Stability, Design and Control of Fork-Join Systems with Redundancy and Heterogeneous Servers

Chutong Gao
Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL 60208, chutong@u.northwestern.edu

Seyed Iravani
Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL 60208, s-iravani@northwestern.edu

Ohad Perry
Department of Operations Research and Engineering Management, Southern Methodist University, Dallas, TX 75205, operry@smu.edu

We consider the stability problem of fork-join systems with redundancy where servers are heterogeneous. In these systems, an arriving job consists of multiple tasks, each of which is sent to a separate queue, and a job is completed once some, but not all, of its tasks are processed by the servers. In the case where two tasks for each job must be processed, we establish a necessary and sufficient condition for the system to be stable under any static capacity-allocation policy, namely, when the service rate of each server is fixed. We then utilize the stability analysis to study how to allocate the total service capacity to maximize the stability region. We show that a static policy achieves the maximal stability region if, and only if, the capacity allocated to each server does not exceed one half of the total capacity. For dynamic policies, under which the total service capacity can be reallocated continuously among the servers, we show that the maximal stability region is the same as under static policies, but it can be achieved under a milder condition. Specifically, the maximal stability region is achieved when the capacity allocated to the server with the shortest queue does not exceed one half of the total service capacity at any time. Our results provide insights into the stability problem of more general fork-join systems with redundancy, which are supported via a simulation study.

Key words: Fork-Join Systems; Redundancy; Stability; Queueing Theory; Coupling Arguments
History: Manuscript version – May, 2025

1. Introduction

We consider fork-join systems with redundancy (abbreviated by FJR) and statistically heterogeneous servers. In these systems, a "fork" operation splits each arriving job into $n > 2$ tasks, each of which is sent to a *queue-buffer* in one of n stations. Tasks are processed on a first-come-first-served (FCFS) basis, and then move to the *join-buffer* associated with its queue-buffer; if there are additional $k-1$ $(1 < k < n)$ tasks from the same job waiting in other join-buffers, then a "join" operation is performed on those k tasks, and the job departs the system. Otherwise (if there are less than $k-1$ additional processed tasks from the same job), the newly processed task waits in its join-buffer until the kth task from the same job is processed for the join operation to occur. Upon a job's departure, its remaining $n-k$ unprocessed tasks are considered *redundant* and are immediately removed from the system without being processed. We denote an FJR system with

n stations (each station consists of a queue-buffer, a server and a join-buffer) in which $k \leq n$ tasks must be processed from each job by (n, k). A $(3, 2)$ system is depicted in Figure 1 below.

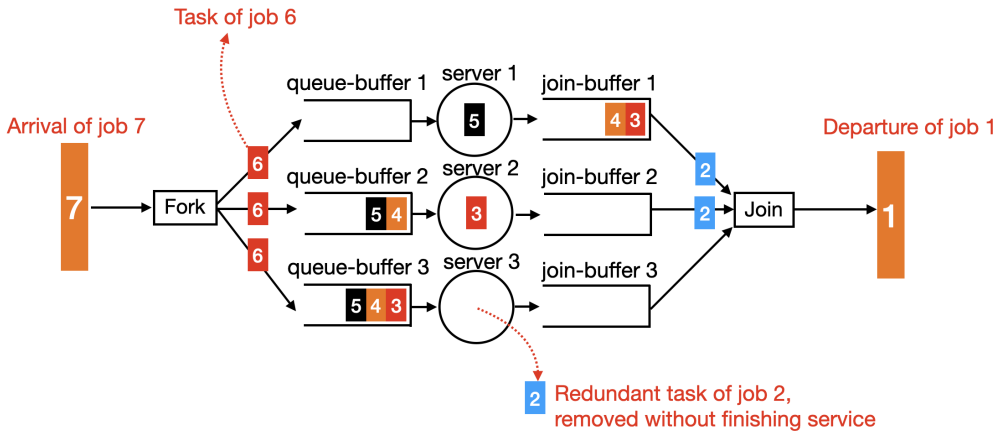

Figure 1 **Illustration of the $(3, 2)$ system. In the figure, the second task of job 2 (from queue-buffer 2) has just entered the join-buffer, so the join operation immediately collects the two tasks of job 2 (in join-buffer 1 and join-buffer 2) to complete job 2. At the same time, the redundant task of job 2 in queue-buffer 3 is removed.**

Such FJR systems are ubiquitous in practice and have been employed as effective models in numerous applications, such as cloud computing (Rizk et al. (2016), Wang et al. (2019), Harchol-Balter (2021)), healthcare systems (Gardner et al. (2016, 2017), Özkan and Ward (2019), Özkan (2022), Carmeli et al. (2023)), parallelized machine-learning algorithms (Chu et al. (2006), Lee et al. (2017a), Li et al. (2020), Hu et al. (2021), Ouyang et al. (2022), Anthropic (2024)), and manufacturing systems (Schol et al. (2022), Meijer et al. (2024)). We note that the term "task" has different meanings in different settings; in particular, in some settings, each job is split into parts, each of which is a (different) task, whereas in other settings, each task is an identical copy of the corresponding job. The former type of tasks is relevant in, e.g., distributed cloud storage systems, in which each file (job) stored in the cloud is encoded into n blocks (tasks), each encoding partial information of the original file. Retrieving k blocks, $k < n$, is sufficient to reconstruct the original file; see Joshi et al. (2012, 2014, 2015, 2017), Lee et al. (2017b). The latter type of tasks is employed in, e.g., large language models (LLMs) in which copies (tasks) of the same prompt (job) are sent to independent LLMs (servers), and some of the copies are collected so as to gain diverse perspectives for higher-confidence output Anthropic (2024)).

However, despite being extensively employed and studied, even basic properties of FJR systems are still unknown. Indeed, even the stability region (the set of system primitives under which the number-in-system process is positive recurrent) in the homogeneous Markovian case (i.e., when all

servers have the same exponential service-time distribution) is unknown. Of course, the characterization of the stability region is even more complex when servers are heterogeneous, as in this paper.

Server and Task Heterogeneity. FJR systems, for example in the contexts of cloud storage and computing systems, constantly add to their service capacity, either via additional service stations, or by adding capacity to existing stations, or both (see Gardner and Stephens (2019)), leading to substantial server heterogeneity. Moreover, in contrast to most of the existing literature, which assumes that task sizes are independent and identically distributed (i.i.d.) random variables, a job's tasks in different queue-buffers are often heterogeneous in size, especially when each constitutes a part of a job (see, e.g., Lee et al. (2017a)). As we show below, server heterogeneity allows us to cover the case of task heterogeneity as well.

Our Stability Analysis. Clearly, the difficulty in analyzing (n, k) systems stems from the fact that a representation of the number-in-system process is high-dimensional even when n and k are relatively small. Thus, our theoretical analysis focuses on the special case with $k = 2$. We then draw on our rigorous analysis of the $(n, 2)$ system to gain insights into the general cases with $k > 2$. In particular, we characterize a necessary and sufficient stability condition for $(n, 2)$ systems with any set of service capacities, and provide a lower bound for the maximal stability region in the general (n, k) case, when $k > 2$. We conjecture that the lower bound is tight, as in the special case $k = 2$, and support that conjecture with simulation experiments.

1.1. Maximal Design and Control

The characterization of the stability region of $(n, 2)$ systems enables us to answer the question of how to allocate a given service capacity across all stations so as to achieve the maximal stability region. We consider two different settings: (i) *static allocation* of the service capacity, namely, when the total service capacity is distributed among the servers at the outset, and that allocation remains fixed throughout; (ii) *dynamic allocation*, in which the total (and fixed) service capacity can be continuously reallocated between the different servers. Note that the problem of how to choose a static allocation policy is a design problem, whereas the problem of choosing a dynamic allocation policy is a control problem. In either case, our goal is to determine conditions ensuring that the policy is maximal, in the sense that the maximal stability region is achieved when that policy is employed.

Maximal Design. As was mentioned above, FJR systems routinely add service capacity, giving rise to the question of how to best allocate that newly added capacity. A first-order consideration when answering this question is to ensure that the new system's design, in terms of how the new total service capacity is distributed among the different stations, achieves the maximal stability

region. We treat this problem as a static-control problem, since one can think of the fixed allocation of the service as a special case of the dynamic control problem in which the service capacity is continuously reallocated at different times. For this subclass of static policies, we prove that an $(n, 2)$ system achieves the maximal stability region if and only if the capacity allocated to any server does not exceed one half of the total service capacity.

Maximal Control. In some applications, a given service capacity can be reallocated to different service stations (cf. Pedarsani et al. (2014a, 2017), Özkan and Ward (2019)), and the controller must therefore decide how to best allocate that capacity at each decision epoch. For example, workers at one station may be reassigned to help workers at other stations with their task (Özkan and Ward (2019), Özkan (2022)), or a fixed power allotment may be continuously redistributed between the different servers, thus changing their individual service rates, while keeping the total service rate fixed (Gandhi et al. (2009)). In such cases, the total service capacity is a given, and one must come up with a control that determines how that service capacity should be distributed among the servers at any given time. In this setting, we prove that the maximal stability region is achieved under any dynamic policy that assigns no more than one half of the total capacity to the server with the shortest queue.

1.2. Contributions

In summary, our contributions are as follows.

Stability. We characterize the stability regions and the maximal stability region of $(n, 2)$ systems with heterogeneous servers, providing, in turn, the stability region of the special case of homogeneous servers. (We note that even the latter simpler case is an open problem.)

Maximal Design. We solve the maximal design problem for $(n, 2)$ systems, namely, the problem of how to allocate the capacity between the different servers so as to maximize the system's stability region. In particular, we establish a necessary and sufficient criterion for static policies to achieve the maximal stability region.

Maximal Control. We prove that the maximal stability region for the class of dynamic policies in $(n, 2)$ systems is the same as the stability region for the subclass of static policies. Hence, the added flexibility does not enlarge the stability region compared to static policies. Nevertheless, we show that dynamic policies can achieve the maximal stability region under a less restrictive capacity-allocation criterion than that of static policies.

General Lower Bounds for the Stability Region. Building on our analyses for $(n, 2)$ systems, we establish a lower bound for the maximal stability region of (n, k) systems with $k > 2$ under both static and dynamic policies, and conduct simulation experiments to support our conjecture that that lower bound is tight.

Analytical Contribution. Analyzing (Markovian) FJR systems is difficult due to the high dimensionality of the underlying state space, and the existence of "phantom" tasks, namely, the redundant tasks which are removed from the system before being processed. We overcome these issues by establishing an equivalent tandem queue representation for $(n,2)$ systems, which substantially reduces the dimensionality of the state space and bypasses the need to keep track of the phantom tasks. It is significant that the equivalent tandem queue representation can be used to analyze more general (n,k) systems with $k>2$, although the number of stations in tandem increases with k. Further, we characterize the stability regions via sample-path stochastic-order bounds using a novel coupling argument with *filter inequalities*. Specifically, those latter inequalities, which we impose on the coupled processes, "filter out" events that cause the order between the coupled processes to be violated; see §6.4 below. This coupling technique can be useful in many other settings, in which the order between coupled processes holds w.p.1 only for certain initial conditions that can be identified via a filter inequality.

1.3. Notation

We let $\mathbb{Z}$ and $\mathbb{R}$ denote the set of integers and real numbers, respectively, with $\mathbb{Z}_+ := \mathbb{Z} \cap [0,\infty)$ and $\mathbb{R}_+ := [0,\infty)$. For $a,b \in \mathbb{Z}_+$, $[\![a,b]\!] := \mathbb{Z}_+ \cap [a,b]$ denotes the set of all integers in the interval $[a,b]$. We let $x \wedge y := \min\{x,y\}$. For a function $f:\mathbb{R}^n \to \mathbb{R}$, we denote by $f(t-)$ the left limit of f at t. We define the 0th norm of a vector $\boldsymbol{x} = (x_1,\cdots,x_n) \in \mathbb{R}^n$, and denote it by $\|\boldsymbol{x}\|_0$, as the number of nonzero components of x, i.e., $\|\boldsymbol{x}\|_0 = \sum_{i=1}^n \mathbb{1}_{\{x_i \neq 0\}}$, where $\mathbb{1}_A$ is the indicator function of set A (that is equal to 1 on A and to 0 otherwise). We write $X \leq_{st} Y$ if for two $\mathbb{R}_+$-valued stochastic processes X and Y, X is less than or equal to process Y in sample-path stochastic order, i.e., there exist two stochastic processes $\mathcal{X}$ and $\mathcal{Y}$, defined on the same probability space, such that (i) $\mathcal{X}(t) \leq \mathcal{Y}(t)$ w.p.1 for all $t \geq 0$, and (ii) $X \overset{\mathrm{d}}{=} \mathcal{X}, Y \overset{\mathrm{d}}{=} \mathcal{Y}$, where $\overset{\mathrm{d}}{=}$ denotes equality in distribution.

1.4. Organization

The remainder of the paper is organized as follows. After a literature review in §2, we introduce the model in detail in §3. In §4 we present an equivalent tandem queue representation of $(n,2)$ systems. We employ the equivalent tandem queue in §5 and in §6 to study the stability of $(n,2)$ systems under static and under dynamic policies, respectively. We examine the stability of (n,k) systems with $k>2$ in §7 and summarize in §8.

2. Literature Review

FJR Systems. We begin with a review of the literature on (n,d,k) systems, where $n \geq d \geq k$. In these systems, each job consists of d tasks, which are assigned to d out of n parallel stations according to a dispatching policy upon the job's arrival. A job is considered complete once any k

of its d tasks have been processed; the remaining $d-k$ tasks are then discarded. When $d=k=1$, the system reduces to a parallel-server system, where each job is routed to one of the n parallel stations and processed by a dedicated server. Even in this simpler setting, stability is not always straightforward, as demonstrated in Moyal and Perry (2022). The FJR systems we study here are the (n,n,k) special case of the (n,d,k) systems, which we denote by (n,k).

These (n,k) systems have *full redundancy*, because a task from each job is sent to all n stations. Full redundancy is prevalent in applications due to its increased efficiency in reducing sojourn times; see Joshi et al. (2014), Shah et al. (2015), Joshi et al. (2015), Shah et al. (2015), Joshi et al. (2017), Gardner et al. (2017), Lee et al. (2017a,b), Gardner et al. (2019). In particular, Shah et al. (2015) prove that when servers are statistically homogeneous with service times that are exponentially distributed, or have a heavier tail than exponential, the mean steady-state sojourn time in an (n,n,k) is the smallest among all (n,d,k) systems with $d<n$, so that full redundancy minimizes the mean steady-state sojourn time. Similarly, Joshi et al. (2014) employed an (n,d,k) system to model distributed cloud storage systems, and showed that download latency is substantially reduced with the full redundancy design; see also Joshi et al. (2015), Lee et al. (2017b).

Gardner et al. (2017) perform exact analysis for the special case of the $(n,d,1)$ system with Poisson job arrivals under the uniform-at-random task dispatching policy and homogeneous exponential servers, and derive a closed-form expression for the mean sojourn time and the stability region of that system. The analytic expression suggests that the mean sojourn time is strictly monotonically decreasing with respect to d, and thus full redundancy minimizes the mean sojourn time. A more detailed relation between the distribution of service times and the benefits of redundancy is established by Joshi et al. (2017) for the $(n,d,1)$ system, who prove that when the service time is log-convex, full redundancy achieves the mean sojourn time optima.

Flatto and Hahn (1984) and Nelson and Tantawi (1988) characterize the stability region of the $(2,2,2)$ system and derive the generating function of its stationary queue-length distribution. For (n,n,n) with $n>2$, only bounds on the mean sojourn time were established; see Baccelli et al. (1989), Varki (1999), Ko and Serfozo (2008), Thomasian (2014), and the survey paper Sethuraman (2022).

It is significant that none of the papers cited above established a stability condition for the (n,n,k) system with full redundancy, not even when the servers are homogeneous, except for the case of $k=1$, which was studied in Gardner et al. (2017), Anton et al. (2021).

Flexible Capacity Allocation. In this paper, we consider a dedicated single server for each queue-buffer, and a general class of dynamic capacity-allocation policies, which determine the service capacity at each station at any given time. Thus, our paper relates to the work on *flexible and collaborative servers*, which studies queueing systems in which servers can be assigned to different

queues, and collaborate on processing a single task. The assumption of flexible and collaborative servers has been considered in various settings, including parallel queueing systems (Down and Lewis (2006), Bassamboo et al. (2012)), tandem queues (Andradóttir and Ayhan (2005), Down and Lewis (2006)), and queueing networks (Andradóttir et al. (2003), Pedarsani et al. (2014a,b), Dai and Harrison (2020)).

However, there are only a few papers that study flexible capacity allocation in fork-join networks. Pedarsani et al. (2014b) consider the problem of allocating flexible and collaborative servers for a queueing network consisting of multiple nested (n, n, n) queues, and propose a linear program to numerically calculate a policy that maximizes throughput. Marin and Rossi (2016) study an (n, n, n) saturated system under flexible and collaborative server allocation, and prove that that saturated system is unstable. Özkan and Ward (2019) consider an (n, d, d) system in which each queue-buffer has a dedicated server, except for a subset of queue-buffers that have a shared single flexible server. They solve the scheduling problem for the single flexible server using an approximating Brownian control problem, and prove asymptotic optimality of their policy under the criterion of stochastic-order optimality. Özkan (2022) extends the result to the case of more than one flexible server. All these studies assume no redundancy.

3. The Model

In this section, we introduce the $(n, 2)$ system in detail. (We consider the more general (n, k) system, with $k > 2$, in §7.) We henceforth refer to the $(n, 2)$ system (with any $n > 2$) as FJR_2. As was mentioned above, the FJR_2 system has n parallel stations, each consisting of a queue-buffer, in which tasks are queued to be processed, a server, and a join-buffer where processed tasks wait to be joined. Jobs, each consisting of n tasks, arrive at the system according to a Poisson process with rate λ. We assume that the task sizes of each job are independent and identically distributed (i.i.d.), each having an exponentially distributed size with mean 1, and a server processes a task at a rate that equals its *capacity*; hence, the time it takes a server with capacity $\mu > 0$ to process a task is exponentially distributed with mean $1/\mu$. (Below, we show that our model naturally incorporates the case in which tasks are not identically distributed; see Example 2.)

Upon arrival of a job, one of its n tasks is sent to each queue-buffer, so that each station receives exactly one task from each job, where the task is served according to FCFS. After a task is processed by the server, it moves to the join-buffer of the same station, where it is either immediately joined with another task from the same job that is waiting in a different join-buffer, if such a task exists, or else waits for a second task from the same job to be processed. When two tasks are joined, the corresponding job is considered complete, departs the system, and the remaining $n-2$ tasks which did not complete their processing are immediately removed from the system.

