrix $\mathcal{M}$ below has (j, k) entry $\mu_{j,k}$.

$$\mathcal{M} = \begin{bmatrix} 1 & 2 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 2 & 1 & 1 \end{bmatrix}$$

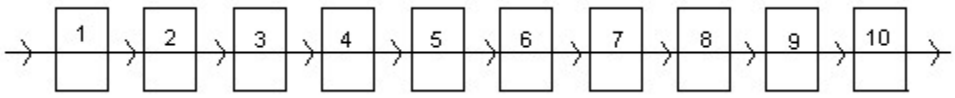

Figure 1: Queueing Network of System 1.

We examine various failure scenarios. Note that from [7], the maximal capacity in this system without failures is (to four decimal places) $\lambda^* = 0.3529$. A second network was considered in [7]. We also performed numerical work for this system, but as the conclusions were similar, in the interest of space, the results have not been included here.

Server failures. First, we examine a system where the servers fail independently, with the probability that each server is failed being 0.1. In this case, the maximal capacity is 0.3176, or 90 percent of that for the system without failures, using Corollary 2.

Next, suppose that the servers still fail independently, but server 2 is failed 25 percent of the time, the remaining two servers 10 percent of the time. In this case, the maximal capacity is 0.3000.

In light of Corollary 1, these results should come as no surprise and indicate that in the presence of server failures only, flexibility will not be particularly useful in compensating for failure effects.

Class failures. The first system is one where classes 5, 7, and 10 fail independently and the probability that each is failed is 0.1. In this case, the maximal capacity is 0.3529, equal to that of the same system without failures. It can be verified that in this instance, condition (21) holds.

If we now increase the probability that class 5 is failed to 0.25 and keep everything else the same, the maximal capacity is reduced to 0.3522. Note that this means that condition (21) no longer holds. However, the decrease from the maximal capacity in the network without failures is less than 0.2 percent in this case, even with the significant increase in failure probability.

For these two scenarios, we also used simulation to estimate the maximal capacity for a policy designed for a system without failures (using the choice of parameters given in the proof of Proposition 1 with $\beta = 1$). This was done by simulating a system with an unlimited supply of customers at class one. The service requirement distributions were all taken to be exponential with mean one (see Section 2.2). The resulting 95 percent confidence intervals for the maximal capacity were 0.3493 ± 0.0029 for the first scenario and 0.3240 ± 0.0044 for the second, demonstrating that taking the failure statistics into account when deciding on a server assignment policy is crucial. (In both cases, we ran a warm-up period of length 100,000 followed by a run of length 3,000,000, which was divided into 30 batches.)

Server and class failures. We looked at a system where there were both server and class failures and these were correlated. This is a combination of the previous two failure scenarios, to be precise, class 5 and server 1 fail and are repaired simultaneously, with both being failed with probability 0.1, class 7 and server 2 act in the same manner, as do class 10 and server 3. In this case, the maximal capacity is 0.3176 which is the same as the corresponding case where there are server failures only, i.e., the server flexibility is able to completely compensate for the class component of the failures. This amplifies the insight that flexibility is useful in compensating for class failures, but not server failures.

Structural properties. Returning to the case where the failure probabilities for classes

5, 7, and 10 are all 0.1 and the servers do not fail, we define $\mathcal{K} = \{1, 2, 3, 4, 6, 8, 9\}$ to be the set of classes that are always working. Upon solving the allocation LP, we then have the following allocations $\delta^*_{1,k,v}$ for server 1 to class k while in state v (all unspecified $\delta^*_{1,k,v}$ correspond to classes which server 1 is not capable of serving and thus are zero):

