Queueing Syst (2017) 85:149–172
DOI 10.1007/s11134-016-9507-9

Optimal control of a single server in a finite-population queueing network

Nilay Tanık Argon[1] · Chao Deng[1] · Vidyadhar G. Kulkarni[1]

Received: 4 September 2015 / Revised: 17 October 2016 / Published online: 16 November 2016
© Springer Science+Business Media New York 2016

Abstract We study the optimal dynamic assignment of a single server to multiple stations in a finite-population queueing network. The objective is to maximize the long-run average reward/throughput. We use sample-path comparisons to identify conditions on the network structure and service time distributions under which the optimal policy is an index policy. This index policy assigns the server to the non-empty station where it takes the shortest amount of time (in some stochastic sense) to complete a job. For example, in a network of multiple parallel stations, the optimal policy assigns the highest priority to the fastest station if service times can be ordered in likelihood ratios. Finally, by means of a numerical study, we test the shortest-expected-remaining-service-time policy on parallel-series networks with three stations and find that this index policy either coincides with the optimal policy or provides a near-optimal performance.

Keywords Scheduling · Queueing networks · Likelihood ratio ordering · SEPT

Mathematics Subject Classification 68M20 · 60K25 · 90B22 · 90B36

✉ Nilay Tanık Argon
nilay@unc.edu

Chao Deng
chaodeng@live.unc.edu

Vidyadhar G. Kulkarni
vkulkarn@email.unc.edu

1 Department of Statistics and Operations Research, University of North Carolina, Chapel Hill, NC 27599, USA

1 Introduction

This paper is concerned with queueing systems with a finite source of customers that cannot be approximated by infinite-population queueing models. For such a system, we are particularly interested in dynamic prioritization of customers for service based on information on their service requirements. Such finite-population systems emerge in various applications. One of the most important applications is the machine interference/repair problem. In this problem, machines (customers) operate for a random period of time, then break down and request service from a repair person (server). Because the repair persons are limited in numbers, the service manager may face two operational decisions during the repair process: which repair person should serve a machine and in what order the machines should be served. There is a rich literature that studies the machine interference problem that is summarized in two survey articles by Stecke and Aronson [18] and Haque and Armstrong [4].

Another application area for finite-population queueing systems is computer communication. In a computer-communication system, multiple terminals (customers) request service from a computer or a peripheral unit (server). Each terminal generates requests after a random amount of time and the server works on each request based on a queueing policy. This queueing policy determines the throughput rate, i.e., the number of requests completed per unit time, which is one of the most important performance measures showing the system's processing power. For examples of work on improving computer systems performance, see Dshalalow [3], Haverkort [5], King [9], and Takagi [20].

Finite-population queueing models can be also useful in other applications such as healthcare, where the utilization of certain resources such as intensive care unit beds is high. For example, in a hospital unit, nurses have many tasks that require their attention and it is not clear which one of these tasks they should prioritize. More specifically, during a patient's stay at the hospital, he/she may have nursing requests such as pain management in addition to admission and discharge procedures that all require attention from a nurse. Modeling such a system as a finite-population queueing system where customers and servers represent patients and nurses, respectively, could help identify effective nurse assignment policies that maximize the throughput of the hospital unit. See de Véricourt and Jennings [2] for an example of a finite-population queueing model that is used to determine optimal nurse-to-patient ratios.

In this work, we consider a queueing system in which requests for service are generated by a finite number of customers and are handled by a single server. A customer can be in three states: inactive, waiting for service, or in service. An inactive customer generates a request for service after a random amount of time independently from the other customers. If the server is busy at the time of this request, the customer needs to join a queue and wait for its turn for service. Once the service is completed, the customer becomes inactive again, and the process repeats. Each time the service of a customer is completed, a reward is earned. The server can handle only one request at a time, and when there are multiple requests, the question is which customer should be prioritized for service so that the long-run average reward is maximized.

Note that such a finite-population service system can be modeled as a closed queueing network with a constant number of customers and two stages of service; see Fig. 1.