For $i \in [\![1,n]\!]$, let $F_i(t)$ be the number of tasks in queue-buffer i at time t, including the job in service, and let $J_i(t)$ be the number of tasks in join-buffer i at time t. Let $\boldsymbol{F}(t) = (F_1(t), F_2(t), \cdots F_n(t))$, $\boldsymbol{J}(t) = (J_1(t), J_2(t), \cdots J_n(t))$, and let $N(t)$ denote the total number of jobs in the system at time t. We remove the time argument from the notation when it is clear that we refer to the corresponding stochastic processes, e.g., $\boldsymbol{F} := \{\boldsymbol{F}(t) : t \geq 0\}$.

3.1. Capacity-Allocation Policies

We assume that the system has a fixed total capacity $\mu^* > 0$, which is distributed among the n servers. A *dynamic* capacity-allocation policy π specifies the proportion $\gamma_i^\pi(t)$ of μ^* that is allocated to server i at time t; in particular, the instantaneous service rate (or "capacity") of server i at time t is $\gamma_i^\pi(t)\mu^*$. We refer to $\boldsymbol{\gamma}^\pi := \{\boldsymbol{\gamma}^\pi(t) : t \geq 0\}$ as the *capacity-allocation process*, where $\boldsymbol{\gamma}^\pi(t) := (\gamma_1^\pi(t), \gamma_2^\pi(t), \cdots, \gamma_n^\pi(t))$ is an n-dimensional probability vector, taking values in the set

$$\Delta^n := \left\{ \boldsymbol{d} \in \mathbb{R}_+^n : \sum_{i=1}^n d_i = 1 \right\}.$$

We assume that π is non-anticipative, namely, that decisions on how to allocate service capacity do not depend on future information, and rely solely on the state of the system at or before the decision epochs. Further, we assume that decisions are made at event times—when a new job arrives to the system, or upon a service completion of a task. We denote by Π the class of all such non-anticipative capacity-allocation policies, and attach superscript π to processes, e.g., $\boldsymbol{F}^\pi, \boldsymbol{J}^\pi$, when we want to emphasize the dependence on the specific policy π. We note that Π is a large family of policies, including all Markovian policies (under which the system evolves as a CTMC), and the commonly considered setting in which the service times of all tasks are i.i.d.

Example 1 (static allocation). For a constant vector $\boldsymbol{d} \in \Delta^n$, let $\pi_{\boldsymbol{d}}$ be the policy that allocates the service capacity according to $\boldsymbol{\gamma}^{\pi_{\boldsymbol{d}}}(t) \equiv \boldsymbol{d}$ for all $t \geq 0$. Under $\pi_{\boldsymbol{d}}$, the system is an FJR_2 queue with heterogeneous servers in which the tasks in station i are processed at rate $\mu_i := d_i\mu^*$. In the special case of $d_i = 1/n$ for all $i \in [\![1,n]\!]$, the system has homogeneous servers, each processing tasks (which are independent and whose size is exponentially distributed with mean 1) at a fixed rate $\mu = \mu^*/n$. In particular, our framework covers the widely considered setting of i.i.d. processing times of i.i.d. tasks.

Example 2 (non-homogeneous tasks and servers). An FJR_2 system in which task sizes in queue-buffer i are exponentially distributed with rate ν_i, and server i processes work at a fixed rate μ_i, is equivalent to our setting when the total capacity is set to $\mu^* = \sum_{i=1}^n \nu_i\mu_i$, and policy π_{nh} is exercised, under which $\gamma_i^{\pi_{nh}}(t) \equiv \nu_i\mu_i/\mu^*$ for all $t \geq 0$ and all $i \in [\![1,n]\!]$.

Example 3 (flexible and collaborative servers). In this setting, there are $m \geq 1$ servers, each having a fixed capacity μ_j, $j \in [\![1,m]\!]$, and each can serve tasks at any of the n

queue-buffers. When more than one server is allocated to one queue-buffer, those servers collaborate on a single task, and the processing rate of that task equals the sum of all the individual server capacities that are handling it; we refer to such a system as FJR_2 system with flexible and collaborative servers, and denote it by FJR_2-FC. Formally, let $\mathbf{P} = (\mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_n) \in [\![1, m]\!]^n$ be a partition of the set of indices $[\![1, m]\!]$, i.e., $\mathbf{P}$ satisfies (i) $\mathbf{P}_i \subseteq [\![1, m]\!], \forall i \in [\![1, n]\!]$; (ii) $\cup_{i=1}^n \mathbf{P}_i = [\![1, m]\!]$; (iii) $\mathbf{P}_i \cap \mathbf{P}_j = \emptyset, \forall i \neq j$ (note that $\mathbf{P}_i$ can be empty). Let $\vec{\mu} := (\mu_1, \mu_2, \cdots, \mu_m) \in \mathbb{R}_+^m$ be such that $\sum_{j=1}^m \mu_j = \mu^*$. A capacity-allocation policy π_{FC} in an FJR_2-FC system is such that, at any given time t, $\boldsymbol{\gamma}^{\pi_{FC}}(t) \in \Delta_{\vec{\mu}}^n$, where

$$\begin{aligned}\Delta_{\vec{\mu}}^n := \{\boldsymbol{d} \in \Delta^n : &\text{there exists a partition } \mathbf{P} \text{ of } [\![1, m]\!] \\ &\text{such that } d_i = \sum_{j \in \mathbf{P}_i} \frac{\mu_j}{\mu^*}, \forall i \in [\![1, n]\!]\}. \end{aligned} \tag{1}$$

Note that $\pi_{FC} \in \Pi$ because $\Delta_{\vec{\mu}}^n \subseteq \Delta^n$.

As demonstrated in Example 1, static allocation policies are a special case of the policies in Π. In particular, a policy in Π is static, if there exists a capacity-allocation vector $\boldsymbol{d} \in \Delta^n$, such that $\boldsymbol{\gamma}^\pi(t) \equiv \boldsymbol{d}$, $\forall t \geq 0$. We refer to the static policy corresponding to $\boldsymbol{d}$ as static-$\boldsymbol{d}$ policy, and denote it by $\pi_{\boldsymbol{d}}$, as in Example 1. The class of static capacity-allocation policies, denoted by $\hat{\Pi}$, is the set of all static-$\boldsymbol{d}$ policies for all $\boldsymbol{d} \in \Delta^n$, i.e., $\hat{\Pi} := \{\pi_{\boldsymbol{d}} \in \Pi : \boldsymbol{d} \in \Delta^n\}$.

3.2. Stability Regions

Our goal is to characterize the stability regions of FJR_2 system that operate under policies in Π, as well as the *maximal* achievable stability region. Formally, for the traffic intensity $\rho := \lambda/\mu^*$, the stability region of an FJR_2 system operating under policy $\pi \in \Pi$ is the set

$$\begin{aligned}\mathcal{S}_\Pi(\pi) := \{\rho \in \mathbb{R}_+ : &N^\pi \textit{ is positive recurrent} \\ &\textit{when the traffic intensity is } \rho\}. \end{aligned}$$

The *maximal stability region* of a set of policies $\tilde{\Pi} \subseteq \Pi$, denoted by $\mathcal{S}^M(\tilde{\Pi})$, is then the largest stability region among all policies in $\tilde{\Pi}$, namely, $\mathcal{S}^M(\tilde{\Pi})$ satisfies (i) $\mathcal{S}_\Pi(\pi) \subseteq \mathcal{S}^M(\tilde{\Pi}), \forall \pi \in \tilde{\Pi}$; (ii) $\exists \pi^* \in \tilde{\Pi}$ such that $\mathcal{S}_\Pi(\pi^*) = \mathcal{S}^M(\tilde{\Pi})$. A policy $\pi^* \in \Pi$ is a maximal dynamic policy if $\mathcal{S}_\Pi(\pi^*) = \mathcal{S}^M(\Pi)$. Similarly, $\pi^* \in \hat{\Pi}$ is a maximal static policy if $\mathcal{S}_\Pi(\pi^*) = \mathcal{S}^M(\hat{\Pi})$

We study the class of static policies $\hat{\Pi}$ in §5, separately from the class of dynamic policies Π, which is studied in §6. Our analyses build on an equivalence relation between FJR_2 systems and a certain tandem queueing system, the latter being easier to analyze.

4. An Equivalent Tandem Queue

FJR_2 systems are difficult to analyze because a representation of their dynamics is high-dimensional. In this section, we first show in §4.1 that one can analyze FJR_2 systems using a three-dimensional state space. Furthermore, the dynamics of an FJR_2 system is equivalent to that of a two-station tandem queueing system, which is presented in §4.2, and the aforementioned equivalence is proved in Theorem 1 below.

4.1. State Descriptor for FJR_2 Systems

From the fact that all n tasks of a job are removed from the system simultaneously upon the service completion of any two of those tasks, it follows that all stations have the same number of tasks at any time, which is also the number of jobs in the system at that time, so that

$$N(t) = F_i(t) + J_i(t), \quad \forall i \in [\![1, n]\!] \text{ and } \forall t \geq 0. \tag{2}$$

Further, since tasks belonging to the same job enter and leave the system simultaneously, tasks in all stations at any time t are from the same $N(t)$ jobs.

Next, observe that there is at most one nonempty join-buffer at any time, namely, that

$$\|\boldsymbol{J}(t)\|_0 \leq 1, \forall t \geq 0. \tag{3}$$

Indeed, assume that Job A arrived before Job B, both Job A and Job B are currently in the system, and a Job-B task is waiting in join-buffer i at some time $t \geq 0$. Then, since FCFS is employed within each station, the Job-A task in station i was necessarily processed prior to time t, implying that there can be no Job-A task in any join-buffer j, $j \neq i$. In particular, if there is Job-A task that was already processed and is waiting in a join-buffer, then that Job-A task must be waiting in the *same* join-buffer i in which the Job-B task is waiting.

Let

$$I(t) := \arg \max_{i \in [\![1,n]\!]} \{J_i(t)\}$$

be the index of the station with the largest number of processed tasks its the join-buffer. It follows from (2) that $I(t)$ is also the index of the station with the smallest number of tasks in the queue-buffer, namely, that $I(t) := \arg\min_{i \in [\![1,n]\!]} \{F_i(t)\}$. To break ties which, due to (3), occur only when all join-buffers are empty, we take $I(t)$ to be the index of the join-buffer that was the last to become empty. Hereafter, when we refer to the *shortest queue at time t*, we refer to the queue whose queue-buffer is indexed by $I(t)$.

It follows from (2) and (3) that the $2n$-dimensional state of $(\boldsymbol{F}(t), \boldsymbol{J}(t)) \in \mathbb{Z}_+^{2n}$ can be recovered from the tuple $(F_{I(t)}(t), J_{I(t)}(t), I(t)) \in \mathbb{Z}_+^2 \times [\![1, n]\!]$ because

$$\begin{aligned} &N(t) = F_{I(t)}(t) + J_{I(t)}(t); \\ &F_i(t) = N(t) \text{ and } J_i(t) = 0, \quad \forall i \neq I(t). \end{aligned}$$

As before, we append $I^\pi(t)$ with superscript π to emphasize its dependence on the policy π, and let $F_I^\pi := \left\{F^\pi_{I^\pi(t)}(t) : t \geq 0\right\}$ and $J_I^\pi := \left\{J^\pi_{I^\pi(t)}(t) : t \geq 0\right\}$. We denote by $\gamma_I^\pi(t) := \gamma^\pi_{I^\pi(t)}(t)$ the proportion of the total capacity allocated to the server with the shortest queue at time t.

We conclude that an FJR_2 system under policy $\pi \in \Pi$ can be completely described by $(F_I^\pi, J_I^\pi, I^\pi, \boldsymbol{\gamma}^\pi)$, where $(F_I^\pi, J_I^\pi, I^\pi)$ is the three-dimensional state process controlled by the n-dimensional capacity-allocation process $\boldsymbol{\gamma}^\pi$.

4.2. The Equivalent Two-Station Tandem Queue

We now introduce the two-station tandem queueing system, denoted by TQ_2, which will be shown in Theorem 1 below to be equivalent to the FJR_2 system: The TQ_2 system has two stations in tandem, each having a single server and an infinite buffer. Jobs arrive to the system according to a Poisson process with rate λ, enter station 1, and after completing service with the server in that station, move into station 2. In both stations, the non-idling FCFS policy is employed. As in the FJR_2 system, we attach the service-time requirements to the jobs, so that each job is characterized via two independent exponentially distributed service-time requirements with mean 1—one for each server. For $i = 1, 2$, we denote by $Q_i(t)$ the number of jobs in station i at time t (including the job in service), by $\boldsymbol{Q} := (Q_1, Q_2)$ the queue-length process, and by $Q_\Sigma := Q_1 + Q_2$ the number-in-system process.

Capacity-Allocation Policies in TQ_2. We assume that the total service capacity in the TQ_2 system is fixed at μ^*, and can be redistributed between the two servers at transition epochs, namely, at arrival and service-completion times. Specifically, let $\mathcal{P}$ denote the set of all non-anticipative capacity-allocation policies. Then a policy $\mathbb{p} \in \mathcal{P}$ determines at each time t the value of a capacity-allocation process $\zeta^\mathbb{p} := \{\zeta^\mathbb{p}(t) : t \geq 0\}$ with values in $[0, 1]$ (we append all stochastic processes and random variables with superscript $\mathbb{p}$ to emphasize their dependence on the policy), such that

- If $Q_2^\mathbb{p}(t) > 0$, a proportion $\zeta^\mathbb{p}(t)$ of the total capacity is allocated to server 1, so that its service rate is $\zeta^\mathbb{p}(t)\mu^*$, and a proportion $1 - \zeta^\mathbb{p}(t)$ of μ^* is allocated to server 2;
- If $Q_2^\mathbb{p}(t) = 0$, then all the capacity is allocated to server 1, namely, the service rate of server 1 is μ^* at that time, *regardless of the value of* $\zeta^\mathbb{p}(t)$.

We emphasize that the value of $\zeta^\mathbb{p}(t)$ need not be equal to 1 at time t for which $Q_2^\mathbb{p}(t) = 0$, and that its exact value is determined by the policy $\mathbb{p}$. The reason for allowing $\zeta^\mathbb{p}(t)$ to have arbitrary values at such times will become clear in Theorem 1 below. Note also that if $Q_1^\mathbb{p}(t) = 0$ and $Q_2^\mathbb{p}(t) > 0$, not all capacity is necessarily allocated to server 2, since $\zeta^\mathbb{p}(t)$ need not equal 0 at such a time t. We conclude that a TQ_2 system under policy $\mathbb{p} \in \mathcal{P}$ is completely described by $(Q_1^\mathbb{p}, Q_2^\mathbb{p}, \zeta^\mathbb{p})$, where $(Q_1^\mathbb{p}, Q_2^\mathbb{p})$ is the two-dimensional state process controlled by a one-dimensional capacity-allocation process $\zeta^\mathbb{p}$.

Let $\rho := \lambda/\mu^*$ denote the traffic intensity, and define the stability region of a TQ_2 system, operating under policy $\mathbb{p} \in \mathcal{P}$, to be the set

$$\mathcal{S}_{\mathcal{P}}(\mathbb{p}) := \{\rho \in \mathbb{R}_+ : Q_\Sigma^{\mathbb{p}} \textit{ is positive recurrent when the traffic intensity is } \rho\}.$$

The next theorem establishes the aforementioned equivalence between the FJR_2 and TQ_2 systems.

THEOREM 1 **(Equivalence of FJR_2 and TQ_2).** *Consider FJR_2 and TQ_2 systems, both having the same arrival rate λ and the same total capacity μ^*. Then, for any policy $\pi \in \Pi$, there exists a policy $\mathbb{p} \in \mathcal{P}$, such that $(F_I^\pi, J_I^\pi, \gamma_I^\pi) \overset{\mathrm{d}}{=} (Q_1^{\mathbb{p}}, Q_2^{\mathbb{p}}, \zeta^{\mathbb{p}})$.*

We refer to policy $\mathbb{p}$ in Theorem 1 as the policy *induced* by policy π, and refer to the TQ_2 system under the induced policy $\mathbb{p}$ as the equivalent TQ_2 system.

REMARK 1. **(An alternative interpretation for the equivalent TQ_2 system).** We can classify jobs in an FJR_2 system into two phases: jobs in phase 0 that have no completed task, and jobs in phase 1 that have exactly one completed task. Then $F_I^\pi(t)$ is the number of phase-0 jobs and $J_I^\pi(t)$ is the number of phase-1 jobs in the system at time t. Therefore, in the equivalent TQ_2 system, jobs in station 1 correspond to phase-0 jobs in the equivalent FJR_2 system, and jobs in station 2 correspond to phase-1 jobs in that FJR_2 system.

It follows from Theorem 1 that the capacity-allocation process $\zeta^{\mathbb{p}}$ in the equivalent TQ_2 system corresponds to the process γ_I^π of the proportion allocated to the server with the shortest queue in the FJR_2 system. Moreover, since the equivalent TQ_2 system is controlled by $\zeta^{\mathbb{p}}$, Theorem 1 implies that policy π determines the dynamics of the FJR_2 system only via the process γ_I^π, namely, via the capacity of the server with the shortest queue. The next result follows immediately from Theorem 1.

COROLLARY 1. *Consider an FJR_2 system under policy $\pi \in \Pi$ and its equivalent TQ_2 system under the induced policy $\mathbb{p} \in \mathcal{P}$. Then $N^\pi \overset{\mathrm{d}}{=} Q_\Sigma^{\mathbb{p}}$, and in particular, $\mathcal{S}_\Pi(\pi) = \mathcal{S}_{\mathcal{P}}(\mathbb{p})$.*

Thus, by virtue of Corollary 1, the stability properties of an FJR_2 system can be determined by studying its equivalent TQ_2 system. The advantage of studying the equivalent TQ_2 system instead of the corresponding FJR_2 system is that the former has a two-dimensional state space, and has no redundancy—thus bypassing the need to handle the "phantom" tasks that are removed before being processed.

5. Stability and Maximal Design under Static Policies

In this section, we focus on the stability of the FJR_2 system under the class of static policies $\hat{\Pi}$. As was mentioned above, by "static policy" $\pi_{\boldsymbol{d}}$ we mean that a proportion d_i of the total capacity μ^* is allocated to station i, $i \in [\![1,n]\!]$, and that the allocation of the total capacity is fixed throughout. Thus, the study of the stability under a static policy is tantamount to determining how a system should be designed to ensure that it has the maximal stability region.