$$\begin{aligned}
v = \{5,7,10\} \cup \mathcal{K} : \{\delta^*_{1,1,v}, \delta^*_{1,2,v}, \delta^*_{1,3,v}, \delta^*_{1,4,v}, \delta^*_{1,5,v}\} &= \{0.2176, 0.1763, 0.2176, 0.1175, 0\},\\
v = \{5,10\} \cup \mathcal{K} : \{\delta^*_{1,1,v}, \delta^*_{1,2,v}, \delta^*_{1,3,v}, \delta^*_{1,4,v}, \delta^*_{1,5,v}\} &= \{0.0405, 0, 0.0405, 0, 0\},\\
v = \{7,10\} \cup \mathcal{K} : \{\delta^*_{1,1,v}, \delta^*_{1,2,v}, \delta^*_{1,3,v}, \delta^*_{1,4,v}, \delta^*_{1,5,v}\} &= \{0.0405, 0, 0.0405, 0, 0\},\\
v = \{5,7\} \cup \mathcal{K} : \{\delta^*_{1,1,v}, \delta^*_{1,2,v}, \delta^*_{1,3,v}, \delta^*_{1,4,v}, \delta^*_{1,5,v}\} &= \{0.0405, 0, 0.0405, 0, 0\},\\
v = \{10\} \cup \mathcal{K} : \{\delta^*_{1,1,v}, \delta^*_{1,2,v}, \delta^*_{1,3,v}, \delta^*_{1,4,v}, \delta^*_{1,5,v}\} &= \{0.0045, 0, 0.0045, 0, 0\},\\
v = \{5\} \cup \mathcal{K} : \{\delta^*_{1,1,v}, \delta^*_{1,2,v}, \delta^*_{1,3,v}, \delta^*_{1,4,v}, \delta^*_{1,5,v}\} &= \{0.0045, 0, 0.0045, 0, 0\},\\
v = \{7\} \cup \mathcal{K} : \{\delta^*_{1,1,v}, \delta^*_{1,2,v}, \delta^*_{1,3,v}, \delta^*_{1,4,v}, \delta^*_{1,5,v}\} &= \{0.0045, 0, 0.0045, 0, 0\},\\
v = \mathcal{K} : \{\delta^*_{1,1,v}, \delta^*_{1,2,v}, \delta^*_{1,3,v}, \delta^*_{1,4,v}, \delta^*_{1,5,v}\} &= \{0.0005, 0, 0.0005, 0, 0\}.
\end{aligned}$$

This conforms with (26). Similar patterns emerge for the other two servers, as a result we have chosen not to include them. It is interesting to note that classes 2 and 4 are only required to be served by server 1 when all of the classes are functioning.

Asymptotics. We now explore how quickly the asymptotics developed in Section 5.3 become useful. Our results also give an indication for how much capacity is lost by not moving a server (whenever possible) when a class fails. To this end, we again simulated the network with failures at classes 5, 7, and 10, no server failures, unlimited supply of customers in front of the first station, and with all underlying distributions assumed to be exponential with mean one. We looked at three scenarios: (i) the failure and repair rates of classes 5, 7, and 10 were 0.01 and 0.1, respectively; (ii) the same failure and repair rates were 0.001 and 0.01, respectively and (iii) the same failure and repair rates were 0.0001 and 0.001, respectively. In each case, the server assignment policy was to calculate the timed generalized round robin policy π_v to be used while in state v by using the server assignment algorithm in Section 3.2 with $\lambda = 0.95\lambda^*$ (the generalized round robin policy for each v was calculated using the choice of parameters given in the proof of Proposition 1 with $\beta = 1$), but when the system moves from state v to state v', the servers stay at the current class if allowed by $\pi_{v'}$, otherwise they move to the class in $V^{\pi}_{j,v'}$ with the smallest number that is larger than the current class that j is at in (u, v). If there is no such class, then server j should go to the first class in $V^{\pi}_{j,v'}$. For scenarios (i) and (ii), we ran a warm-up period of length 100,000 followed by a run of length 3,000,000, which was divided into 30 batches. For scenario (iii), the warm-up period and run length were increased to 1,000,000 and 30,000,000,

respectively. The 95 percent confidence intervals for the maximal capacity were

$$\begin{array}{ll} \text{(i)} & 0.3401 \pm 0.0047, \\ \text{(ii)} & 0.3484 \pm 0.0143, \\ \text{(iii)} & 0.3505 \pm 0.0137. \end{array}$$

Compared to the actual capacity of 0.3529, scenarios (ii) and (iii) have the maximal capacity within their confidence intervals, and in general we see a tendency to the maximal value as predicted by Theorem 5.

7 Concluding Remarks

We have provided a simple means to compute the maximal capacity for a queueing network with flexible servers, i.i.d. interarrival and service times, and failure/repair dynamics which can be described by a Markov process. An algorithm has been given to explicitly construct timed generalized round robin policies that achieve any given capacity less than the maximal capacity. We have also provided insights into when a given flexibility structure can compensate for failures and discussed how implementation of our policies may be simplified.

Our research yields the following managerial insights:

1. The maximal capacity of a system with failures and repairs can be easily calculated and dynamic server assignment policies that have capacity arbitrarily close to the maximal capacity are simple to construct.

2. The maximal capacity depends on mean values only, second order effects (including correlations) have no impact. In particular, the maximal capacity only depends on the failures through the proportions of time that particular sets of servers and classes are working.

3. Server flexibility is a useful way to (completely) compensate for class failures but has little impact with respect to server failures.

4. The set of classes visited in the constructed policies is relatively insensitive to the failure state of the system and if there are server failures only, completely insensitive.