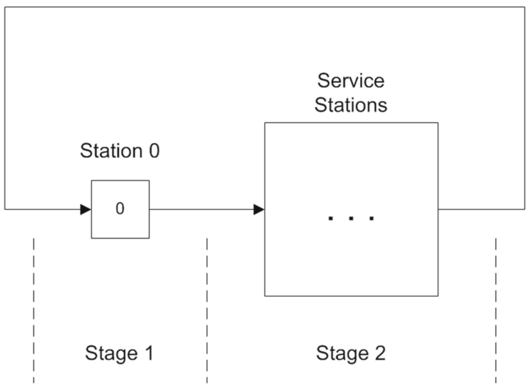

Fig. 1 A closed queueing system with two stages of stations

The first stage, which is labeled as station 0 in the network, represents the place where the customers spend their inactive time. The customers do not need any servers during their sojourn at station 0 and hence there is no queue at this station. The second stage consists of multiple "service" stations and a single flexible server who is able to work at any one of the service stations. After completion of service at the second stage, customers are rerouted to the first stage and the process repeats in this manner. We are interested in dynamically allocating the single server to each of the service stations so that the long-run average reward (or the weighted throughput) of the system is maximized.

In the remainder of the paper, we first provide a literature review of the relevant work in Sect. 2, followed by the model description and problem formulation presented in Sect. 3. Using sample-path arguments, we provide sufficient conditions on the network structure and service time distributions for the optimality of some form of the *shortest-remaining-service-time* policy within the set of preemptive and non-preemptive policies in Sects. 4 and 5, respectively. In Sect. 6, by means of a numerical study, we test the performance of the *shortest-expected-remaining-service-time heuristic* for finite-population queueing systems that do not satisfy the conditions identified by our analytic results. Finally, in Sect. 7, we provide our concluding remarks. Proofs of our theoretical results are provided in the Appendix.

2 Literature review

We start by reviewing the most relevant work on finite-population queueing systems and refer the interested reader to Sztrik [19] for an excellent bibliography of articles on these queueing systems and their applications. Palesano and Chandra [13] study a machine interference problem with multiple types of failures. The authors conduct a numerical study to compare the system performance under different server assignment policies. One of their main findings is that the mean number of machines waiting for repair increases when the machines that require the longest mean service time are prioritized. Consistent with this numerical observation, we prove a result that shows that the optimal policy that maximizes the long-run average throughput gives priority to the customers with the shortest service times in some stochastic sense. Another relevant line of work that studies machine interference problem is by Iravani and Kolfal [7] and Iravani et al. [8]. Both papers study the problem with heterogenous machines that have distinct up and repair time distributions as well as different downtime costs. Their objective is to find dynamic server allocation policies to these heterogenous machines so as to minimize the long-run average cost. Iravani and Kolfal [7] find the conditions under which certain static priority rules are optimal within the class of preemptive policies. Iravani et al. [8] show that the optimal policy may never serve some classes of machines if preemption is not allowed; for those classes that are served, a static priority policy is optimal. Note that the queueing model studied in these two papers differs from our model, in that their model allows for heterogenous jobs (leading to multiple closed classes of jobs in the network that cannot change class) but each class of job requesting only one type of service, whereas our model allows for multiple types of services but for a single class of jobs.

There is also a vast literature on queueing networks with flexible servers where the servers are capable of working at multiple stations. In the remainder of this section, we review the most relevant work from this literature.

Both Righter and Shanthikumar [15] and Ahn and Righter [1] study the problem of maximizing the job completion process in queueing networks with flexible servers. [15] considers queueing networks that involve "controllable" stations each having a single server and increasing likelihood ratio service distributions. The authors show that any non-preemptive policy at the controllable stations with a single class of customers stochastically maximizes the joint departure process. [1] considers a tandem queueing network where servers are trained to do a subset of consecutive tasks. The authors prove that the *last-buffer-first-served policy* for each server stochastically maximizes the job completion process in open or closed tandem networks under certain conditions on service time distributions. These results in [1] and [15] have important implications for certain specialized cases of our problem, and hence we discuss them in more detail in Sects. 4 and 5.