We first characterize the policy in TQ_2 induced by every static-$\boldsymbol{d}$ policy in FJR_2 (§5.1). We then characterize a necessary and sufficient condition for stability for the equivalent TQ_2 system under the induced policy, and thus, for the corresponding FJR_2 system (§5.2). Finally, we study the maximal design problem, namely, how to partition the total capacity between the n servers so as to achieve the maximal stability region (§5.3).

5.1. The Induced Policy in the Equivalent TQ_2 System

Consider the queue process in station 2, Q_2, in the TQ_2 system. Let $\tau_1 := 0$ and for $m \geq 2$, let $\tau_m := \inf\{t > \tau_{m-1} : Q_2(t) > 0, Q_2(t-) = 0\}$. Then $[\tau_m, \tau_{m+1})$ is the mth busy cycle of Q_2, where we take time 0 to be the beginning of the first busy cycle. (Note that each τ_m is finite w.p.1 whenever station 2 is stable.) Let $\boldsymbol{d}$ be the capacity-allocation vector corresponding to policy $\pi_{\boldsymbol{d}}$ in an FJR_2 system. For that vector $\boldsymbol{d}$, let $\mathbb{p}_{\boldsymbol{d}}$ be the following policy in a corresponding TQ_2 system that has the same arrival rate λ and total capacity μ^* as in the FJR_2 system: At the beginning of the mth cycle (of station 2), $m \geq 1$, $\mathbb{p}_{\boldsymbol{d}}$ assigns the value d_i to $\zeta^{\mathbb{p}_{\boldsymbol{d}}}$ with probability d_i, independent of everything else, and the value of $\zeta^{\mathbb{p}_{\boldsymbol{d}}}$ is kept fixed throughout the mth busy cycle. Specifically, for $m \geq 1$,

$$\begin{aligned} &\mathbb{P}(\zeta^{\mathbb{p}_{\boldsymbol{d}}}(\tau_m) = d_i) = d_i, \ i \in [\![1,n]\!] \quad \textit{and} \\ &\zeta^{\mathbb{p}_{\boldsymbol{d}}}(t) = \zeta^{\mathbb{p}_{\boldsymbol{d}}}(\tau_m) \quad \forall t \in (\tau_m, \tau_{m+1}). \end{aligned} \tag{4}$$

Proposition 1. *Policy $\mathbb{p}_{\boldsymbol{d}}$ is the policy induced in the equivalent TQ_2 system by policy $\pi_{\boldsymbol{d}}$.*

5.2. Stability of FJR_2 Systems under Static Policies

The explicit identification of policy $\mathbb{p}_{\boldsymbol{d}}$ in the equivalent TQ_2 system to the FJR_2 system under consideration allows us to characterize the stability regions of the two systems. For a vector $\boldsymbol{d} \in \Delta^n$, let

$$d_M = \max_{i \in [\![1,n]\!]} d_i. \tag{5}$$

Theorem 2. *The stability regions of an FJR_2 system under static-$\boldsymbol{d}$ policy $\pi_{\boldsymbol{d}} \in \hat{\Pi}$, and of the equivalent TQ_2 system under the induced policy $\mathbb{p}_{\boldsymbol{d}} \in \mathcal{P}$, are*

$$\mathcal{S}_{\Pi}(\pi_{\boldsymbol{d}}) = \mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\boldsymbol{d}}) = \left[0, \frac{1}{2} \wedge (1 - d_M)\right). \tag{6}$$

To intuitively see why condition $\rho < 1/2 \wedge (1 - d_M)$ is necessary for the TQ_2 system under the induced policy $\mathbb{p}_{\boldsymbol{d}}$ to be stable, note that both stations in the tandem system must each be stable. Since the service rate at each station depends on the choice of $\zeta^{\mathbb{p}_{\boldsymbol{d}}}$, whose value is reassigned at the beginning of each busy cycle of Q_2, station 2 must be stable for any possible value of $\zeta^{\mathbb{p}}$. Now, when station 1 is stable, its output rate, which is the input rate of station 2, is λ. Since the minimum service rate at station 2 is $1 - d_M$ when $\mathbb{p}_{\boldsymbol{d}}$ is employed, the input rate λ must satisfy $\lambda < (1 - d_M)\mu^*$ or, equivalently, $\rho < 1 - d_M$. In addition, each job is processed by both servers in the TQ_2 system, bringing with it a workload that is exponentially distributed with mean 1 to each server, for a total workload that is Erlang distributed with mean 2. On the other hand, the sum of the service rates of the two servers that the workload receives is at most μ^*. Since the rate at which workload arrives must be smaller than the maximal possible sum of the service rates, the inequality $2\lambda < \mu^*$ must hold, which is equivalent to $\rho < 1/2$.

The fact that ρ necessarily takes value in $\mathcal{S}_\Pi(\pi_{\boldsymbol{d}})$ in (6) can also be intuitively explained for the FJR_2 system directly. First, to see that the critical value of ρ under a static policy must not be larger than $1/2$, note that, since jobs arrive at rate λ, and since each job requires two of its tasks to be processed, the effective rate at which tasks arrive is 2λ. On the other hand, at most μ^* processing capacity can be allocated to each job, so that 2λ must be strictly smaller than μ^*, or equivalently, $\rho := \lambda/\mu^* < 1/2$. To see why ρ must also be smaller than $1 - d_M$, consider the case $d_M > 1/2$ and note that the total capacity allocated to all remaining $n - 1$ servers, other than the one with the largest capacity (referred to as the *fast server*), is $(1 - d_M)\mu^*$. Now, in a stationary system, most jobs have one of their completed tasks processed by the fast server, since it occupies more than one-half of the total capacity, and thus the second completed task is processed by one of the remaining $n - 1$ servers. Accordingly, the queue of the fast server should be shorter (on average) than the other queues, and become negligible compared to the other queues when the system is heavily loaded. Under such circumstances, the FJR_2 system behaves like a split system, where the first task of an incoming job is processed at rate $d_M\mu^*$ (by the fast server), and the second is processed at rate $(1 - d_M)\mu^*$. In particular, the arrival stream of those "second processed tasks" has rate λ, and their processing time is the minimum of the remaining $n - 1$ servers, whose combined rate is $(1 - d_M)\mu^*$. Hence, λ must be smaller than $(1 - d_M)\mu^*$ (equivalently, $\rho < 1 - d_M$) for the system to be stable.

5.2.1. d_M as a Measure of Heterogeneity. It follows from Theorem 2 that the stability region $\mathcal{S}_\Pi(\pi_{\boldsymbol{d}})$ is determined by the value of d_M: If $d_M \leq 1/2$, then the system is stable for all $\rho < 1/2$, so that the specific value of d_M has no impact on the stability region. However, the stability region is directly determined by the value of d_M when $d_M > 1/2$. We can therefore think of the

value of d_M as a measure of the heterogeneity of the servers. In particular, minimal heterogeneity is achieved when d_M is equal to its smallest possible value of $1/n$—in which case the servers are homogeneous—whereas maximal heterogeneity is achieved when d_M is equal to its largest possible value of 1 (in which case the system is unstable for any $\lambda > 0$).

To see why $d_M = 1/2$ is an inflection point, consider the case $d_M > 1/2$ and $\rho \in (1 - d_M, 1/2)$, so that the system is unstable. Let i_M be the station of the fast server which has the proportion d_M of the total capacity μ^*. Since tasks arrive to each station at rate λ, and since $\lambda < d_M \mu^*$ (because $\lambda < 1/2\mu^*$), the queue of unprocessed tasks in station i_M is stable. However, the remaining $n-1$ servers cannot handle the arriving work (since $\lambda > (1-d_M)\mu^*$), and thus their queues grow without bound w.p.1. Eventually, the system splits to two subsystems—one consisting of station i_M, and the other of the remaining $n-1$ servers, such that one task from each job is processed by the first subsystem (server i_M), and the other task is processed by one of the servers in the second subsystem. Clearly, all $n-1$ servers in the second subsystem are processing tasks from the same job at any time point, so that the effective processing rate of this subsystem is $(1-d_M)\mu^*$. It follows that the split system behaves like a two-station fork-join system with no redundancy, with one station processing tasks at rate $d_M\mu^*$ and the other at rate $(1-d_M)\mu^*$. Hence, d_M completely characterizes the heterogeneity of that split system.

5.3. Maximal Design of $\mathbf{FJR_2}$

We use Theorem 2 to derive the maximal stability region of the class of static polices, and the criterion for the maximal static policies. As given in (6) and depicted in Figure 2, the critical value of the stability region, as a function of d_M, is a piecewise-linear function with two pieces. Under low heterogeneity ($d_M \leq 1/2$), the critical value of ρ is fixed at $1/2$, and is independent of the specific value of d_M and of the service rates of all other servers (e.g., the capacity-allocation vectors $\boldsymbol{d}^1$ and $\boldsymbol{d}^2$ in Figure 2). Once the value of d_M increases above $1/2$, the critical value decreases linearly at a slope of -1. For example, $d_M = 3/4$ in the capacity-allocation vector $\boldsymbol{d}^3$ in the figure, so the critical value of ρ for this example is $1/4$. An immediate consequence is that a static policy in $\hat{\Pi}$ is a maximal static policy if and only if it assigns no more than $1/2$ of the total capacity to any of the servers, as stated formally in the following theorem.

THEOREM 3. *$\mathcal{S}^M(\hat{\Pi}) = [0, 1/2)$. Moreover, $\pi_{\boldsymbol{d}} \in \hat{\Pi}$ is a maximal static policy if and only if $d_M \leq 1/2$.*

It follows from Theorem 3 that the maximal stability region $[0, 1/2)$ is robust up to a certain level of server heterogeneity: As long as no server is allocated with more than one half of the total capacity, the stability region of the system remains maximal, and the specific allocation of the total capacity across all servers has no affect on stability. This result suggests that when designing

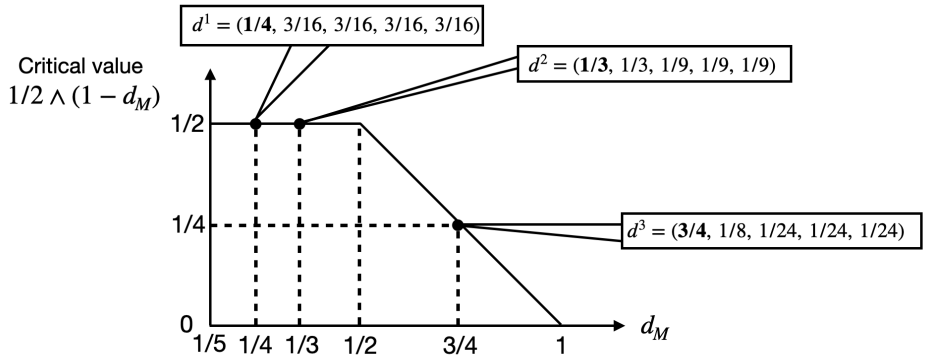

Figure 2 **The critical value vs. d_M in FJR$_2$ systems with 5 stations.**

a system (either from scratch, or by adding capacity to an existing system), server heterogeneity should be kept "low" (with $d_M < 1/2$). In fact, the heuristic explanation in §5.2.1 suggests that, for systems with $d_M > 1/2$, at least when the system is heavily loaded, the sojourn time of a job is, with high probability, determined by a task that is processed by one of the $n-1$ servers excluding the fastest server. This suggests that the lower the heterogeneity, the better the performance in terms of sojourn times.

6. Stability and Maximal Control under Dynamic Policies

We now consider the class of dynamic policies Π in which the capacity allocation among the servers can be dynamically redistributed.

6.1. Non-Maximality of Static Policies

Consider an FJR$_2$-FC system, as in Example 3, having $m \geq 1$ servers with rates $\mu_1, \ldots, \mu_m$, such that $\sum_{j=1}^{m} \mu_j = \mu^*$, and let $\vec{\mu} = (\mu_1, \mu_2, \cdots, \mu_m)$ denote the service-rates vector. In such a system, the total capacity μ^* cannot be split arbitrarily, and is always an additive combination of the m elements of $\vec{\mu}$. In a dynamic capacity-allocation policy, the controller decides which queue-buffer each capacity should be allocated to at any given time, so that the action space is $\Delta^n_{\vec{\mu}}$ defined in (1). Accordingly, for a given $\vec{\mu}$, we define two subclasses of Π corresponding to dynamic and static policies within the FJR$_2$-FC system, respectively:

$$\Pi_{\vec{\mu}} := \{\pi \in \Pi : \boldsymbol{\gamma}^\pi(t) \in \Delta^n_{\vec{\mu}}, \forall t \geq 0\} \quad \text{and}$$
$$\hat{\Pi}_{\vec{\mu}} := \{\pi_{\boldsymbol{d}} \in \hat{\Pi} : \boldsymbol{d} \in \Delta^n_{\vec{\mu}}\}.$$

Proposition 2. *Consider an FJR$_2$-FC system with a given $\vec{\mu}$. If $\mu_i > \mu^*/2$ for some $i \in [\![1, m]\!]$, then $S^M(\hat{\Pi}_{\vec{\mu}}) \subsetneq S^M(\hat{\Pi}) = [0, 1/2)$. In particular, no policy in $\hat{\Pi}_{\vec{\mu}}$ is a maximal static policy.*

Clearly, whether the maximal stability region of $\hat{\Pi}$ can be achieved by static policies in $\hat{\Pi}_{\vec{\mu}}$ depends on the specific components of the service-rates vector $\vec{\mu}$. In contrast, we show in Proposition 3 in §6.3 below that the maximal stability region of Π can be achieved with a dynamic policy in $\Pi_{\vec{\mu}}$, for any service-rates vector $\vec{\mu}$.

6.2. Bounds for the Stability Region

The stability region for a given dynamic policy cannot, in general, be characterized explicitly. We therefore begin by deriving useful bounds in §6.2.1 for stability regions of TQ_2 systems, and then translate the results to FJR_2 in §6.2.2. The derived bounds will be employed to characterize the maximal stability region of the class of dynamic policies, as well as the criterion for maximal dynamic policies, in §6.3.

6.2.1. Bounds for TQ_2 Systems. A simple subclass of policies in the equivalent TQ_2 systems is the set of policies that assign the same value to the capacity-allocation process ζ. For such a policy we can characterize the stability region explicitly.

Definition 1 (Fixed-value policies in TQ_2). A policy $\mathbb{p}_c \in \mathcal{P}$ is a fixed-value policy if $\zeta^{\mathbb{p}_c}(t) \equiv c, \forall t \geq 0$ and for some $c \in [0,1]$.

Lemma 1. *The stability region for a fixed-value policy $\mathbb{p}_c \in \mathcal{P}$ is*

$$\mathcal{S}_{\mathcal{P}}(\mathbb{p}_c) = \left[0, \frac{1}{2} \wedge (1-c)\right). \tag{7}$$

The next result proves that fixed-value policies provide bounds for stability regions under policies in $\mathcal{P}$.

Lemma 2. *If for two constants $\underline{c}$ and $\bar{c}$ satisfying $0 \leq \underline{c} \leq \bar{c} \leq 1$ and a policy $\mathbb{p} \in \mathcal{P}$ it holds that $\underline{c} \leq \zeta^{\mathbb{p}}(t) \leq \bar{c}, \forall t \geq 0$, then $\mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\bar{c}}) \subseteq \mathcal{S}_{\mathcal{P}}(\mathbb{p}) \subseteq \mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\underline{c}})$.*

6.2.2. Bounds for Dynamic Policies in FJR_2 Systems. We employ the bounds in Lemma 2 for the equivalent TQ_2 system to establish the corresponding bounds for the stability region in the original FJR_2 systems. To this end, note that for policy $\mathbb{p} \in \mathcal{P}$ induced by policy $\pi \in \Pi$, it holds that $\mathcal{S}_{\Pi}(\pi) = \mathcal{S}_{\mathcal{P}}(\mathbb{p})$ by virtue of Corollary 1. Moreover, if policy $\pi \in \Pi$ satisfies $\underline{c} \leq \gamma_I^{\pi}(t) \leq \bar{c},\ \forall t \geq 0$ for two constants $\underline{c}$, $\bar{c} \in [0,1]$, then its induced policy $\mathbb{p} \in \mathcal{P}$ in the equivalent TQ_2 system satisfies $\underline{c} \leq \zeta^{\mathbb{p}}(t) \leq \bar{c}, \forall t \geq 0$. It then follows from Lemma 2 that $\mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\underline{c}}) \subseteq \mathcal{S}_{\Pi}(\pi) = \mathcal{S}_{\mathcal{P}}(\mathbb{p}) \subseteq \mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\bar{c}})$. We summarize the above result in the following theorem.

Theorem 4. *If for a policy $\pi \in \Pi$ and two constants $\underline{c}$ and $\bar{c}$ it holds that $0 \leq \underline{c} \leq \gamma_I^{\pi}(t) \leq \bar{c} \leq 1$ $\forall t \geq 0$, then*

$$\mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\bar{c}}) \subseteq \mathcal{S}_{\Pi}(\pi) \subseteq \mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\underline{c}}). \tag{8}$$

6.3. Maximal Control

We now study the maximal-control problem in FJR_2 systems, namely, the problem of how to dynamically allocate the total service capacity such that the stability region is maximized. The proof of the next theorem, which appears in Appendix A.6, demonstrates how the bounds in Theorem 4 can be utilized to derive the criterion for maximal dynamic policies.

THEOREM 5. *$\mathcal{S}^M(\Pi) = [0, 1/2)$. Moreover, any policy $\pi \in \Pi$ with $\gamma_I^\pi(t) \le 1/2\ \forall t \ge 0$ is a maximal dynamic policy.*

In particular, a dynamic policy is maximal if the proportion of the total capacity allocated to the server with the shortest queue is no larger than $1/2$ at any time.

Static Policies vs. Dynamic Policies in FJR_2 Systems. It follows from Theorem 3 and Theorem 5 that $\mathcal{S}^M(\hat{\Pi}) = \mathcal{S}^M(\Pi) = [0, 1/2)$. Therefore, dynamic policies do not enlarge the maximal stability region compared to static policies. Nevertheless, the criterion for a dynamic policy to be maximal is weaker than for static policies. Indeed, maximal static policies require that the capacity allocated to *each server* does not exceed one half of the total capacity. On the other hand, for a dynamic policy to be maximal, it suffices to have the capacity allocated to the *server with the shortest queue* be no larger than one half of the total capacity at any given time. In particular, the capacity of any other server might be larger than $\mu^*/2$. We summarize the results in Table 1.