5. If the failure/repair dynamics are on a much slower time scale than the server movement, then little capacity is lost if a server attempts to avoid movement when a failure or repair occurs.

Acknowledgments. The research of the first author was supported by the National Science Foundation under Grants DMI-0000135, DMI-0217860, and DMI-0400260. The research

of the second author was supported by the National Science Foundation under Grants DMI-9908161 and DMI-9984352. The research of the third author was supported by the National Science Foundation under Grant DMI-0000135 and the Natural Sciences and Engineering Research Council of Canada.

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*, 31:1095-1117, 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] R. Akella and P.R. Kumar. Optimal control of production rate in a failure prone manufacturing system. *IEEE Transactions on Automatic Control*, 31:116-126, 1986.

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

[6] 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.

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

[8] N. Bambos and G. Michailidis. Queueing and scheduling in random environments. *Advances in Applied Probability*, 36:293-317, 2004.

[9] J.J. Bartholdi and D.D. Eisenstein. A production line that balances itself. *Operations Research*, 44:21-34, 1996.

[10] J.J. Bartholdi, D.D. Eisenstein, and R.D. Foley. Performance of bucket brigades when work is stochastic. *Operations Research*, 49:710-719, 2001.

[11] 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.

[12] 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.

[13] D.P. Bischak. Performance of a manufacturing module with moving workers. *IIE Transactions*, 28:723-733, 1996.

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

[15] J.A. Buzacott and J.G. Shanthikumar. *Stochastic Models of Manufacturing Systems.* Prentice Hall, 1992.

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

[17] J.G. Dai. *Stability of Fluid and Stochastic Processing Networks*. Publication No. 9, Centre for Mathematical Physics and Stochastics (`http://www.maphysto.dk/`), 1999.

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

[19] J.G. Dai and S.P. Meyn. Stability and convergence of moments for multiclass queueing networks via fluid models. *IEEE Transactions on Automatic Control*, 40:1889-1904, 1995.

[20] Y. Dallery and S.B. Gershwin. Manufacturing flow line systems: A review of models and analytical results. *Queueing Systems*, 12:13-94, 1992.

[21] M.H.A. Davis. Piecewise deterministic Markov processes: A general class of diffusion stochastic models. *Journal of Royal Statistic Society, Series B*, 46:353-388, 1984.

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

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

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

[25] J.M. Harrison and J.A. van Mieghem. Dynamic control of Brownian networks: State space collapse and equivalent workload formulations. *Annals of Applied Probability*, 7:747-771, 1997.

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

[27] W.J. Hopp and M.P. van Oyen. Agile workforce evaluation: A framework for cross-training and coordination. *IIE Transactions*, 36:919-940, 2004.

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

[29] E.L. Lawler. *Combinatorial Optimization: Networks and Matroids.* Holt, Rinehart, and Winston, New York, NY, 1976.

[30] C.N. Laws. Resource pooling in queueing networks with dynamic routing. *Advances in Applied Probability*, 24:699-726, 1992.

[31] A. Mandelbaum and A.L. Stolyar. Scheduling flexible servers with convex delay costs: heavy traffic optimality of the generalized $c\mu$ rule. *Operations Research*, 52:836-855, 2004.

[32] S.P. Meyn and D. Down. Stability of generalized Jackson networks. *Annals of Applied Probability*, 9:124-148, 1994.

[33] J. Ostalaza, J. McClain and J. Thomas. The use of dynamic (state-dependent) assembly-line balancing to improve throughput. *Journal of Manufacturing and Operations Management*, 3:105-133, 1990.

[34] J.R. Perkins and R. Srikant. Scheduling multiple part-types in an unreliable single machine manufacturing system. *IEEE Transactions on Automatic Control*, 42:364-377, 1997.

[35] A.N. Rybko and A.L. Stolyar. Ergodicity of stochastic processes describing the operation of open queueing networks. *Problems of Information Transmission*, 28:199-220, 1992.

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

[37] K. Sigman. The stability of open queueing networks. *Stochastic Processes and their Applications*, 35:11-25, 1990.

[38] M.S. Squillante, C.H. Xia, D.D. Yao, and L. Zhang. Threshold-based priority policies for parallel-server systems with affinity scheduling. *Proceedings of the 2001 American Control Conference*, 2992-2999, 2001.

[39] L. Tassiulas and P.B. Bhattacharya. Allocation of independent 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] 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.

[42] E. Zavadlav, J.O. McClain, and L.J. Thomas. Self-buffering, self-balancing, self-flushing production lines. *Management Science*, 42:1151-1164, 1996.