Another relevant work is by Koole and Righter [10], who study open and closed tandem networks, where the stations are divided into several non-overlapping sets of adjacent stations and each server is able to work on only one set of stations. The authors show that the optimal policy assigns each server to work at its last nonempty station, which is consistent with one of the results proved in this paper. However, in the network studied by Koole and Righter [10], there is no stage 1, where customers can be in an inactive state, which is the case in our model.

Van Oyen et al. [21] study a serial production system with multiple flexible servers. When servers are collaborative, i.e., when servers are able to work together on the same job at the same time, the authors show that the expedite policy is optimal to minimize the cycle time for each job. (The expedite policy pools all servers into a team and assigns the team to a job to work it through from the first station to the last station without any interruptions.) The authors also extend the optimality of the expedite policy to a closed queueing network and prove that the policy is optimal with respect to throughput.

Finally, Hopp et al. [6] study a closed tandem queueing network with a single server and a mix of manual and automated stations. The first station is an automated station, which processes jobs automatically without the need for a server but requires the server for loading. The other stations are manual stations that cannot process any jobs unless attended by the server. For the case with three stations, the authors show that the optimal control policy that maximizes the long-run average throughput is a static policy that gives priority to keep the automated station busy for as long as possible. This structure for the optimal policy is in line with the structure that we identify in this paper but under different modeling assumptions.

3 Problem formulation

In this paper we focus on a special class of finite-population queueing systems with B customers ($1 < B < \infty$), which we call a *parallel-series* system, as shown in Fig. 2. (We interchangeably use the words "customer" and "job" throughout this paper.) The second stage of such a system consists of $J \geq 1$ parallel branches, where the jth branch ($j = 1, 2, \ldots, J$) consists of $i_j \geq 1$ service stations in series. We label the ith station in the jth branch as station (i, j). A customer stays in station 0 for a random amount of time and then moves to station $(1, j)$ with probability $p_j > 0\,(1 \leq j \leq J)$, where $\sum_{j=1}^{J} p_j = 1$. A customer stays in station (i, j) until the server completes the associated task, then moves to station $(i+1, j)$ if $i < i_j$, and to station 0 if $i = i_j$. This process repeats in the same manner forever. We let $S_{i,j}$ be the generic random variable that represents the service time in station (i, j) $(1 \leq i \leq i_j, 1 \leq j \leq J)$. We assume that all service times at the service stations and sojourn times at station 0 are independent of each other.

A parallel-series system generalizes several special systems that arise in real-life applications. For example, a system with $J = 1$ represents a tandem queueing network that is observed commonly in manufacturing systems. On the other hand, a system with $i_j = 1$ for all $j = 1, 2, \ldots, J$ represents a parallel queueing network with J service stations. Such a parallel system arises where the service needs can be classified into J types as in the case of the machine repair problem with multiple types of failures.