	Class of Static Policies $\hat{\Pi}$	Class of Dynamic Policies Π
Maximal Stability Region	$\mathcal{S}^M(\hat{\Pi}) = [0, 1/2)$	$\mathcal{S}^M(\Pi) = [0, 1/2)$
Criterion to Achieve the Maximal Stability Region	$d_M \le 1/2$ (necessary and sufficient): *The capacity of each server does not exceed $\mu^*/2$.*	$\gamma_I(t) \le 1/2, \forall t \ge 0$ (sufficient): *The capacity of the server with the shortest queue does not exceed $\mu^*/2$ at any time.*

Table 1 Comparison of the maximal stability region and the criterion to achieve it for static and dynamic policies.

Static policies vs. dynamic policies in FJR_2-FC systems. Consider FJR_2-FC systems with a given $\vec{\mu}$. If $\mu_i > \mu^*/2$ for some $i \in [\![1, m]\!]$ then, by Proposition 2, there exists no static policy in $\hat{\Pi}_{\vec{\mu}}$ that is maximal. This, however, is not the case for dynamic policies, as stated in the next proposition.

PROPOSITION 3. *For FJR_2-FC system with a service-rates vector $\vec{\mu}$, it holds that $\mathcal{S}^M(\Pi_{\vec{\mu}}) = \mathcal{S}^M(\Pi) = [0, 1/2)$.*

In particular, there exists a maximal dynamic policy $\pi^* \in \Pi_{\vec{\mu}}$ for FJR_2-FC systems, regardless of its service-rates vector $\vec{\mu}$.

6.4. Discussion on the Proof of Lemma 2

To prove the bounds for $\mathcal{S}_\Pi(\pi)$ in Lemma 2, we show that for the three policies $\mathbb{p}_{\underline{c}}$, $\mathbb{p}$ and $\mathbb{p}_{\bar{c}}$, the following sample-path stochastic-order relations hold:

$$Q_\Sigma^{\underline{c}} \leq_{st} Q_\Sigma^{\mathbb{p}} \leq Q_\Sigma^{\bar{c}}, \tag{9}$$

given that $\boldsymbol{Q}^{\underline{c}}(0) = \boldsymbol{Q}^{\mathbb{p}}(0) = \boldsymbol{Q}^{\bar{c}}(0)$. We focus on the stochastic inequality on the lef-hand side, noting that the second stochastic inequality can be proved using similar arguments.

We refer to the systems under policies $\mathbb{p}_{\underline{c}}$ and $\mathbb{p}$ as system L and system U, respectively. The proof employs a coupling argument and induction on the event times T_m, $m \geq 0$, where an event is an arrival or a service completion from either system L or U. However, there are two difficulties in coupling these two systems together on the same probability space: First, either system has state-dependent transition rates, and those dependencies are different in the two systems. Moreover, when both stations have jobs in both systems, the service rate of station 1 in system L is no larger than that in system U (since $\underline{c} \leq \zeta^{\mathbb{p}}(t)$), whereas the service rate of station 2 in system L is no smaller than that in system U (since $1 - \underline{c} \geq 1 - \zeta^{\mathbb{p}}(t)$), making it difficult to synchronize the next service completion. We resolve this issue by a service-time synchronization procedure, which we refer to as *rate decomposition*. The second, and more significant difficulty with the coupling argument, is that (9) cannot be shown to hold at event time T_{m+1} when assuming it holds at time T_m, because some events may occur in one system and not in the other in such a way that the inequalities in (9) are violated. We solve this problem by considering an extra inequality which "filters out" the possibility of such order-violating events, and prove that, if that *filter inequality* and (9) hold initially, then the filter inequality and (9) hold together throughout.

7. Stability of (n, k) Fork-Join Systems

We now consider (n, k) systems, with $n > k > 2$. Specifically, each job consists of n tasks as before, but requires $k > 2$ tasks to be processed, so that $n - k$ tasks are redundant. We denote the class of dynamic capacity-allocation policies by $\Pi^{(n,k)}$, and the class of static capacity-allocation policies by $\hat{\Pi}^{(n,k)}$.

Whereas establishing the maximal stability region in this more general setting is beyond the scope of this paper, we nevertheless establish a lower bound for the maximal stability region among all static policies in $\hat{\Pi}^{(n,k)}$, which is also a lower bound for the maximal stability region among all dynamic policies in $\Pi^{(n,k)}$. We conjecture that the lower bound is tight, namely, that it is the actual maximal stability region, and build on insights from our analysis of FJR_2 systems to heuristically devise a criterion for the maximal static policies in (n, k) systems. Numerical studies support our criterion.

7.1. Lower Bound for the Maximal Stability Region

We derive a lower bound for the maximal stability region $\mathcal{S}^M(\hat{\Pi}^{(n,k)})$ of the class of static policies, which is also a lower bound for $\mathcal{S}^M(\Pi^{(n,k)})$ (note that $\mathcal{S}^M(\hat{\Pi}^{(n,k)}) \subseteq \mathcal{S}^M(\Pi^{(n,k)})$ since $\hat{\Pi}^{(n,k)} \subsetneq \Pi^{(n,k)}$). We do so by constructing a special static policy whose stability region could be derived in closed form.

PROPOSITION 4. *For (n,k) systems, it holds that $[0,1/k) \subseteq \mathcal{S}^M(\hat{\Pi}^{(n,k)})$, so that $[0,1/k) \subseteq \mathcal{S}^M(\Pi^{(n,k)})$.*

We conjecture that the lower bound in Proposition 4 is tight, namely, that $\mathcal{S}^M(\hat{\Pi}^{(n,k)}) = \mathcal{S}^M(\Pi^{(n,k)}) = [0,1/k)$. To see why, note that, since the arrival rate to the system is λ and each job requires the completions of k tasks, the *effective arrival rate* of tasks that end up being processed is $k\lambda$. It therefore stands to reason that $k\lambda < \mu^*$ is a necessary condition in order for the system to be stable, implying that $\mathcal{S}^M(\hat{\Pi}^{(n,k)}) \subseteq \mathcal{S}^M(\Pi^{(n,k)}) \subseteq [0,1/k)$. These latter containment relations, together with Proposition 4, would then imply our conjecture. We further conjecture that a static-$\boldsymbol{d}$ policy is maximal if and only if $d_M \leq 1/k$. We summarize in the following statement.

CONJECTURE 1. *For an (n,k) system, the following hold.*

1. $\mathcal{S}^M(\hat{\Pi}^{(n,k)}) = \mathcal{S}^M(\Pi^{(n,k)}) = [0,1/k)$.
2. *A static-$\boldsymbol{d}$ policy $\pi_{\boldsymbol{d}} \in \hat{\Pi}^{(n,k)}$ is maximal if and only if $d_M \leq 1/k$.*

7.2. Simulation Study

For the simulation experiments, we consider a $(5,3)$ FJR system, normalizing μ^* to be equal to 1. We consider two different capacity-allocation vectors, with $d_M \in \{1/5, 1/3\}$; specifically, we take $\boldsymbol{d}^1 = (1/5, 1/5, 1/5, 1/5, 1/5)$ and $\boldsymbol{d}^2 = (1/3, 1/6, 1/6, 1/6, 1/6)$. To test our conjecture, we choose two traffic intensities: $\rho \in \{0.332, 0.334\}$—one slightly above, and the other slightly below the conjectured critical value $1/3 \approx 0.333$. Figure 3 depicts the simulated sample paths of the number-in-system processes N for each of the two capacity-allocation vectors and each of the two traffic intensities. Each sample path is generated with 10^6 arrivals.

We observe that, when $d_M \leq 1/3$ and $\rho = 0.332$, the sample paths keep returning to 0, suggesting that the systems are stable. However, the sample paths for those two systems when $\rho = 0.334$ seem to drift away from the empty state, suggesting that both systems are unstable under this traffic intensity. Together, the simulated sample paths in Figure 3 suggest that the critical value of the stability region is within a small neighborhood of $1/3$, and provide support to our conjecture that the critical value is equal to $1/3$, and such critical value is achieved when $d_M \leq 1/3$.

To provide support for our conjecture that the maximal stability region cannot be achieved with $d_M > 1/3$, we simulate $(5,3)$ FJR systems with $d_M \in \{0.34, 1/2, 2/3\}$ and search for the

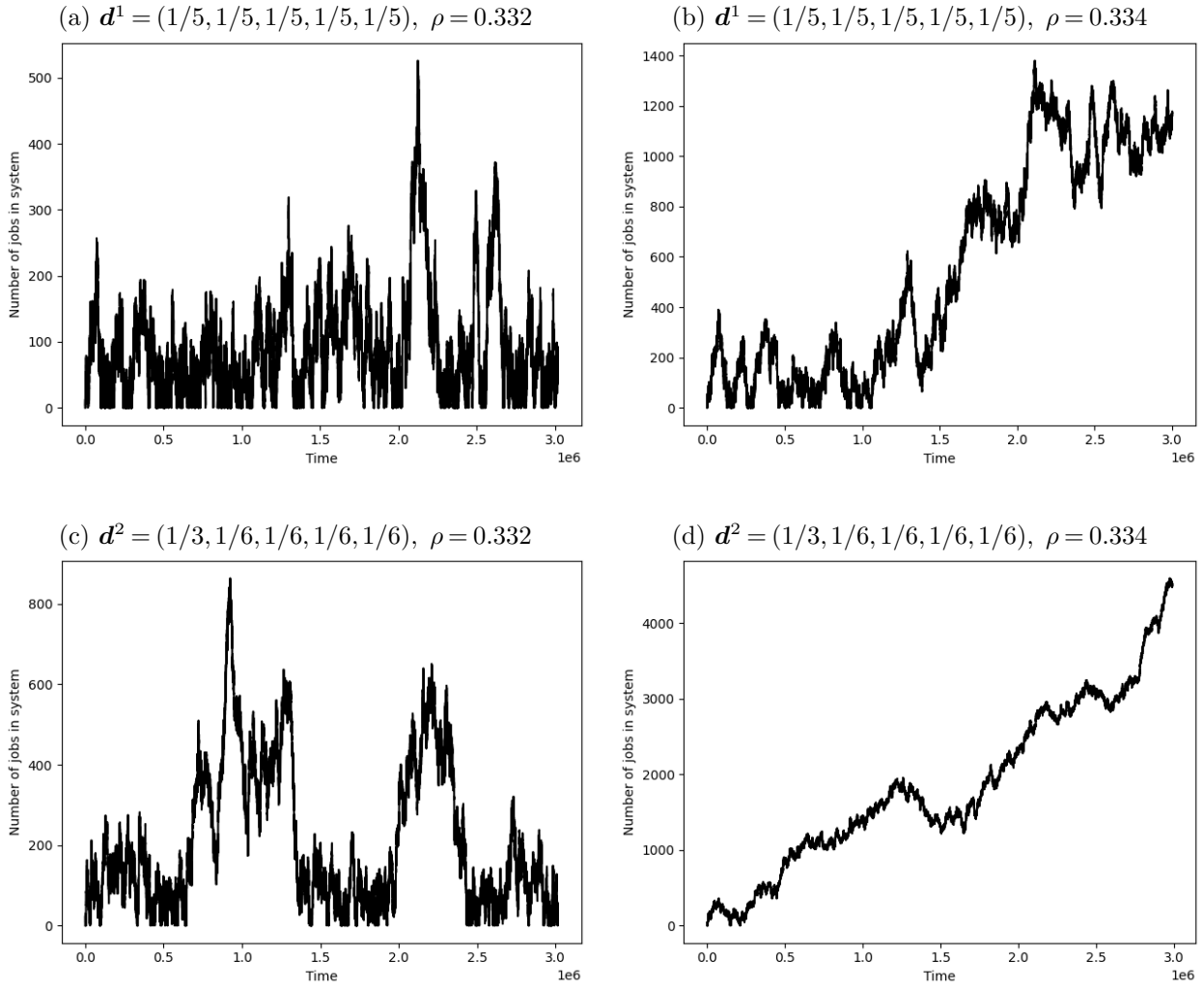

Figure 3 **Sample paths of the number-in-system processes in** $(5,3)$ **systems with values of** ρ **that are close to the conjectured critical values.**

neighborhood of their critical values. Two sample paths for each of these systems are shown in Figure 4—one of (what seems to be) a stationary system, and the other of a transient one. In particular, the critical value seems to be in the neighborhood $[0.330, 0.332]$ when $d_M = 0.34$ (Panels (a) and (b) of the figure), while it seems to be in the neighborhood of $[0.249, 0.251]$ when $d_M = 1/2$ (Panels (c) and (d)), and in the neighborhood of $[0.165, 0.167]$ when $d_M = 2/3$ (Panels (e) and (f)). These simulation experiments demonstrate that the maximal stability region is not achieved when $d_M > 1/3$ and suggest that the stability region is decreasing in d_M.

8. Summary

In this paper we studied the stability of fork-join systems with redundancy of the form $(n, 2)$ under static policies—having a fixed allocation of the total capacity, which incorporates the case

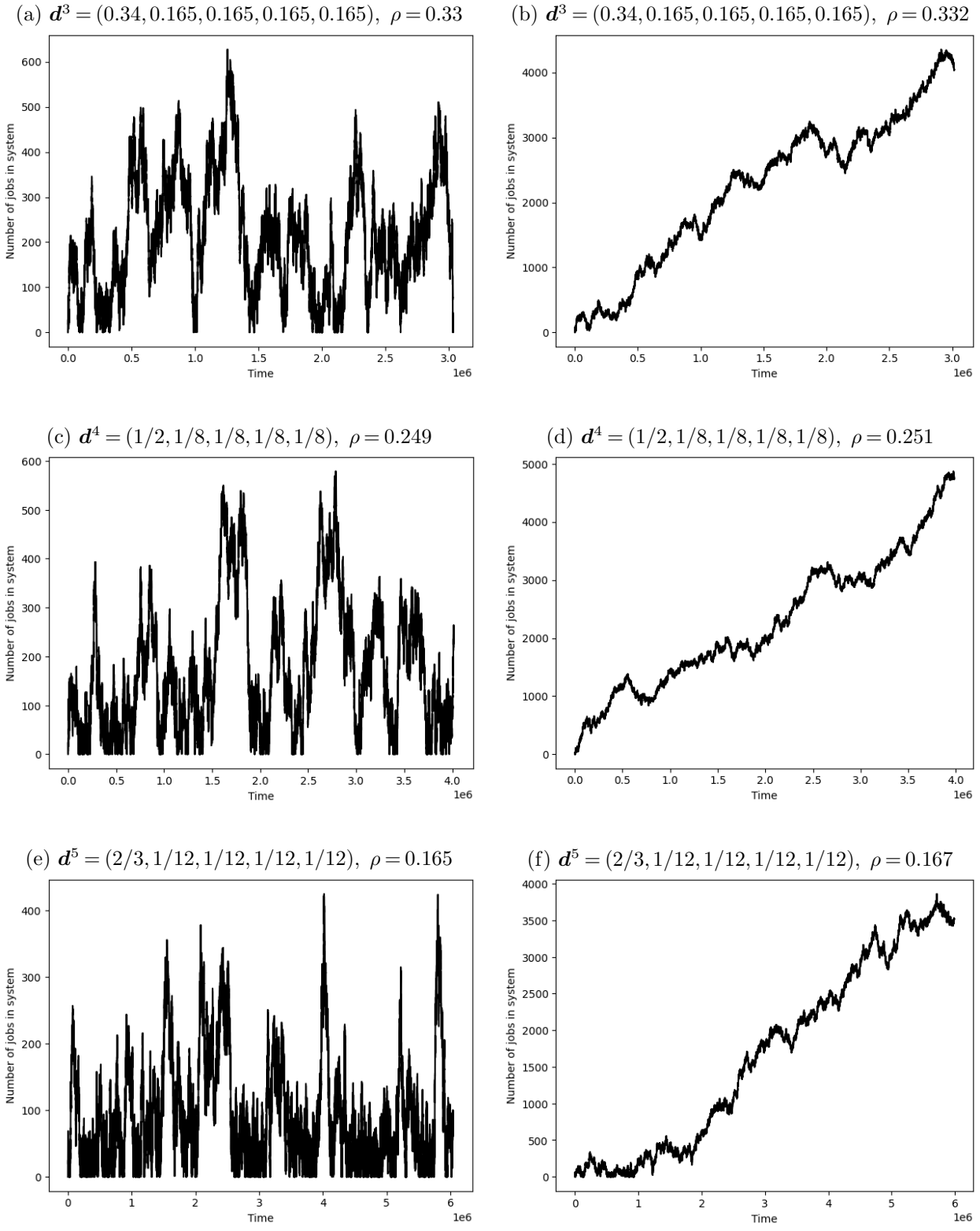

Figure 4 **Sample paths of the number-in-system processes in $(5,3)$ systems under policies $\pi_{\boldsymbol{d}^3}, \pi_{\boldsymbol{d}^4}, \pi_{\boldsymbol{d}^5}$.**

of heterogeneous servers—and dynamic policies, in which a given capacity can be continuously redistributed among the different stations. To determine the stability region and the maximal stability region under either class of policies, we presented a two-station tandem queueing system, whose dynamics, under a specific policy, are equivalent to those of the FJR system, so that both systems have the same stability properties.

We first showed that under static policies, the stability region is of the form $[0, 1/2 \wedge (1 - d_M))$, where d_M is the proportion of the total capacity that is allocated to the server with the largest capacity. Therefore, the maximal stability region is achieved if and only if a static policy assigns no more than $1/2$ of the total capacity to each server, so that $d_M \leq 1/2$.

Next, we considered dynamic policies, and derived lower and upper bounds for the stability region of any such policy. Using these bounds, we showed that a sufficient criterion to maximize the stability region is to allocate no more than $1/2$ of the total capacity to the server with the shortest queue at any time. In particular, the maximal stability region of dynamic policies is the same as that of static ones. However, instead of requiring that the capacity allocated to all servers be no larger than $1/2$ of the total capacity, as is the case for static policies, the maximal stability region of dynamic policies is achieved as long as the server with the shortest queue at any given time does not receive more than $1/2$ of the total capacity.

Finally, for general (n, k) FJR systems with $2 < k < n$, we proved that $[0, 1/k)$ yields a lower bound for the maximal stability region for both classes of static and dynamic policies. Drawing on the insights obtained for the $(n, 2)$ special case, we conjectured that the maximal stability region of an (n, k) system is indeed $[0, 1/k)$, and that it is achieved if and only if $d_M \leq 1/k$. We employed simulation experiments to support this conjecture.