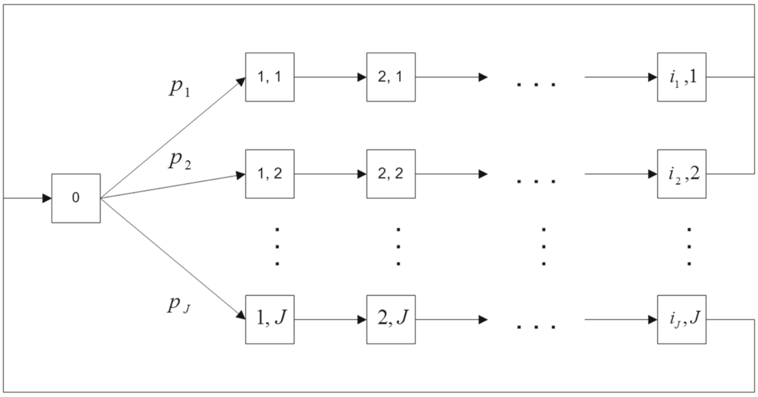

Fig. 2 A finite-population queuing system with parallel-series structure

We next define further notation to formally state our optimization problem and also simplify exposition in the rest of the paper. Let K be the total number of stations in the second stage of a parallel-series system, i.e., $K = \sum_{j=1}^{J} i_j$. Let also $D_0^\pi(t)$ and $D_{i,j}^\pi(t)$ denote the number of service completions in station 0 and in station (i, j) $(1 \le i \le i_j,\ 1 \le j \le J)$, respectively, during $[0, t]$ under policy π. (A policy determines the priority decisions that the server should make, possibly dynamically with time and state.) Suppose that a positive and finite reward $r_{i,j}$ is gained when service is completed in station (i, j) $(1 \le i \le i_j, 1 \le j \le J)$. We define

$$R^\pi \equiv \liminf_{t\to\infty} \sum_{j=1}^{J} \sum_{i=1}^{i_j} r_{i,j} \frac{\mathrm{E}\left[D_{i,j}^\pi(t)\right]}{t} \tag{1}$$

to be the long-run average reward of the system under policy π when the limit exists.

Let Π_P and Π_NP denote the set of preemptive-resume and non-preemptive policies, respectively. For preemptive-resume policies, the server is allowed to make decisions at any given point in time and the service of a job that is preempted will resume from where it was left off the next time the server works on it. Under non-preemptive policies, the server is allowed to switch to work at other stations only when it completes service at the current station. In this paper we consider both preemptive-resume and non-preemptive policies.

Let T^π denote the long-run average throughput at station 0 under policy π, i.e.,

$$T^\pi \equiv \liminf_{t\to\infty} \frac{\mathrm{E}\left[D_0^\pi(t)\right]}{t}. \tag{2}$$

Throughout the paper, we will refer to T^π as the system throughput. Our first result establishes a relation between (1) and (2).

Proposition 1 *Under a given policy $\pi \in \Pi_\mathrm{P} \cup \Pi_\mathrm{NP}$, we have*

$$R^\pi = T^\pi \sum_{j=1}^{J} \sum_{i=1}^{i_j} p_j r_{i,j}. \tag{3}$$

Based on Proposition 1 and the fact that the double sum in Eq. (3) is a finite constant independent of π, our objective reduces to maximizing T^π within Π_P and Π_NP.

Before we continue with our analysis, we provide some definitions that we use frequently in this paper. Let X and Y be two continuous [or discrete] random variables with densities [or probability mass functions] $h(t)$ and $g(t)$, respectively. If $\Pr\{X > x\} \le \Pr\{Y > x\}$ for all real x, then X is said to be smaller than Y in the usual stochastic order (denoted by $X \le_\mathrm{st} Y$). If $h(x)g(y) \ge h(y)g(x)$ for all $x \le y$, then X is said to be smaller than Y in the likelihood ratio order (denoted by $X \le_\mathrm{lr} Y$). Also, a random variable X is said to be increasing in likelihood ratio (ILR) if $[X - t|X \ge t] \le_\mathrm{lr} [X - s|X \ge s]$ for $s \le t$. Finally, we define an ILR family of distributions as a set of distributions that are all ILR and that have non-overlapping hazard rates, i.e., if r_k and

r_l are the hazard rate functions of two different distributions of the family, it must be that $r_k(s) \geq r_l(t)$ for all $s, t \geq 0$, or vice versa. For more on these stochastic orders and the ILR family of distributions, see, for example, Shaked and Shanthikumar [17] and Righter [14], respectively.

In Sects. 4 and 5 we present analytic results on the optimal policies within the sets of preemptive and non-preemptive policies, respectively. More specifically, we provide sufficient conditions under which certain state-independent policies maximize $D_0^\pi(t)$ in the usual stochastic order for all $t \geq 0$, and hence the long-run average throughput whenever the limit in (2) exists.

4 Preemptive policies

In this section we study the dynamic control problem under preemptive-resume policies. By definition, in a preemptive policy, the server can change its decision at any point in time. These decisions are either to idle or to work on a job at one of the service stations. For this problem, the state of the system can be defined by the current station for each job and the amount of service (in time) attained by that job at its current station.

We first present results that are already proved in the literature for two special network structures. First, consider a series system as shown in Fig. 3, where $J = 1$. For notational convenience, we relabel the service station $(k, 1)$ as station k, for $k = 1, \ldots, K$.

Proposition 2 (Ahn and Righter [1]) *For a series system with ILR service times at each service station, the non-idling policy that gives priority to the job with the most service attained at the non-empty station with the largest index maximizes $D_0^\pi(t)$ in the usual stochastic order for all $t \geq 0$ within the set of all preemptive-resume policies Π_P.*

Proposition 2 implies that the server should prioritize the job that is closest to the entry into station 0 in order to maximize the number of departures from station 0, and hence the throughput. The intuition is that the earlier a job goes back to station 0, the earlier this job leaves station 0 to request service, which increases the utilization of the server and hence the throughput of the system. Note that, after clearing any partially completed jobs that are present at the service stations initially, the optimal policy turns into a policy under which the server picks a job from the queue in front of station 1 and completes the service of this job at all service stations $1, 2, \ldots, K$ sequentially before it starts working on another job waiting in front of station 1. Such

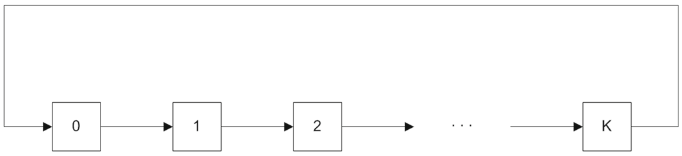

Fig. 3 A finite-population queueing system with K service stations in series

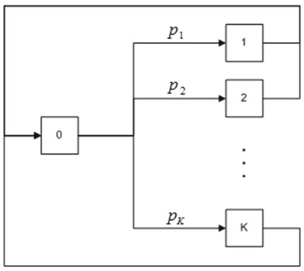

Fig. 4 A finite-population queueing system with K parallel service stations

a policy is sometimes referred to as a *sequential* policy or a *pick-and-run* policy. It is also important to notice that the optimal policy is a non-preemptive one even though preemption is allowed.

Next, consider a parallel system as shown in Fig. 4, where $i_j = 1$ for each $j = 1, \ldots, J$ and hence $K = J$. For notational convenience, we relabel the service station $(1, k)$ as station k, for $k = 1, \ldots, K$.

Proposition 3 (Righter and Shanthikumar [15]; Righter [14]) *For a parallel system with processing times at service stations belonging to an ILR family of distributions, the policy that gives priority to the job with the shortest expected remaining service time maximizes $D_0^\pi(t)$ in the usual stochastic order for all $t \geq 0$ within the set of all preemptive-resume policies Π_P.*

Proposition 3 follows the same intuition as for the series system, where the optimal policy pushes jobs towards station 0 as soon as possible, but this time in a parallel network. Also, unlike for the series system, the optimal policy for the parallel system can be preemptive because the arrival of a “faster” job from station 0 may cause the server to stop working on the current job and switch to serving this new arrival. Righter [14] proves a similar result (Theorem 13.D.8) for a clearing system with no arrivals but, as the author notes, the proof can be easily extended to the case with arrivals (in an open or closed network setting) as is also shown by Righter and Shanthikumar [15].

We next provide a result for a general parallel-series system but with the condition that service times at each station are independent and identically distributed (i.i.d.) and ILR.

Proposition 4 *For a parallel-series system with i.i.d. and ILR service times at all service stations, the non-idling policy that always assigns the server to the job with the fewest number of tasks remaining and, in case of a tie, to the job with the most service attained at its current station, maximizes $D_0^\pi(t)$ in the usual stochastic order for all $t \geq 0$ within Π_P.*

Proposition 4 implies that if the total service required on each branch of the network can be split into multiple tasks that take an i.i.d. time that is also ILR, then the job that is closest to enter station 0 should be prioritized in order to maximize the total number of departures and thus the long-run average throughput. For example, when the total service time of a job at each branch of the network is deterministic, then Proposition 4 gives the classical result that the optimal policy prioritizes the job with the shortest remaining service time at any point in time; see Schrage [16].