There are many directions for future work. First, it remains to study the stability properties of (n, k) systems, and to prove that $[0, 1/k)$ is indeed the maximal stability region for these systems under static and dynamic policies. Second, while our work focused on the most fundamental performance measure (stability), there are natural questions regarding which maximal policies are "better" than others, e.g., because they minimize prespecified steady-state performance measures such as the expected sojourn time.

References

Andradóttir, S., H. Ayhan. 2005. Throughput maximization for tandem lines with two stations and flexible servers. *Operations Research* **53**(3) 516–531.

Andradóttir, S., H. Ayhan, D. G. Down. 2003. Dynamic server allocation for queueing networks with flexible servers. *Operations Research* **51**(6) 952–968.

Anthropic. 2024. Building effective agents. URL `https://www.anthropic.com/research/building-effective-agents`. Accessed: February 2, 2025.

Anton, E., U. Ayesta, M. Jonckheere, I. M. Verloop. 2021. On the stability of redundancy models. *Operations Research* **69**(5) 1540–1565.

Baccelli, F., A. M. Makowski, A. Shwartz. 1989. The fork-join queue and related systems with synchronization constraints: Stochastic ordering and computable bounds. *Advances in Applied Probability* **21**(3) 629–660.

Bassamboo, A., R. S. Randhawa, J. A. V. Mieghem. 2012. A little flexibility is all you need: on the asymptotic value of flexible capacity in parallel queuing systems. *Operations Research* **60**(6) 1423–1435.

Carmeli, N., G. B. Yom-Tov, O. J. Boxma. 2023. State-dependent estimation of delay distributions in fork-join networks. *Manufacturing & Service Operations Management* **25**(3) 1081–1098.

Chu, C.-T., S. Kim, Y.-A. Lin, Y. Yu, G. Bradski, K. Olukotun, A. Ng. 2006. Map-reduce for machine learning on multicore. *Advances in neural information processing systems* **19**.

Dai, J., J. M. Harrison. 2020. *Processing networks: fluid models and stability*. Cambridge University Press.

Down, D. G., M. E. Lewis. 2006. Dynamic load balancing in parallel queueing systems: Stability and optimal control. *European Journal of Operational Research* **168**(2) 509–519.

El-Taha, M., M. Stidham, S. Stidham Jr. 1999. *Sample-path analysis of queueing systems*, vol. 11. Springer Science & Business Media.

Fayolle, G., R. Iasnogorodski, V. Malyshev. 2017. *Random Walks in the Quarter Plane: Algebraic Methods, Boundary Value Problems, Applications to Queueing Systems and Analytic Combinatorics*, vol. 40. Springer.

Flatto, L., S. Hahn. 1984. Two parallel queues created by arrivals with two demands i. *SIAM Journal on Applied Mathematics* **44**(5) 1041–1053.

Gandhi, A., M. Harchol-Balter, R. Das, C. Lefurgy. 2009. Optimal power allocation in server farms. *ACM SIGMETRICS Performance Evaluation Review* **37**(1) 157–168.

Gardner, K., M. Harchol-Balter, A. Scheller-Wolf, M. Velednitsky, S. Zbarsky. 2017. Redundancy-d: The power of d choices for redundancy. *Operations Research* **65**(4) 1078–1094.

Gardner, K., E. Hyytiä, R. Righter. 2019. A little redundancy goes a long way: Convexity in redundancy systems. *Performance Evaluation* **131** 22–42.

Gardner, K., C. Stephens. 2019. Smart dispatching in heterogeneous systems. *ACM SIGMETRICS Performance Evaluation Review* **47**(2) 12–14.

Gardner, K., S. Zbarsky, S. Doroudi, M. Harchol-Balter, E. Hyytiä, A. Scheller-Wolf. 2016. Queueing with redundant requests: exact analysis. *Queueing Systems* **83**(3) 227–259.

Harchol-Balter, M. 2021. Open problems in queueing theory inspired by datacenter computing. *Queueing Systems* **97**(1) 3–37.

Hu, S., X. Chen, W. Ni, E. Hossain, X. Wang. 2021. Distributed machine learning for wireless communication networks: Techniques, architectures, and applications. *IEEE Communications Surveys & Tutorials* **23**(3) 1458–1493.

Joshi, G., Y. Liu, E. Soljanin. 2012. Coding for fast content download. *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 326–333.

Joshi, G., Y. Liu, E. Soljanin. 2014. On the delay-storage trade-off in content download from coded distributed storage systems. *IEEE Journal on Selected Areas in Communications* **32**(5) 989–997.

Joshi, G., E. Soljanin, G. Wornell. 2015. Queues with redundancy: Latency-cost analysis. *ACM SIGMETRICS Performance Evaluation Review* **43**(2) 54–56.

Joshi, G., E. Soljanin, G. Wornell. 2017. Efficient redundancy techniques for latency reduction in cloud systems. *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)* **2**(2) 1–30.

Ko, S.-S., R. F. Serfozo. 2008. Sojourn times in g/m/1 fork-join networks. *Naval Research Logistics (NRL)* **55**(5) 432–443.

Kulkarni, V. G. 2016. *Modeling and analysis of stochastic systems*. Crc Press.

Lee, K., M. Lam, R. Pedarsani, D. Papailiopoulos, K. Ramchandran. 2017a. Speeding up distributed machine learning using codes. *IEEE Transactions on Information Theory* **64**(3) 1514–1529.

Lee, K., N. B. Shah, L. Huang, K. Ramchandran. 2017b. The mds queue: Analysing the latency performance of erasure codes. *IEEE Transactions on Information Theory* **63**(5) 2822–2842.

Li, T., A. K. Sahu, A. Talwalkar, V. Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine* **37**(3) 50–60.

Marin, A., S. Rossi. 2016. Dynamic control of the join-queue lengths in saturated fork-join stations. *International Conference on Quantitative Evaluation of Systems*. Springer, 123–138.

Meijer, M. S., D. Schol, W. van Jaarsveld, M. Vlasiou, B. Zwart. 2024. Optimization of inventory and capacity in large-scale assembly systems using extreme-value theory. *Stochastic Systems* .

Moyal, P., O. Perry. 2022. Stability of parallel server systems. *Operations Research* **70**(4) 2456–2476.

Nelson, R., A. N. Tantawi. 1988. Approximate analysis of fork/join synchronization in parallel queues. *IEEE transactions on computers* **37**(6) 739–743.

Ouyang, L., J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* **35** 27730–27744.

Özkan, E. 2022. Control of fork-join processing networks with multiple job types and parallel shared resources. *Mathematics of Operations Research* **47**(2) 1310–1334.

Özkan, E., A. R. Ward. 2019. On the control of fork-join networks. *Mathematics of Operations Research* **44**(2) 532–564.

Pedarsani, R., J. Walrand, Y. Zhong. 2014a. Robust scheduling in a flexible fork-join network. *53rd IEEE Conference on Decision and Control*. IEEE, 3669–3676.

Pedarsani, R., J. Walrand, Y. Zhong. 2014b. Scheduling tasks with precedence constraints on multiple servers. *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 1196–1203.

Pedarsani, R., J. Walrand, Y. Zhong. 2017. Robust scheduling for flexible processing networks. *Advances in Applied Probability* **49**(2) 603–628.

Rizk, A., F. Poloczek, F. Ciucu. 2016. Stochastic bounds in fork–join queueing systems under full and partial mapping. *Queueing Systems* **83**(3-4) 261–291.

Schol, D., M. Vlasiou, B. Zwart. 2022. Large fork-join queues with nearly deterministic arrival and service times. *Mathematics of Operations Research* **47**(2) 1335–1364.

Sethuraman, S. 2022. *Analysis of fork-join systems: Network of queues with precedence constraints*. CRC Press.

Shah, N. B., K. Lee, K. Ramchandran. 2015. When do redundant requests reduce latency? *IEEE Transactions on Communications* **64**(2) 715–722.
Thomasian, A. 2014. Analysis of fork/join and related queueing systems. *ACM Computing Surveys (CSUR)* **47**(2) 1–71.
Varki, E. 1999. Mean value technique for closed fork-join networks. *ACM SIGMETRICS Performance Evaluation Review* **27**(1) 103–112.
Wang, W., M. Harchol-Balter, H. Jiang, A. Scheller-Wolf, R. Srikant. 2019. Delay asymptotics and bounds for multi-task parallel jobs. *ACM SIGMETRICS Performance Evaluation Review* **46**(3) 2–7.
Whitt, W. 1991. The pointwise stationary approximation for mt/mt/s queues is asymptotically correct as the rates increase. *Management Science* **37**(3) 307–314.

Appendix A: Proofs

We present the proofs of the statements in the order that they appear in the paper, except for the proof of Theorem 2 which appears at the end of the appendix, in §A.9.

A.1. Proof of Theorem 1

Proof. We characterize the transition probabilities of the process $(F_I^\pi, J_I^\pi, I^\pi, \boldsymbol{\gamma}^\pi)$, and demonstrate that $(F_I^\pi, J_I^\pi, \gamma_I^\pi)$ is indeed distributed as a TQ_2 system. For a given time $t \geq 0$, we condition on the event $\{(F_I^\pi(t), J_I^\pi(t)) = (i,j)\}$, $i,j \geq 0$, and consider all the possible state transitions of the process (F_I^π, J_I^π). First, an arrival of a job increases F_I^π by 1, so that, for $i,j \geq 0$ and as $h \to 0^+$,

$$\mathbb{P}(F_I^\pi(t+h) = i+1, J_I^\pi(t+h) = j \mid$$
$$F_I^\pi(t) = i, J_I^\pi(t) = j, \gamma_I^\pi(t)) = \lambda h + o(h),$$

where $o(h)$ denotes a function f such that $f(h)/h \to 0$ as $h \to 0^+$.

Next, consider the service completion of a task. We consider all possible cases separately.

- **Case 1.** $i > 0, j > 0$.

1.a) The next event is the completion of a task from queue-buffer $I^\pi(t)$, which happens at rate $\gamma_I^\pi(t)\mu^*$. Since all other join-buffers are empty, this task has no other task to be matched with, and thus remains in the join-buffer. Therefore,

$$\mathbb{P}(F_I^\pi(t+h) = i-1, J_I^\pi(t+h) = j+1 \mid$$
$$F_I^\pi(t) = i, J_I^\pi(t) = j, \gamma_I^\pi(t)) = \gamma_I^\pi(t)\mu^* h + o(h).$$

1.b) The next event is the completion of a task from some queue-buffer $k \neq I^\pi(t)$, which occurs at rate $\gamma_k^\pi(t)\mu^*$. Note that the combined processing rate of all the servers excluding server $I^\pi(t)$ is $(1-\gamma_I^\pi(t))\mu^*$. Since the newly completed task has a match in join-buffer $I^\pi(t)$, the number of tasks in join-buffer $I^\pi(t)$ decreases by 1, and the number of tasks in queue-buffer $I^\pi(t)$ remains unchanged. Further, whether join-buffer $I^\pi(t)$ empties or not after the match occurs, the value of I^π remains the same. Thus,

$$\mathbb{P}(F_I^\pi(t+h) = i, J_I^\pi(t+h) = j-1 \mid$$
$$F_I^\pi(t) = i, J_I^\pi(t) = j, \gamma_I^\pi(t)) = (1-\gamma_I^\pi(t))\mu^* h + o(h).$$

Case 2. $i > 0, j = 0$. In this case, all the queue- and join-buffers have the same length at time t, and the next service completion of a task occurs with rate μ^*. The completed task moves to its join-buffer, which becomes the unique nonempty joint-buffer with exactly one task. Thus,

$$\mathbb{P}(F_I^\pi(t+h) = i-1, J_I^\pi(t+h) = 1 \mid$$
$$F_I^\pi(t) = i, J_I^\pi(t) = 0, \gamma_I^\pi(t)) = \mu^* h + o(h).$$

Case 3. $i=0, j>0$. The combined service rate for all servers, excluding the one at station $I^\pi(t)$, is $(1-\gamma_I^\pi(t))\mu^*$, and so

$$\mathbb{P}(F_I^\pi(t+h)=0, J_I^\pi(t+h)=j-1 \mid F_I^\pi(t)=0,$$
$$J_I^\pi(t)=j, \gamma_I^\pi(t)) = (1-\gamma_I^\pi(t))\,\mu^* h + o(h).$$

We summarize the infinitesimal transition rates of the process (F_I^π, J_I^π) under the capacity-allocation process γ_I^π, in Table 2. It is easy to see that the infinitesimal transition rates in Table 2 are also the infinitesimal

values of (i,j) at time t	transition to state	Transition rate
$i\geq 0, j\geq 0$	$(i+1,j)$	λ
$i>0, j>0$	$(i-1,j+1)$	$\gamma_I^\pi(t)\mu^*$
	$(i,j-1)$	$(1-\gamma_I^\pi(t))\mu^*$
$i>0, j=0$	$(i-1,j+1)$	μ^*
$i=0, j>0$	$(i,j-1)$	$(1-\gamma_I^\pi(t))\mu^*$

Table 2 Transition rates of (F_I^π, J_I^π) at time $t\geq 0$ given the state $(F_I^\pi(t), J_I^\pi(t))=(i,j)$ and the control $\gamma_I^\pi(t)$.

transition rates of a TQ_2 system under the capacity-allocation process $\zeta^{\mathbb{p}}=\gamma_I^\pi$ when $(Q_1^{\mathbb{p}}, Q_2^{\mathbb{p}})$ is the state process; in particular, it holds that $(F_I^\pi, J_I^\pi, \gamma_I^\pi) \overset{\mathrm{d}}{=} (Q_1^{\mathbb{p}}, Q_2^{\mathbb{p}}, \zeta^{\mathbb{p}})$. □

A.2. Proof of Proposition 1

Before proving the proposition, we elaborate on the evolution of the shortest-queue process I. For $\tilde{\tau}_0 := 0$, $\tilde{\mathcal{T}}_0 := 0$ and $m\geq 1$, let $\tilde{\tau}_m = \inf\{t>\tilde{\tau}_{m-1} : ||\boldsymbol{J}(t)||_0 = 1, ||\boldsymbol{J}(t-)||_0 = 0\}$, and $\tilde{\mathcal{T}}_m = \inf\{t>\tilde{\tau}_m : ||\boldsymbol{J}(t)||_0 = 0, ||\boldsymbol{J}(t-)||_0 = 1\}$. We call $[\tilde{\tau}_m, \tilde{\tau}_{m+1})$ the mth busy cycle of process $\boldsymbol{J}$, and $[\tilde{\tau}_m, \tilde{\mathcal{T}}_m)$ the mth busy period of $\boldsymbol{J}$. Note that $[\tilde{\tau}_m, \tilde{\mathcal{T}}_m) \subseteq [\tilde{\tau}_m, \tilde{\tau}_{m+1})$.

LEMMA 3. *The following holds for all $\pi\in\Pi$ and all $m\geq 0$.*

(i) $\mathbb{P}(I(\tilde{\tau}_m)=i) = \gamma_i(\tilde{\tau}_m-),\ i\in[\![1,n]\!]$;

(ii) $I(t)\equiv I(\tilde{\tau}_m),\ \forall t\in(\tilde{\tau}_m, \tilde{\tau}_{m+1})$.

Proof. At the beginning of the mth busy cycle of $\boldsymbol{J}$, a task enters the join-buffer with all other join-buffers being empty. Since the service rate of server i is $\gamma_i(\tilde{\tau}_m-)\mu^*$ immediately before time $\tilde{\tau}_m$, the probability that the task comes from queue-buffer i is $(\gamma_i(\tilde{\tau}_m-)\mu^*) \,/\, \left(\sum_{j=1}^n \gamma_j(\tilde{\tau}_m-)\mu^*\right) = \gamma_i(\tilde{\tau}_m-)$, which proves part (i) of the statement.