5 Non-preemptive policies

In this section we consider the optimal dynamic assignment of a single server to K service stations under non-preemptive policies. Due to its non-preemptive nature, in this control problem the decision epochs are completion time of service at a service station and arrival time of a job from station 0 when the server is idle. Furthermore, the state definition for the problem under preemptive policies needs to be expanded to include the job that the server is currently working on.

As in Sect. 4, we next present several results that characterize the policy that stochastically maximizes the total number of departures from station 0 at any point in time. For convenience, we assume that no job present at a service station at time zero has received earlier service at that station. Note that results in this section can be easily extended to the case with more general initial conditions with partially completed jobs at service stations under conditions such as service times having an ILR distribution.

We first provide a result that completely characterizes an optimal policy for a series system.

Proposition 5 *For a series system, the non-idling policy that gives priority to the non-empty station with the largest index within* Π_{NP} *maximizes* $D_0^{\pi}(t)$ *in the usual stochastic order for all* $t \geq 0$.

Proposition 5 is consistent with what we observed in the case with preemptive service: serving the job that is closest to joining station 0 is optimal for the series system. Note that the optimal policy for the series system characterized in Proposition 2 within the set of preemptive-resume policies is actually non-preemptive. Hence, Proposition 2 implies that the pick-and-run policy is optimal within Π_{NP} if the service times are ILR. Strengthening this result, Proposition 5 shows that pick-and-run is also optimal within Π_{NP} for a series system even when service times are not ILR. A similar result was provided by Ahn and Righter [1] under the condition that service times at all stations are identically distributed but under the objective of stochastically maximizing the joint task completion process from all stations, which is a stronger performance measure.

Having characterized the optimal policy for the series system, in the remainder of this section, we focus on parallel-series networks with at least two branches.

Proposition 6 *Suppose that there exists a branch* $j^* \in \{1, 2, \ldots, J\}$ *in a parallel-series system with* $J \geq 2$ *such that* $S_{i_{j^*}, j^*} \leq_{\mathrm{lr}} S_{i_j, j}$*, for all* $j \in \{1, 2, \ldots, J\} \backslash \{j^*\}$*. Within* Π_{NP}*, it suffices to consider policies that do not idle the server whenever there is a job at station* (i_{j^*}, j^*) *and give priority to that job in order to maximize* $D_0^{\pi}(t)$ *in the usual stochastic order for all* $t \geq 0$.

Proposition 6 implies that whenever there is a job at the “end station” that has the shortest service time in likelihood ratio ordering among all end stations, the server should not idle but serve that job. We next use Proposition 6 to develop an algorithm (Algorithm A-1) that yields a partial characterization for optimal policies when $J \geq 2$. Corollary 1 proves that this algorithm works.

Algorithm A-1 Algorithm for obtaining a partial characterization of optimal policies for $J \geq 2$.

S.1 Let ℓ be the iteration number and set it to zero. Let also $H = 0$. Refer to the original parallel-series system as Network 0. Let $S_{i,j}^{(\ell)}$ be the random variable that denotes the i.i.d. service time at node (i, j) in Network ℓ and let $i_j(\ell)$ be the number of stations in the jth branch of Network ℓ, where $j = 1, 2, \ldots, J$ and $\ell = 0, 1, 2, \ldots$. Let $i_j(0) = i_j$ for all $j = 1, 2, \ldots, J$ and $S_{i,j}^{(0)} = S_{i,j}$ for all $j = 1, 2, \ldots, J$ and $i = 1, 2, \ldots, i_j$.

S.2 In Network ℓ, if there exists a branch j^* such that $S_{i_{j^*}(\ell),j^*}^{(\ell)} \leq_{\text{lr}} S_{i_j(\ell),j}^{(\ell)}$ for all $j \in \{1, \ldots, J\}\backslash\{j^*\}$, then go to **S.3**; otherwise go to **S.4**.

S.3 (a) If $i_{j^*}(\ell) > 1$, then obtain a new network, namely Network $\ell+1$, by pooling stations $(i_{j^*}(\ell)-1, j^*)$ and $(i_{j^*}(\ell), j^*)$ in Network ℓ and keeping all other stations intact. Let the pooled station in Network $\ell+1$ be labeled as $(i_{j^*}(\ell)-1, j^*)$ and let the labels of all the remaining nodes be the same as in Network ℓ. Then, let $i_{j^*}(\ell+1) = i_{j^*}(\ell) - 1$, $i_j(\ell+1) = i_j(\ell)$ for $j \in \{1, 2, \ldots, J\}\backslash\{j^*\}$, $S_{i_{j^*}(\ell+1),j^*}^{(\ell+1)} = S_{i_{j^*}(\ell)-1,j^*}^{(\ell)} + S_{i_{j^*}(\ell),j^*}^{(\ell)}$ and $S_{i,j}^{(\ell+1)} = S_{i,j}^{(\ell)}$ for all other stations in Network $\ell + 1$. Finally, increment ℓ by one, and go to **S.2**.

(b) Otherwise (if $i_{j^*}(\ell) = 1$), increment ℓ by one, let $H = j^*$, and go to **S.4**.

S.4 Stop.

Corollary 1 *For $J \geq 2$, if Algorithm A-1 stops at iteration $\bar{\ell} \geq 1$, then in order to maximize $D_0^\pi(t)$ in the usual stochastic order for all $t \geq 0$, it is sufficient to consider policies in the set $\Gamma \subset \Pi_{\text{NP}}$, which is defined as follows: Under any policy $\gamma \in \Gamma$,*

(a) *if $i_j(\bar{\ell}) \neq i_j(0)$ for some branch $j \in \{1, 2, \ldots, J\}$, then once the server starts working on a job at one of stations $(i_j(\bar{\ell}), j), (i_j(\bar{\ell}) + 1, j), \ldots, (i_j(0), j)$, the server keeps working on that job without any idling until the job enters station 0; and*

(b) *if Algorithm A-1 stops with $H > 0$, then branch H receives the highest priority for service.*

In Sects. 5.1 and 5.2 we consider two special networks to demonstrate how Algorithm A-1 can be employed to characterize optimal policies. In Sect. 5.3 we obtain more characterizations for the optimal policy in a parallel-series network with i.i.d. service times at all service stations.

5.1 Parallel system

Consider the parallel system shown in Fig. 4 under the non-preemptive service assumption. For notational convenience, let X_k be the random variable that denotes the i.i.d. service time at station k (instead of $S_{1,k}$) for $k = 1, 2, \ldots, K$. Corollary 1 leads to the following partial characterization of the optimal policy for this system:

Corollary 2 *Suppose that in a parallel system there exists a station $i \in \{1, 2, \ldots, K\}$ for which $X_i \leq_{\text{lr}} X_j$, for all $j \in \{1, 2, \ldots, K\}\backslash\{i\}$. Then, within Π_{NP}, it is sufficient to consider policies that do not idle the server whenever there is a job at station i and that give priority to this job in order to maximize $D_0^\pi(t)$ in the usual stochastic order for all $t \geq 0$.*

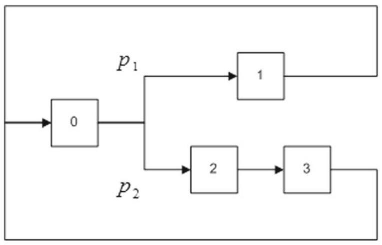

Fig. 5 A finite-population queueing system with three service stations on two parallel branches

Corollary 2 says that whenever there is a job at the service station that has the shortest service time in likelihood ratio ordering among all service stations, then the server should not idle and take that job into service. For a parallel network with two branches, this result completely characterizes the optimal policy if the service times at the two service stations can be ordered according to the likelihood ratio ordering. For networks with more than two branches, Corollary 2 only gives a partial characterization of the optimal policy. For such a network, serving the job with a shorter service time (in the likelihood ratio ordering) does not necessarily maximize the total number of departures from station 0 in the usual stochastic ordering; see, for example, Example 1 in the Appendix. However, a numerical study (presented in Sect. 6) for a parallel system with exponential service times and three service stations supports the conjecture that the shortest-expected-service-time policy maximizes the long-run average throughput among all non-preemptive and non-idling policies.