It is easy to see that the unique longest join-buffer queue, which we denote here by i_ℓ, remains the longest throughout a busy period. Indeed, if a task is processed and enters a join-buffer $i\neq i_\ell$, then that task necessarily belongs to a job whose task in queue-buffer i_ℓ was already processed and is waiting in join-buffer i_ℓ. Thus, the two tasks immediately leave join-buffers i and i_ℓ, and all the redundant tasks from the same job are removed from the system at the same time, implying that all the join-buffers, other than possibly join-buffer i_ℓ, remain empty. (Join-buffer i_ℓ may also become empty at this point, in which case the busy period ends.) Therefore, we conclude that $I(t)=I(\tilde{\tau}_m)$ for all $t\in(\tilde{\tau}_m, \tilde{T}_m)$. Due to the definition of I, we have $I(t)=I(\tilde{T}_m-)=I(\tilde{\tau}_m)$ for all $t\in[\tilde{T}_m, \tilde{\tau}_{m+1})$, as stated in (ii) of the statement. □

Proof of Proposition 1. In this proof, we assume all processes are under policy $\pi_{\boldsymbol{d}}$, unless otherwise specified. Note that for the static policy $\pi_{\boldsymbol{d}}$, it holds that $\gamma_i(t) = d_i$ for all $t \geq 0$ and $i \in [\![1, n]\!]$. By Lemma 3, I does not change its value during a busy cycle of $\boldsymbol{J}$, so that $d_{I(t)}$ remains fixed at $d_{I(\tilde{\tau}_m)}$ for any $t \in (\tilde{\tau}_m, \tilde{\tau}_{m+1})$, $m \geq 0$. Therefore,

$$\gamma_I(t) = d_{I(t)} = d_{I(\tilde{\tau}_m)} = \gamma_I(\tilde{\tau}_m),\ \forall t \in (\tilde{\tau}_m, \tilde{\tau}_{m+1}).$$

By part (i) of Lemma 3, $\mathbb{P}(I(\tilde{\tau}_m) = i) = \gamma_i(\tilde{\tau}_m -) = d_i$. Hence, $\mathbb{P}(\gamma_I(\tilde{\tau}_m) = \gamma_i(\tilde{\tau}_m)) = d_i$; equivalently, $\mathbb{P}(\gamma_I(\tilde{\tau}_m) = d_i) = d_i$. We conclude that under the policy $\pi_{\boldsymbol{d}}$,

$$\begin{aligned} &\mathbb{P}(\gamma_I^{\pi_{\boldsymbol{d}}}(\tilde{\tau}_m) = d_i) = d_i,\ i \in [\![1, n]\!], \quad \text{and} \\ &\gamma_I^{\pi_{\boldsymbol{d}}}(t) = \gamma_I^{\pi_{\boldsymbol{d}}}(\tilde{\tau}_m)\ \forall t \in (\tilde{\tau}_m, \tilde{\tau}_{m+1}), \quad \text{for all } m \geq 0. \end{aligned} \tag{10}$$

Let $\tilde{\mathbb{p}}_{\boldsymbol{d}}$ be the policy induced by $\pi_{\boldsymbol{d}}$. By Theorem 1, we have $(F_I^{\pi_{\boldsymbol{d}}}, J_I^{\pi_{\boldsymbol{d}}}, \gamma_I^{\pi_{\boldsymbol{d}}}) \overset{\mathrm{d}}{=} \left(Q_1^{\tilde{\mathbb{p}}_{\boldsymbol{d}}}, Q_2^{\tilde{\mathbb{p}}_{\boldsymbol{d}}}, \zeta^{\tilde{\mathbb{p}}_{\boldsymbol{d}}}\right)$. Replacing $J_I^{\pi_{\boldsymbol{d}}}$ by $Q_2^{\tilde{\mathbb{p}}_{\boldsymbol{d}}}$, $\gamma_I^{\pi_{\boldsymbol{d}}}$ by $\zeta^{\tilde{\mathbb{p}}_{\boldsymbol{d}}}$, $\tilde{\tau}_m$ by τ_m in (10) gives

$$\begin{aligned} &\mathbb{P}(\zeta^{\tilde{\mathbb{p}}_{\boldsymbol{d}}}(\tau_m) = d_i) = d_i,\ i \in [\![1, n]\!], \quad \text{and} \\ &\zeta^{\tilde{\mathbb{p}}_{\boldsymbol{d}}}(t) = \zeta^{\tilde{\mathbb{p}}_{\boldsymbol{d}}}(\tau_m)\ \forall t \in (\tau_m, \tau_{m+1}). \end{aligned} \tag{11}$$

(11) completely characterizes the policy $\tilde{\mathbb{p}}_{\boldsymbol{d}}$. Comparing (11) with (4), which describes a policy $\mathbb{p}_{\boldsymbol{d}}$, we conclude that $\tilde{\mathbb{p}}_{\boldsymbol{d}}$ and $\mathbb{p}_{\boldsymbol{d}}$ are the same policy, and in turn, $\mathbb{p}_{\boldsymbol{d}}$ is the policy induced by policy $\pi_{\boldsymbol{d}}$. □

A.3. Proof of Proposition 2

Proof. Consider any partition $\mathbf{P} = (\mathbf{P}_1, \cdots, \mathbf{P}_n)$ of $[\![1, m]\!]$, so that the corresponding capacity-allocation vector $\boldsymbol{d} \in \Delta_{\vec{\mu}}^n$ satisfies $d_k = \sum_{j \in \mathbf{P}_k} \mu_j / \mu^*$ for any $k \in [\![1, n]\!]$. For $\vec{\mu}$ with $\mu_i > \mu^*/2$, it must hold that $i \in \mathbf{P}_k$ for some $k \in [\![1, n]\!]$. Therefore, we have $d_M \geq d_k = \sum_{j \in \mathbf{P}_k} \mu_j / \mu^* \geq \mu_i / \mu^* > 1/2$. By Theorem 3, $\pi_{\boldsymbol{d}}$ is not a maximal static policy. Since the partition $\mathbf{P}$ is arbitrarily, we conclude that $S^M(\hat{\Pi}_{\vec{\mu}}) \subsetneq S^M(\hat{\Pi})$. □

A.4. Proof of Lemma 1

For notational convenience, we henceforth denote by $\boldsymbol{Q}^c = (Q_1^c, Q_2^c)$ the queue-length process in a TQ_2 system under a fixed-value policy $\mathbb{p}_c$, where $c \in [0, 1]$. Namely, $\boldsymbol{Q}^c := \boldsymbol{Q}^{\mathbb{p}_c}$. Under a fixed-value policy $\mathbb{p}_c$, $\boldsymbol{Q}^c$ is a continuous-time Markov chain (CTMC). We prove the result by finding the stability condition for the discrete-time Markov chain (DTMC) embedded in the transition times of $\boldsymbol{Q}^c$, and then by applying the drift condition in (Fayolle et al. 2017, Theorem 1.2.1), which we state here in Theorem 6 below for completeness.

To this end, consider first the partition of $\mathbb{Z}_+^2$ into the following four disjoint subspaces: $\mathbb{S} = \{(x, y) \in \mathbb{Z}_+^2 : x \geq 1, y \geq 1\}$ (the interior); $\mathbb{S}' = \{(x, y) \in \mathbb{Z}_+^2 : x \geq 1, y = 0\}$ (the x-axis); $\mathbb{S}'' = \{(x, y) \in \mathbb{Z}_+^2 : x = 0, y \geq 1\}$ (the y-axis); and $\mathbb{S}^0 = \{(0, 0)\}$ (the origin). Let $\mathcal{L} = \{(\mathcal{L}_{n,x}, \mathcal{L}_{n,y}) \in \mathbb{Z}_+^2 : n \geq 0\}$ be a random walk in $\mathbb{Z}_+^2$ that (i) has homogeneous transitions starting from any states within each subspace, and (ii) has at most one-unit jumps each time in both the x-axis and the y-axis. Namely, the transition probability $p_{\alpha,\alpha+(i,j)}$ from the state α to the state $\alpha + (i, j)$ is invariant for any $\alpha \in \mathbb{S}$ (resp. $\mathbb{S}', \mathbb{S}'', \mathbb{S}^0$), and $p_{\alpha,\alpha+(i,j)} = 0$ for any $|i|, |j| \geq 2$. We let $p_{ij} := p_{\alpha,\alpha+(i,j)}$, $i, j \in \{-1, 0, 1\}$, for an arbitrarily chosen $\alpha \in \mathbb{S}$, and analogously define $p'_{ij}, p''_{ij}, p^0_{ij}$

when α is an element in $\mathbb{S}'$, $\mathbb{S}''$ or $\mathbb{S}^0$, respectively. Let $\boldsymbol{M} = (M_x, M_y) \in \mathbb{Z}_+^2$ be the mean drift vector of $\mathcal{L}$ in $\mathbb{S}$, where, for $\alpha \in \mathbb{S}$,

$$\begin{aligned} M_x &:= \mathbb{E}[\mathcal{L}_{n+1,x} - \mathcal{L}_{n,x} | (\mathcal{L}_{n,x}, \mathcal{L}_{n,y}) = \alpha] \quad \text{and} \\ M_y &:= \mathbb{E}[\mathcal{L}_{n+1,y} - \mathcal{L}_{n,y} | (\mathcal{L}_{n,x}, \mathcal{L}_{n,y}) = \alpha]. \end{aligned} \tag{12}$$

(Note that the choice of α in $\mathbb{S}$ does not change the value of $\boldsymbol{M}$ since $\mathcal{L}$ has state-homogeneous transition probabilities within each subspaces.) We analogously define the mean drift vectors $\boldsymbol{M}' := (M_x', M_y')$, $\boldsymbol{M}'' := (M_x'', M_y'')$ by taking α in (12) to be an element in $\mathbb{S}'$ and $\mathbb{S}''$, respectively.

THEOREM 6 **(Theorem 1.2.1 in Fayolle et al. (2017))**. *When $\boldsymbol{M} \neq (0,0)$, the random walk $\mathcal{L}$ is positive recurrent if and only if one of the following conditions holds:*

$$\begin{aligned} &\textit{C1: } M_x < 0,\ M_y < 0,\ M_x M_y' - M_y M_x' < 0, \\ &\qquad\qquad\qquad\qquad M_y M_x'' - M_x M_y'' < 0; \\ &\textit{C2: } M_x < 0,\ M_y \geq 0,\ M_y M_x'' - M_x M_y'' < 0; \\ &\textit{C3: } M_x \geq 0,\ M_y < 0,\ M_x M_y' - M_y M_x' < 0. \end{aligned}$$

The next lemma will be used to verify that $\boldsymbol{M} \neq (0,0)$, which is required in order to apply Theorem 6.

LEMMA 4. *If $\boldsymbol{Q}^c$ is positive recurrent, then $\lambda < (1-c)\mu^*$.*

Proof of Lemma 4. If Q_1^c is positive recurrent, then the long-run departure rate from station 1 must equal the long-run arrival rate λ to station 1 (see, e.g., (El-Taha et al. 1999, Lemma 1.5)), which is the long-run arrival rate to station 2. Under policy $\mathbb{p}_c$, the service rate in station 2 is fixed at $(1-c)\mu^*$ and thus, if Q_2^c is positive recurrent, then it must hold that $\lambda < (1-c)\mu^*$.

Proof of Lemma 1. Let $\{\tilde{\boldsymbol{Q}}_n^c\}$ be the embedded DTMC of the CTMC $\boldsymbol{Q}^c$, i.e., $\{\tilde{\boldsymbol{Q}}_n^c\} := \{\boldsymbol{Q}^c(\bar{T}_n) : n \geq 0\}$, where $\bar{T}_n$ is the nth transition time of $\boldsymbol{Q}^c$, with $\bar{T}_0 := 0$. The DTMC $\{\tilde{\boldsymbol{Q}}_n^c\}$ is a 2-dimensional random walk in $\mathbb{Z}_+^2$ which satisfies the assumptions on $\mathcal{L}$ (see Figure 5 for an illustration), and has the following one-step transition probabilities:

$$\begin{bmatrix} p_{-1,1} & p_{0,1} & p_{1,1} \\ p_{-1,0} & p_{0,0} & p_{1,0} \\ p_{-1,-1} & p_{0,-1} & p_{1,-1} \end{bmatrix} = \begin{bmatrix} \frac{c\mu^*}{\lambda+\mu^*} & 0 & 0 \\ 0 & 0 & \frac{\lambda}{\lambda+\mu^*} \\ 0 & \frac{(1-c)\mu^*}{\lambda+\mu^*} & 0 \end{bmatrix},$$

$$\begin{bmatrix} p'_{-1,1} & p'_{0,1} & p'_{1,1} \\ p'_{-1,0} & p'_{0,0} & p'_{1,0} \\ p'_{-1,-1} & p'_{0,-1} & p'_{1,-1} \end{bmatrix} = \begin{bmatrix} \frac{\mu^*}{\lambda+\mu^*} & 0 & 0 \\ 0 & 0 & \frac{\lambda}{\lambda+\mu^*} \\ 0 & 0 & 0 \end{bmatrix},$$

$$\begin{bmatrix} p''_{-1,1} & p''_{0,1} & p''_{1,1} \\ p''_{-1,0} & p''_{0,0} & p''_{1,0} \\ p''_{-1,-1} & p''_{0,-1} & p''_{1,-1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{\lambda}{\lambda+(1-c)\mu^*} \\ 0 & \frac{(1-c)\mu^*}{\lambda+(1-c)\mu^*} & 0 \end{bmatrix},$$

$$\begin{bmatrix} p^0_{-1,1} & p^0_{0,1} & p^0_{1,1} \\ p^0_{-1,0} & p^0_{0,0} & p^0_{1,0} \\ p^0_{-1,-1} & p^0_{0,-1} & p^0_{1,-1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

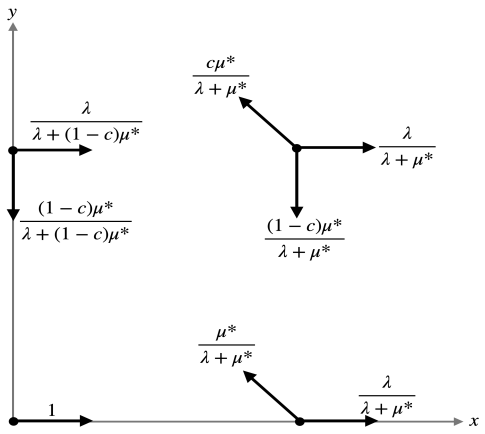

Figure 5 **Transition probabilities of the random walk $\{\hat{\boldsymbol{Q}}_n^c\}$.**

Given the transition probabilities above, the mean drift vectors of $\{\tilde{\boldsymbol{Q}}_n^c\}$ satisfy

$$\begin{aligned}
\boldsymbol{M} &= \left(\sum_{i,j\in\{-1,0,+1\}} ip_{ij} \quad , \sum_{i,j\in\{-1,0,+1\}} jp_{ij} \right) \\
&= \left(\frac{\lambda - c\mu^*}{\lambda+\mu^*}, \frac{(2c-1)\mu^*}{\lambda+\mu^*} \right), \\
\boldsymbol{M}' &= \left(\sum_{i,j\in\{-1,0,+1\}} ip'_{ij} \quad , \sum_{i,j\in\{-1,0,+1\}} jp'_{ij} \right) \\
&= \left(\frac{\lambda - \mu^*}{\lambda+\mu^*}, \frac{\mu^*}{\lambda+\mu^*} \right), \\
\boldsymbol{M}'' &= \left(\sum_{i,j\in\{-1,0,+1\}} ip''_{ij} \quad , \sum_{i,j\in\{-1,0,+1\}} jp''_{ij} \right) \\
&= \left(\frac{\lambda}{\lambda+(1-c)\mu^*}, \frac{(c-1)\mu^*}{\lambda+(1-c)\mu^*} \right).
\end{aligned}$$

Furthermore,

$$\begin{aligned}
M_x M'_y - M_y M'_x &= \frac{(1-c)\mu^*(2\lambda-\mu^*)}{(\lambda+\mu^*)^2}; \\
M_y M''_x - M_x M''_y &= \frac{c\mu^*(\lambda-(1-c)\mu^*)}{(\lambda+\mu^*)(\lambda+(1-c)\mu^*)}.
\end{aligned}$$

Now, if $\boldsymbol{M} = (0,0)$, then $c = 1/2$ and $\lambda = \mu^*/2$, so that, in particular, $\lambda = (1-c)\mu^*$, which in turn implies that $\boldsymbol{Q}^c$ is unstable by virtue of Lemma 4. Therefore, we need only consider the case $\boldsymbol{M} \neq (0,0)$, for which Theorem 6 can be applied. To this end, we consider different values of c. Under each case, we establish the set of inequalities that makes one of the three conditions C1, C2 or C3 in Theorem 6 hold. We first note that $\boldsymbol{Q}^c$ is trivially transient when $c = 1$, because the second station does not receive capacity at any time, so that only value of c in $[0,1)$ need to be considered.

When $c = 0$, neither C1 nor C2 can hold because $M_x = \lambda/(\lambda+\mu^*) > 0$. In this case, one of C1, C2, or C3 holds if and only if C3 holds. Since $M_y = -\mu^*/(\lambda+\mu^*) < 0$ is always satisfied, C3 holds if and only if the inequality

$$M_x M'_y - M_y M'_x = \frac{(2\lambda-\mu^*)\mu^*}{(\lambda+\mu^*)^2} < 0$$

holds, which in turn holds if and only if $\lambda < \mu^*/2$. We conclude that when $c = 0$, the DTMC $\{\tilde{\boldsymbol{Q}}_n^c\}$ is positive recurrent if and only if $\lambda < \mu^*/2$, or equivalently, if and only if $\rho < 1/2$.

When $c \in (0,1)$, we consider the three conditions C1, C2 and C3, respectively. First, C1 holds if the following four inequalities hold.

$$\begin{aligned} M_x &= \frac{\lambda - c\mu^*}{\lambda + \mu^*} < 0, \\ M_y &= \frac{(2c-1)\mu^*}{\lambda + \mu^*} < 0, \\ M_x M_y' - M_y M_x' &= \frac{(1-c)\mu^*(2\lambda - \mu^*)}{(\lambda + \mu^*)^2} < 0, \\ M_y M_x'' - M_x M_y'' &= \frac{c\mu^*(\lambda - (1-c)\mu^*)}{(\lambda + \mu^*)(\lambda + (1-c)\mu^*)} < 0. \end{aligned}$$

It is easy to check that the four inequalities above hold simultaneously if and only if $c < 1/2$ and $\lambda < c\mu^*$, or equivalently, if and only if $\rho < c < 1/2$.

Next, for C2 to hold, we require that, in addition to $M_x < 0$ and $M_y \geq 0$, the following inequality holds.

$$M_y M_x'' - M_x M_y'' = \frac{c\mu^*(\lambda - (1-c)\mu^*)}{(\lambda + \mu^*)(\lambda + (1-c)\mu^*)} < 0.$$

These three inequalities hold together if and only if $c \geq 1/2$ and $\lambda < (1-c)\mu^*$, namely, if and only if $\rho < 1 - c \leq 1/2$.

Finally, C3 is satisfied if, in addition to $M_x \geq 0$ and $M_y < 0$, it holds that

$$M_x M_y' - M_y M_x' = \frac{(1-c)\mu^*(2\lambda - \mu^*)}{(\lambda + \mu^*)^2} < 0.$$

These three inequalities hold together if and only if $c < 1/2$ and $c\mu^* \leq \lambda < \mu^*/2$, or equivalently, if and only if $c \leq \rho < 1/2$.

Overall, exactly one of the conditions C1, C2 and C3 holds when $c \in (0,1)$ if and only if

$$\rho < 1/2 \wedge (1-c).$$

To summarize, the DTMC $\{\tilde{\boldsymbol{Q}}_n^c\}$ is positive recurrent if and only if $\rho < 1/2 \wedge (1-c)$, for any $c \in [0,1]$. (Note that the case $c = 1$ is incorporated, in which case the stability condition is never satisfied, so that $\{\tilde{\boldsymbol{Q}}_n^c\}$ is not positive recurrent.) Finally, the transition rates of the CTMC $\boldsymbol{Q}^c$ are bounded from below by λ and from above by $\lambda + \mu^*$, and it is thus positive recurrent whenever its embedded DTMC is positive recurrent; see, e.g., (Kulkarni 2016, Theorem 6.18). Hence, $\mathcal{S}_{\mathcal{P}}(\mathbb{p}_c) = [0, 1/2 \wedge (1-c))$ as stated. □

A.5. Proof of Lemma 2

The proof of Lemma 2 is based on the following result, whose proof is given immediately after the proof of Lemma 2.

LEMMA 5. *Consider a policy $\mathbb{p} \in \mathcal{P}$ and two fixed-value policies $\mathbb{p}_{\underline{c}}$ and $\mathbb{p}_{\bar{c}}$. If $0 \leq \underline{c} \leq \zeta^{\mathbb{p}}(t) \leq \bar{c} < 1, \forall t \geq 0$, and $\boldsymbol{Q}^{\mathbb{p}}(0) = \boldsymbol{Q}^{\underline{c}}(0) = \boldsymbol{Q}^{\bar{c}}(0)$, then*

$$\begin{aligned} &Q_\Sigma^{\underline{c}} \leq_{st} Q_\Sigma^{\mathbb{p}} \leq_{st} Q_\Sigma^{\bar{c}} \quad \textit{and} \\ &2Q_1^{\underline{c}} + Q_2^{\underline{c}} \leq_{st} 2Q_1^{\mathbb{p}} + Q_1^{\mathbb{p}} \leq_{st} 2Q_1^{\bar{c}} + Q_2^{\bar{c}}. \end{aligned} \tag{13}$$

Proof of Lemma 2 Since $\boldsymbol{Q}^{\bar{c}}$, with $\bar{c} \in [0,1)$, is an irreducible CTMC, it is positive recurrent if and only if any of its states is. It follows from Lemma 5 that if the three processes $\boldsymbol{Q}^{\underline{c}}$, $\boldsymbol{Q}^{\mathbb{p}}$ and $\boldsymbol{Q}^{\bar{c}}$ are initialized at the same state in $\mathbb{Z}_+^2$, then

$$Q_\Sigma^{\underline{c}}(t) \le Q_\Sigma^{\mathbb{p}}(t) \le Q_\Sigma^{\bar{c}}(t), \ \forall t \ge 0, \ \text{w.p.1.}$$

Hence, the positive recurrence of process $Q_\Sigma^{\mathbb{p}}$ follows from the positive recurrence of $Q_\Sigma^{\bar{c}}$. In turn, the positive recurrence of $Q_\Sigma^{\underline{c}}$ follows from that of $Q_\Sigma^{\mathbb{p}}$. Therefore, $\mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\bar{c}}) \subseteq \mathcal{S}_{\mathcal{P}}(\mathbb{p}) \subseteq \mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\underline{c}})$, as stated. □

Proof of Lemma 5. We prove the statement via a coupling argument and induction. We first prove the stochastic-order inequalities between $\boldsymbol{Q}^{\underline{c}}$ and $\boldsymbol{Q}^{\mathbb{p}}$. To this end, we construct two state processes $\mathcal{Q}^{\underline{c}} = (\mathcal{Q}_1^{\underline{c}}, \mathcal{Q}_2^{\underline{c}})$ and $\mathcal{Q}^{\mathbb{p}} = (\mathcal{Q}_1^{\mathbb{p}}, \mathcal{Q}_2^{\mathbb{p}})$ for two TQ_2 systems under policy $\mathbb{p}_{\underline{c}}$ and $\mathbb{p}$, respectively, on a common probability space, such that $\mathcal{Q}^{\underline{c}} \overset{\mathrm{d}}{=} \boldsymbol{Q}^{\underline{c}}$ and $\mathcal{Q}^{\mathbb{p}} \overset{\mathrm{d}}{=} \boldsymbol{Q}^{\mathbb{p}}$. We refer to the systems corresponding to processes $\mathcal{Q}^{\underline{c}}$ and $\mathcal{Q}^{\mathbb{p}}$ as system L and system U.

Synchronizing the next event by rate decomposition. We begin by giving systems L and U the same arrival process of jobs. We initialize both systems at the same state, i.e., $\mathcal{Q}^{\mathbb{p}}(0) = \mathcal{Q}^{\underline{c}}(0)$. For $m \ge 0$, let T_m be the mth event time (arrival or service completion from either system), with $T_0 := 0$. Suppose that at time T_m, the control of system L is $\zeta^{\mathbb{p}_{\underline{c}}}(T_m) \equiv \underline{c}$, and that of system U is $\zeta^{\mathbb{p}}(T_m)$, which satisfies $\underline{c} \le \zeta^{\mathbb{p}}(T_m)$ by assumption. Then the next event in either system is generated as follows:

(i) At time T_m, we generate four mutually independent exponential random variables: $A^{(m)}$ with rate λ; $S_1^{(m)}$ with rate $\underline{c}\mu^*$; $S_2^{(m)}$ with rate $(\zeta^{\mathbb{p}}(T_m) - \underline{c})\,\mu^*$; and $S_3^{(m)}$with rate $(1 - \zeta^{\mathbb{p}}(T_m))\,\mu^*$. $A^{(m)}$ is used to determine the next potential interarrival time to both systems, and $S_1^{(m)}$, $S_2^{(m)}$ and $S_3^{(m)}$ are used to determine the potential remaining service time at station i of system j, $i = 1,\ 2$ and $j = \mathrm{L},\ \mathrm{U}$, which is deonted by $R_{i,j}^{(m)}$. In particular, these remaining service time are determined according to the procedure listed in Table 3; note that $R_{i,j}^{(m)} := \infty$ when station i in system j is empty at time T_m.

(ii) The next event time is scheduled at $T_{m+1} = T_m + M^{(m)}$, where

$$M^{(m)} := \min\{A^{(m)},\ R_{1,\mathrm{L}}^{(m)},\ R_{2,\mathrm{L}}^{(m)},\ R_{1,\mathrm{U}}^{(m)},\ R_{2,\mathrm{U}}^{(m)}\}.$$

For example, if $M^{(m)} = A^{(m)}$, then the next event at time T_{m+1} is an arrival to both systems; if $M^{(m)} = R_{2,\mathrm{L}}^{(m)} = R_{1,\mathrm{U}}^{(m)}$, then the next event is a departure from station 2 in system L and a departure from station 1 in system U.

We next prove that

$$\begin{aligned} &\mathcal{Q}_\Sigma^{\underline{c}}(t) \le \mathcal{Q}_\Sigma^{\mathbb{p}}(t) \ \text{ and} \\ &2\mathcal{Q}_1^{\underline{c}}(t) + \mathcal{Q}_2^{\underline{c}}(t) \le 2\mathcal{Q}_1^{\mathbb{p}}(t) + \mathcal{Q}_2^{\mathbb{p}}(t), \ \forall t \ge 0 \quad \text{w.p.1.} \end{aligned} \tag{14}$$

Since the processes are all piecewise constant, changing values only at event times, it suffices to show that, for any $m \ge 0$,

$$\begin{aligned} &\mathcal{Q}_\Sigma^{\underline{c}}(T_m) \le \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) \ \text{ and} \\ &2\mathcal{Q}_1^{\underline{c}}(T_m) + \mathcal{Q}_2^{\underline{c}}(T_m) \le 2\mathcal{Q}_1^{\mathbb{p}}(T_m) + \mathcal{Q}_2^{\mathbb{p}}(T_m), \quad \text{w.p.1,} \end{aligned} \tag{15}$$

system L	Cases	$R_{1,\mathrm{L}}^{(m)}$	$R_{2,\mathrm{L}}^{(m)}$
	i. $\mathcal{Q}_1^{\mathrm{c}}(T_m)>0, \mathcal{Q}_2^{\mathrm{c}}(T_m)>0$	$S_1^{(m)}$	$\min\{S_2^{(m)}, S_3^{(m)}\}$
	ii. $\mathcal{Q}_1^{\mathrm{c}}(T_m)>0, \mathcal{Q}_2^{\mathrm{c}}(T_m)=0$	$\min\{S_1^{(m)}, S_2^{(m)}, S_3^{(m)}\}$	∞
	iii. $\mathcal{Q}_1^{\mathrm{c}}(T_m)=0, \mathcal{Q}_2^{\mathrm{c}}(T_m)>0$	∞	$\min\{S_2^{(m)}, S_3^{(m)}\}$
	iv. $\mathcal{Q}_1^{\mathrm{c}}(T_m)=0, \mathcal{Q}_2^{\mathrm{c}}(T_m)=0$	∞	∞
system U	Cases	$R_{1,\mathrm{U}}^{(m)}$	$R_{2,\mathrm{U}}^{(m)}$
	a. $\mathcal{Q}_1^{\mathrm{p}}(T_m)>0, \mathcal{Q}_2^{\mathrm{p}}(T_m)>0$	$\min\{S_1^{(m)}, S_2^{(m)}\}$	$S_3^{(m)}$
	b. $\mathcal{Q}_1^{\mathrm{p}}(T_m)>0, \mathcal{Q}_2^{\mathrm{p}}(T_m)=0$	$\min\{S_1^{(m)}, S_2^{(m)}, S_3^{(m)}\}$	∞
	c. $\mathcal{Q}_1^{\mathrm{p}}(T_m)=0, \mathcal{Q}_2^{\mathrm{p}}(T_m)>0$	∞	$S_3^{(m)}$
	d. $\mathcal{Q}_1^{\mathrm{p}}(T_m)=0, \mathcal{Q}_2^{\mathrm{p}}(T_m)=0$	∞	∞

Table 3 Potential remaining service times at time T_m for system L and system U.

or equivalently, that

$$\begin{aligned}&\mathcal{Q}_{\Sigma}^{\mathrm{c}}(T_m) \le \mathcal{Q}_{\Sigma}^{\mathrm{p}}(T_m) \ \text{ and}\\ &\mathcal{Q}_1^{\mathrm{c}}(T_m) - \mathcal{Q}_1^{\mathrm{p}}(T_m) \le \mathcal{Q}_{\Sigma}^{\mathrm{p}}(T_m) - \mathcal{Q}_{\Sigma}^{\mathrm{c}}(T_m), \quad \text{w.p.1.}\end{aligned} \tag{16}$$

We prove the inequalities in (16) by induction. First, both inequalities in (16) hold at $T_0=0$ by the assumption that $\mathcal{Q}^{\mathrm{c}}(0)=\mathcal{Q}^{\mathrm{p}}(0)$, and we thus make the induction hypothesis that they hold at T_m for some $m\ge 0$. (For the remainder of the proof, all inequalities hold w.p.1, unless stated otherwise.)

$$\begin{aligned}&\textbf{(Induction hypothesis)}\ \mathcal{Q}_{\Sigma}^{\mathrm{c}}(T_m) \le \mathcal{Q}_{\Sigma}^{\mathrm{p}}(T_m) \ \text{ and}\\ &\mathcal{Q}_1^{\mathrm{c}}(T_m) - \mathcal{Q}_1^{\mathrm{p}}(T_m) \le \mathcal{Q}_{\Sigma}^{\mathrm{p}}(T_m) - \mathcal{Q}_{\Sigma}^{\mathrm{c}}(T_m).\end{aligned} \tag{17}$$

The goal is to show that the orders in (17) remain to hold at time T_{m+1}; in particular, to prove that

$$\begin{aligned}&\mathcal{Q}_{\Sigma}^{\mathrm{c}}(T_{m+1}) \le \mathcal{Q}_{\Sigma}^{\mathrm{p}}(T_{m+1}) \ \text{ and}\\ &\mathcal{Q}_1^{\mathrm{c}}(T_{m+1}) - \mathcal{Q}_1^{\mathrm{p}}(T_{m+1}) \le \mathcal{Q}_{\Sigma}^{\mathrm{p}}(T_{m+1}) - \mathcal{Q}_{\Sigma}^{\mathrm{c}}(T_{m+1}).\end{aligned} \tag{18}$$

Now, if the $(m+1)$th event is an arrival (to both systems), then (18) holds trivially given the induction hypothesis in (17), and it therefore suffices to consider service completions. Namely, it suffices to consider the case $A^{(m)} > S_i^{(m)}$, $i=1,\ 2,\ 3$. Since, as seen in Table 3, the state of $\mathcal{Q}^{\mathrm{c}}(T_m)$ belongs to one of the four cases (i)–(iv), and the state of $\mathcal{Q}^{\mathrm{p}}(T_m)$ belongs to one of the four cases (a)–(d), there are a total of 16 different cases to consider. In addition, there can be different random variables being the minimum of $S_1^{(m)}$, $S_2^{(m)}$, $S_3^{(m)}$ to assign the values of $R_{i,j}^{(m)}$, $i=1,\ 2$, $j=$ U, L, so that there are a total of 48 different cases to consider. We order the 48 different cases in tuples of the form (i,j,k), where $i\in$ {i,ii, iii, iv}, $j\in$ {a,b,c,d} and $k := \arg\min_{k=1,2,3}\{S_k^{(m)}\}$. For example, (i, a, 2) represents the case $\mathcal{Q}_1^{\mathrm{c}}(T_m)>0$ and $\mathcal{Q}_2^{\mathrm{c}}(T_m)>0$ in system L, $\mathcal{Q}_1^{\mathrm{p}}(T_m)>0$ and $\mathcal{Q}_2^{\mathrm{p}}(T_m)>0$ in system U, and $S_2^{(m)}$ is the minimum of $S_1^{(m)}$, $S_2^{(m)}$, $S_3^{(m)}$, so that $M^{(m)}=R_{2,\mathrm{L}}^{(m)}=R_{1,\mathrm{U}}^{(m)}$. In this case, the event at time T_{m+1} is a departure from station 2 in system L, and a departure from station 1 in system U.

We classify the 48 cases into three categories.

• Category 1: Cases that do not satisfy the induction hypothesis (17). This includes all cases in which system U is in one of the cases (i)-(iii), and system L is in case (d), so that $\mathcal{Q}_{\Sigma}^{\mathrm{c}}(T_m) > \mathcal{Q}_{\Sigma}^{\mathrm{p}}(T_m)=0$, and thus the first part of (17) is violated.

- Category 2: Cases that have the same event type in both systems at time T_{m+1}, i.e., an arrival to both systems or a departure from the same station in both systems simultaneously, so that the inequalities in (17) are trivially preserved, so that (18) holds. For example, in case (i, a, 1), we have $M^{(m)} = R_{1,\mathrm{L}}^{(m)} = R_{1,\mathrm{U}}^{(m)} = S_1^{(m)}$, so that the event at time T_{m+1} is a departure from station 1 in both systems U and L.
- Category 3: All remaining cases (that do not belong to either category 1 or 2).

We summarize all the cases in categories 1 and 2 in Table 4.

Category 1	(i, d, 1), (i, d, 2), (i, d, 3), (ii, d, 1), (ii, d, 2), (ii, d, 3), (iii, d, 1), (iii, d, 2), (iii, d, 3).
Category 2	(i, a, 1), (i, a, 3), (i, b, 1), (i, c, 3), (ii, a, 1), (ii, a, 2), (ii, b, 1), (ii, b, 2), (ii, b, 3), (iii, a, 3), (iii, c, 1), (iii, c, 3), (iv, c, 1), (iv, c, 2), (iv, d, 1), (iv, d, 2), (iv, d, 3).

Table 4 Cases that belong to Categories 1, 2 and do not need to be considered

We next prove that (18) holds for the cases in Category 3 by grouping those cases into 6 sets, according to the event at time T_{m+1}.

(1) **Cases (i, a, 2), (i, b, 2), (i, b, 3), (iii, a, 2), (iii, b, 2), (iii, b, 3).** The event at time T_{m+1} is a departure from station 2 in system L and a departure from station 1 in system U. Then

$$\mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}) = \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) - 1 \overset{\text{by (17)}}{\leq}$$
$$\mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - 1 < \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) = \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}),$$

so that the first inequality in (18) holds. Next,

$$\begin{aligned}
&\mathcal{Q}_1^{\mathrm{c}}(T_{m+1}) - \mathcal{Q}_1^{\mathbb{p}}(T_{m+1}) = \mathcal{Q}_1^{\mathrm{c}}(T_m) - (\mathcal{Q}_1^{\mathbb{p}}(T_m) - 1) \\
&= (\mathcal{Q}_1^{\mathrm{c}}(T_m) - \mathcal{Q}_1^{\mathbb{p}}(T_m)) + 1 \\
&\overset{\text{by (17)}}{\leq} (\mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m)) + 1 \\
&= \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - (\mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) - 1) \\
&= \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}) - \mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}),
\end{aligned}$$

proving the second inequality in (18).

(2) **Cases (i, c, 1), (ii, c, 1), (ii, c, 2), (iii, a, 1), (iii, b, 1).** The event at time T_{m+1} is a departure from station 1 in system L, so that the number of jobs in each system does not change. Then

$$\mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}) = \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) \overset{\text{by (17)}}{\leq} \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) = \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}),$$

which proves the first inequality in (18), and

$$\begin{aligned}
&\mathcal{Q}_1^{\mathrm{c}}(T_{m+1}) - \mathcal{Q}_1^{\mathbb{p}}(T_{m+1}) = (\mathcal{Q}_1^{\mathrm{c}}(T_m) - 1) - \mathcal{Q}_1^{\mathbb{p}}(T_m) \\
&\overset{\text{by (17)}}{\leq} \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) - 1 \\
&< \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) = \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}) - \mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}).
\end{aligned}$$

which proves that the second inequality in (18) holds as well.

(3) **Cases (i, c, 2), (iii, c, 2).** The event at time T_{m+1} is a departure from station 2 in system L. Hence

$$\begin{aligned}\mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}) &= \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) - 1 \\ &\overset{\text{by (17)}}{\leq} \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - 1 < \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) = \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}),\end{aligned}$$

proving the first inequality in (18). The second inequality in (18) holds as well, because

$$\begin{aligned}\mathcal{Q}_1^{\mathrm{c}}(T_{m+1}) - \mathcal{Q}_1^{\mathbb{p}}(T_{m+1}) &= \mathcal{Q}_1^{\mathrm{c}}(T_m) - \mathcal{Q}_1^{\mathrm{c}}(T_m) \\ &\overset{\text{by (17)}}{\leq} \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) \\ &< \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - (\mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) - 1) \\ &= \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}) - \mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}).\end{aligned}$$

(4) **Cases (ii, a, 3), (ii, c, 3).** The event at time T_{m+1} is a departure from station 1 in system L, and a departure from station 2 in system U, so that $\mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}) = \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m)$ and $\mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}) = \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - 1$. To prove that $\mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}) \leq \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1})$, we use the second inequality in the induction hypothesis (17), which is equivalent to

$$\mathcal{Q}_1^{\mathrm{c}}(T_m) + \frac{\mathcal{Q}_2^{\mathrm{c}}(T_m)}{2} \leq \mathcal{Q}_1^{\mathbb{p}}(T_m) + \frac{\mathcal{Q}_2^{\mathbb{p}}(T_m)}{2}. \tag{19}$$

In addition, for both cases (ii, a, 3) and (ii, c, 3), it holds that $\mathcal{Q}_2^{\mathrm{c}}(T_m) = 0$ and $\mathcal{Q}_2^{\mathbb{p}}(T_m) > 0$, so that

$$\begin{aligned}\mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) &= \mathcal{Q}_1^{\mathrm{c}}(T_m) + \frac{\mathcal{Q}_2^{\mathrm{c}}(T_m)}{2} \\ &\overset{\text{by (19)}}{\leq} \mathcal{Q}_1^{\mathbb{p}}(T_m) + \frac{\mathcal{Q}_2^{\mathbb{p}}(T_m)}{2} \\ &< \mathcal{Q}_1^{\mathbb{p}}(T_m) + \mathcal{Q}_2^{\mathbb{p}}(T_m) = \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m),\end{aligned} \tag{20}$$

where the first equality holds because $\mathcal{Q}_2^{p}(T_m) = 0$, and the last inequality is strict because $\mathcal{Q}_2^{q}(T_m) > 0$. Hence, $\mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) \leq \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - 1$, because the two queues are integer-valued, implying in turn that

$$\mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}) = \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) \leq \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - 1 = \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}),$$

so that the first inequality in (18) holds. Next,

$$\begin{aligned}\mathcal{Q}_1^{\mathrm{c}}(T_{m+1}) - \mathcal{Q}_1^{\mathbb{p}}(T_{m+1}) &= (\mathcal{Q}_1^{\mathrm{c}}(T_m) - 1) - \mathcal{Q}_1^{\mathbb{p}}(T_m) \\ &\overset{\text{by (17)}}{\leq} (\mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - 1) - \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) \\ &= \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}) - \mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}),\end{aligned}$$

which proves the second inequality in (18).

(5) **Cases (iv, a, 1), (iv, a, 2), (iv, b, 1), (iv, b, 2), (iv, b, 3).** The event at time T_{m+1} is a departure from station 1 in system U. The number of jobs in each system does not change, so the first inequality in (18) holds:

$$\mathcal{Q}_\Sigma^{\mathrm{c}}(T_{m+1}) = \mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) \overset{\text{by (17)}}{\leq} \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) = \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}).$$

For the second inequality in (18), note that in these cases,

$$\mathcal{Q}_\Sigma^{\mathrm{c}}(T_m) = 0 \quad \text{and} \quad \mathcal{Q}_1^{\mathbb{p}}(T_m) \geq 1, \tag{21}$$

and therefore

$$\begin{aligned}\mathcal{Q}_1^{\underline{c}}(T_{m+1}) - \mathcal{Q}_1^{\mathbb{p}}(T_{m+1}) &= \mathcal{Q}_1^{\underline{c}}(T_m) - (\mathcal{Q}_1^{\mathbb{p}}(T_m) - 1) \\ &\overset{\text{by (21)}}{\leq} 0 \overset{\text{by (17)}}{\leq} \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - \mathcal{Q}_\Sigma^{\underline{c}}(T_m) \\ &= \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}) - \mathcal{Q}_\Sigma^{\underline{c}}(T_{m+1}),\end{aligned}$$

proving the second inequality in (18).

(6) **Cases (iv, a, 3), (iv, c, 3).** The event at time T_{m+1} is a departure from station 2 in system U. In both cases, we have

$$\mathcal{Q}_\Sigma^{\underline{c}}(T_m) = 0 \quad \text{and} \quad \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - 1 \geq 0. \tag{22}$$

Therefore,

$$\begin{aligned}\mathcal{Q}_\Sigma^{\underline{c}}(T_{m+1}) &= \mathcal{Q}_\Sigma^{\underline{c}}(T_m) = 0 \\ &\leq \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - 1 = \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}),\end{aligned}$$

proving the first inequality in (18). For the second inequality in (18), we have

$$\begin{aligned}\mathcal{Q}_1^{\underline{c}}(T_{m+1}) - \mathcal{Q}_1^{\mathbb{p}}(T_{m+1}) &= \mathcal{Q}_1^{\underline{c}}(T_m) - \mathcal{Q}_1^{\mathbb{p}}(T_m) \\ &\overset{\text{by (22)}}{=} -\mathcal{Q}_1^{\mathbb{p}}(T_m) \leq 0 \\ &\overset{\text{by (22)}}{\leq} \mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - 1 = (\mathcal{Q}_\Sigma^{\mathbb{p}}(T_m) - 1) - \mathcal{Q}_\Sigma^{\underline{c}}(T_m) \\ &= \mathcal{Q}_\Sigma^{\mathbb{p}}(T_{m+1}) - \mathcal{Q}_\Sigma^{\underline{c}}(T_{m+1}).\end{aligned}$$

Thus far we have shown that $Q_\Sigma^{\underline{c}} \leq_{st} Q_\Sigma^{\mathbb{p}}$ and $2Q_1^{\underline{c}} + Q_2^{\underline{c}} \leq_{st} 2Q_1^{\mathbb{p}} + Q_2^{\mathbb{p}}$. The proof that the two inequalities $Q_\Sigma^{\mathbb{p}} \leq_{st} Q_\Sigma^{\bar{c}}$ and $2Q_1^{\mathbb{p}} + Q_2^{\mathbb{p}} \leq_{st} 2Q_1^{\bar{c}} + Q_2^{\bar{c}}$ hold can be proved by the similar arguments. □

A.6. Proof of Theorem 5

Proof. To show that the maximal stability region of Π is $[0, 1/2)$, we first note that any dynamic policy π satisfies $0 \leq \gamma_I^\pi(t)$ trivially. Therefore, taking $\underline{c} = 0$ in Theorem 4, we have

$$\mathcal{S}_\Pi(\pi) \subseteq [0, \frac{1}{2}), \ \forall \pi \in \Pi. \tag{23}$$

On the other hand, we construct a particular dynamic policy $\pi^{1/2} \in \Pi$ with $\gamma_I^{\pi^{1/2}}(t) \equiv 1/2, \ \forall t \geq 0$, i.e., $\pi^{1/2}$ is the policy that always allocates exactly $1/2$ of μ^*, at any given time t, to the server with the shortest queue. Taking $\underline{c} = \bar{c} = 1/2$ in Theorem 4, we have $[0, 1/2) \subseteq \mathcal{S}_\Pi(\pi^{1/2}) \subseteq [0, 1/2)$, or equivalently,

$$\mathcal{S}_\Pi(\pi^{1/2}) = \left[0, \frac{1}{2}\right). \tag{24}$$

Combining (23) and (24), we conclude that $\mathcal{S}^M(\Pi) = [0, 1/2)$.

Next, we show that any dynamic policy $\pi \in \Pi$ has $\mathcal{S}_\Pi(\pi) = \mathcal{S}^M(\Pi)$ if it satisfies $\gamma_I^\pi(t) \leq 1/2, \ \forall t \geq 0$. Since $0 \leq \gamma_I^\pi(t) \leq 1/2, \ \forall t \geq 0$, taking $\underline{c} = 0$ and $\bar{c} = 1/2$ in Theorem 4, we have

$$\begin{aligned}\left[0, \frac{1}{2} \wedge (1 - \bar{c})\right) &= \left[0, \frac{1}{2}\right) \subseteq \mathcal{S}_\Pi(\pi) \\ &\subseteq \left[0, \frac{1}{2} \wedge (1 - \underline{c})\right) = \left[0, \frac{1}{2}\right).\end{aligned}$$

Therefore, $\mathcal{S}_\Pi(\pi) = \mathcal{S}^M(\Pi) = [0, 1/2)$. In particular, policy π is a maximal dynamic policy. □

A.7. Proof of Proposition 4

Proof. Consider a static policy $\pi_{\tilde{\boldsymbol{d}}} \in \hat{\Pi}^{(n,k)}$ where $\tilde{d}_i = 1/k$ for $i \le k$, and $\tilde{d}_i = 0$ otherwise. Under policy $\pi_{\tilde{\boldsymbol{d}}}$, a job can only be completed via the completions of its k tasks by stations 1 to k. Therefore, the (n,k) system is equivalent to a (k,k) system with homogeneous servers, each with capacity μ^*/k. Since there is no redundancy in the (k,k) system, all tasks in a queue-buffer must be processed. In this case, each station is an $M/M/1$ queue with arrival rate λ and service rate μ^*/k. Then the (k,k) system is stable if and only if each station is stable, which in turn holds if and only if $\lambda < \mu^*/k$; equivalently, $\rho < 1/k$. It follows that

$$\mathcal{S}_{\Pi^{(n,k)}}(\pi_{\tilde{\boldsymbol{d}}}) = [0, 1/k) \subseteq \mathcal{S}^M(\hat{\Pi}^{(n,k)}) \subseteq \mathcal{S}^M(\Pi^{(n,k)}).$$

□

A.8. Proof of Proposition 3

Proof. For any given $\vec{\mu}$, we construct a maximal dynamic policy $\pi^* \in \Pi_{\vec{\mu}}$ as follows. At any given time $t \ge 0$, let the partition $\mathbf{P}$ of $[\![1, m]\!]$ be such that $\mathbf{P}_k = \emptyset$ for $k = I^{\pi^*}(t)$, and thus $\gamma_I^{\pi^*}(t) = \sum_{j \in \mathbf{P}_k} \mu_j/\mu^* = 0$. The allocation of capacity to the remaining stations can be arbitrary. Therefore, $\gamma_I^{\pi^*}(t) < 1/2$ for all $t \ge 0$, and by Theorem 5, π^* is a maximal dynamic policy. □

A.9. Proof of Theorem 2

The following lemma, whose proof appears in §A.9.1 below, is employed in the proof of Theorem 2.

LEMMA 6. *For a constant $c \in (1/2, 1]$, consider $\boldsymbol{Q}^c$ under the fixed-value policy $\mathbb{p}_c$. If $\rho \ge 1 - c$, then Q_2^c is not positive recurrent.*

Proof of Theorem 2. We prove from the result for the the equivalent TQ_2 system under policy $\mathbb{p}_{\boldsymbol{d}} \in \mathcal{P}$, induced by the static policy $\pi_{\boldsymbol{d}} \in \Pi$. First, consider the case $d_M < 1/2$, and recall that the capacity-allocation process $\zeta^{\mathbb{p}_{\boldsymbol{d}}}$ under policy $\mathbb{p}_{\boldsymbol{d}}$ satisfies $0 \le \zeta^{\mathbb{p}_{\boldsymbol{d}}}(t) \le d_M,\ \forall t \ge 0$. Taking $\underline{c} = 0$ and $\bar{c} = d_M$ in Lemma 2 gives

$$\begin{aligned}\left[0, \frac{1}{2} \wedge (1 - \bar{c})\right) &= \left[0, \frac{1}{2}\right) \subseteq \mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\boldsymbol{d}}) \\ &\subseteq \left[0, \frac{1}{2} \wedge (1 - \underline{c})\right) = \left[0, \frac{1}{2}\right).\end{aligned}$$

Hence, $\mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\boldsymbol{d}}) = [0, 1/2)$.

Next, consider the case $d_M > 1/2$. To show that the stability region is $\mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\boldsymbol{d}}) = [0, 1 - d_M)$, we prove that the condition $\rho < 1 - d_M$ is necessary and sufficient for the TQ_2 system under policy $\mathbb{p}_{\boldsymbol{d}}$ to be positive recurrent. To prove the necessity of the aforementioned condition, assume that $\rho \ge 1 - d_M$ and recall that under $\mathbb{p}_{\boldsymbol{d}}$, the control $\zeta^{\mathbb{p}_{\boldsymbol{d}}}(\tau_m)$ is set at d_i with probability d_i, and $\zeta^{\mathbb{p}_{\boldsymbol{d}}(t)}$ is fixed at d_i for any $t \in (\tau_m, \tau_{m+1})$, where τ_m is the beginning of the mth busy cycle of $Q_2^{\mathbb{p}_{\boldsymbol{d}}}$. Since $P(\zeta^{\mathbb{p}_{\boldsymbol{d}}}(\tau_m) = d_M) = d_M > 0$, the event $\{\zeta^{\mathbb{p}_{\boldsymbol{d}}}(\tau_m) = d_M \text{ for some } m \ge 0\}$ must occur. In turn, we have $\mathbb{E}[\tau_{m+1} - \tau_m | \tau_m < \infty, \zeta^{\mathbb{p}_{\boldsymbol{d}}}(\tau_m) = d_M] = \infty$ by Lemma 6, implying that $Q_2^{\mathbb{p}_{\boldsymbol{d}}}$ is not positive recurrent. The sufficiency of the condition $\rho < 1 - d_M$ for $\boldsymbol{Q}^{\mathbb{p}_{\boldsymbol{d}}}$ to be positive recurrent follows from Lemma 2, because $\zeta^{\mathbb{p}_{\boldsymbol{d}}}(t) \le d_M,\ \forall t \ge 0$.

Finally, since $\mathbb{p}_{\boldsymbol{d}}$ is induced by $\pi_{\boldsymbol{d}}$, it holds that $\mathcal{S}_{\Pi}(\pi_{\boldsymbol{d}}) = \mathcal{S}_{\mathcal{P}}(\mathbb{p}_{\boldsymbol{d}}) = \left[0, \frac{1}{2} \wedge (1 - d_M)\right)$ by Corollary 1. □

A.9.1. Proof of Lemma 6

Proof. We first claim that it suffices to consider $\rho = 1-c$. To show this, we temporarily add a superscript λ to the queue-length process, namely, $\boldsymbol{Q}^{c,\lambda} = (Q_1^{c,\lambda}, Q_2^{c,\lambda})$, and show that for two arrival rates $\lambda_1 < \lambda_2$ and a fixed total capacity μ^*, it holds that

$$Q_1^{c,\lambda_1} \leq_{st} Q_1^{c,\lambda_2} \quad \text{and} \quad Q_2^{c,\lambda_1} \leq_{st} Q_2^{c,\lambda_2} \tag{25}$$

whenever $\boldsymbol{Q}^{c,\lambda_1}(0) = \boldsymbol{Q}^{c,\lambda_2}(0)$. We prove (25) via a coupling argument.

Denote the system associated with process $\boldsymbol{Q}^{c,\lambda_i}$ by system i, $i = 1, 2$. We give system 2 a Poisson arrival process with rate λ_2, and at each arrival epoch to system 2, we give system 1 an arrival with probability λ_1/λ_2, so that system 1 is fed by a Poisson arrival process with rate λ_1. In particular, each arrival to system 1 is also an arrival to system 2. Let $\hat{T}_m$ denote the mth event time (arrival to, or departure from either system), and $\hat{T}_0 := 0$. At each event time $\hat{T}_m$, we use two independent exponential random variables $S_1^{(m)}$ with rate $c\mu^*$ and $S_2^{(m)}$ with rate $(1-c)\mu^*$ to generate the potential remaining service time at station i in system j, denoted by $R_{i,j}^{(m)}$, $i, j = 1, 2$, according to Table 5.

System 1	Cases	$R_{1,1}^{(m)}$	$R_{2,1}^{(m)}$
	i. $Q_1^{c,\lambda_1}(\hat{T}_m) > 0, Q_2^{c,\lambda_1}(\hat{T}_m) > 0$	$S_1^{(m)}$	$S_2^{(m)}$
	ii. $Q_1^{c,\lambda_1}(\hat{T}_m) > 0, Q_2^{c,\lambda_1}(\hat{T}_m) = 0$	$\min\{S_1^{(m)}, S_2^{(m)}\}$	∞
	iii. $Q_1^{c,\lambda_1}(\hat{T}_m) = 0, Q_2^{c,\lambda_1}(\hat{T}_m) > 0$	∞	$S_2^{(m)}$
	iv. $Q_1^{c,\lambda_1}(\hat{T}_m) = 0, Q_2^{c,\lambda_1}(\hat{T}_m) = 0$	∞	∞
System 2	Cases	$R_{1,2}^{(m)}$	$R_{2,2}^{(m)}$
	a. $Q_1^{c,\lambda_2}(\hat{T}_m) > 0, Q_2^{c,\lambda_2}(\hat{T}_m) > 0$	$S_1^{(m)}$	$S_2^{(m)}$
	b. $Q_1^{c,\lambda_2}(\hat{T}_m) > 0, Q_2^{c,\lambda_2}(\hat{T}_m) = 0$	$\min\{S_1^{(m)}, S_2^{(m)}\}$	∞
	c. $Q_1^{c,\lambda_2}(\hat{T}_m) = 0, Q_2^{c,\lambda_2}(\hat{T}_m) > 0$	∞	$S_2^{(m)}$
	d. $Q_1^{c,\lambda_2}(\hat{T}_m) = 0, Q_2^{c,\lambda_2}(\hat{T}_m) = 0$	∞	∞

Table 5 **Realization of potential remaining service times at time $\hat{T}_m$ for system 1 and system 2.**

From the above construction, it is easy to see that, if $\boldsymbol{Q}^{c,\lambda_1}(\hat{T}_m) \leq \boldsymbol{Q}^{c,\lambda_2}(\hat{T}_m)$ w.p.1 for $m \geq 0$ (where the inequality holds componentwise), then $\boldsymbol{Q}^{c,\lambda_1}(t) \leq \boldsymbol{Q}^{c,\lambda_2}(t)$ w.p.1 for all $t \in [\hat{T}_m, \hat{T}_{m+1}]$, and thus, by induction, $\boldsymbol{Q}^{c,\lambda_1}(t) \leq \boldsymbol{Q}^{c,\lambda_2}(t)$ w.p.1 for all $t \geq 0$. This proves the stochastic orders in (25).

Now, since $Q_2^{c,\lambda}$ is nondecreasing in λ by (25), it suffices to show that $Q_2^{c,\lambda}$ is not positive recurrent when $\lambda = (1-c)\mu^*$. To this end, we first show that Q_1^c is positive recurrent for $\lambda = (1-c)\mu^*$. Note that Q_1^c has an arrival rate $\lambda = (1-c)\mu^*$, and service rate that is bounded from below by $c\mu^*$. Let Z be the number-in-system process of an $M/M/1$ queue with arrival rate $(1-c)\mu^*$ and service rate $c\mu^*$. By the sample-path stochastic order of state-dependent birth-and-death processes (Whitt 1991, Lemma 1), it holds that $Q_1^c \leq_{st} Z$. Because of the assumption that $c > 1/2$ (so that $(1-c)\mu^* < c\mu^*$), Z is positive recurrent, and so Q_1^c is positive recurrent as well.

For a positive recurrent Q_1^c, the long-run average arrival rate to station 1 equals the long-run average departure rate from station 1 (see, e.g., (El-Taha et al. 1999, Lemma 1.5)), which is also the long-run average arrival rate to station 2. Thus, the long-run average arrival rate to station 2 is $(1-c)\mu^*$. Since Q_2^c has an invariant service rate $(1-c)\mu^*$, we conclude that Q_2^c is not positive recurrent under the condition $\rho = 1-c$, and by (25), it is not positive recurrent for any $\rho \geq 1-c$. □