5.2 A network with two branches and three service stations

Consider a parallel-series system with two branches and three service stations, i.e., $J = 2$, $i_1 = 1$, and $i_2 = 2$, and hence $K = 3$. As shown in Fig. 5, we relabel stations (1, 1), (1, 2), and (2, 2) as stations 1, 2, and 3, respectively, for notational convenience. This queueing system is motivated by the nurse assignment problem discussed in Sect. 1. In particular, this closed queueing system can be used to model a hospital ward where each patient can be in four different states: She seeks admission service from a nurse (station 3); she stays at a bed not requiring any service from a nurse (station 0); she requests nursing service (station 1), which may repeat several times, and finally she seeks discharge service from a nurse (station 2). We assume that a new patient is ready to be admitted to the ward immediately after a patient is discharged. We define X_k to be the random variable denoting the i.i.d. service time at station k for $k = 1, 2, 3$.

Corollary 1 yields the following characterizations for the optimal policy:

Corollary 3 *When the objective is to maximize $D_0^{\pi}(t)$ in the usual stochastic order over $\pi \in \Pi_{\mathrm{NP}}$ for all $t \geq 0$ for the parallel-series system with two branches and three service stations shown in Fig. 5,*

(a) *it suffices to consider only policies that do not idle the server whenever there is a job at station* 1[3] *and that give priority to that job if $X_1[X_3] \leq_{\mathrm{lr}} X_3[X_1]$;*

(b) *there exists a non-idling optimal policy that gives priority to stations* 3, 1, *and* 2 *in that order if* $X_3 \leq_{\mathrm{lr}} X_1 \leq_{\mathrm{lr}} X_2 + X_3$*; and*
(c) *there exists a non-idling optimal policy that gives priority to stations* 3, 2, *and* 1 *in that order if* $X_3 \leq_{\mathrm{lr}} X_1$ *and* $X_2 + X_3 \leq_{\mathrm{lr}} X_1$.

Corollary 3(a) provides partial characterizations of the optimal policy where either one of the two end stations (stations 1 or 3) should be prioritized at all times. On the other hand, parts (b) and (c) of Corollary 3 provide conditions under which the optimal policy is completely characterized. More specifically, if the conditions in part (b) of Corollary 3 are satisfied, then all jobs that are initially present at station 3 should be given priority, and then the tasks at stations 2 and 3 should be pooled for all other jobs, after which the priority is given to jobs at station 1 over the jobs at the pooled station. If the conditions in part (c) of Corollary 3 are satisfied, then all jobs that are initially present at station 3 should be given priority, and then the tasks at stations 2 and 3 should be pooled for all other jobs, after which the priority is given to jobs at this pooled station over the jobs at station 1.

In the remainder of this section, we provide distributional examples where the conditions given in Corollary 3 are satisfied. In Proposition 7, we consider the case where the service times are uniformly distributed at each service station.

Proposition 7 *Suppose that* X_i *is uniformly distributed over* $[a_i, b_i]$ *for* $i = 1, 2, 3$*, and* X_2 *and* X_3 *are independent.*

(a) *If* $a_1 \leq a_3$ *and* $b_1 \leq b_3$*, then* $X_1 \leq_{\mathrm{lr}} X_3$.
(b) *If* $a_3 \leq a_1 \leq a_2 + a_3$ *and* $b_3 \leq b_1 \leq b_2 + b_3 - \min\{b_2 - a_2, b_3 - a_3\}$*, then* $X_3 \leq_{\mathrm{lr}} X_1 \leq_{\mathrm{lr}} X_2 + X_3$.
(c) *If* $a_1 \geq a_2 + a_3 + \min\{b_2 - a_2, b_3 - a_3\}$ *and* $b_1 \geq b_2 + b_3$*, then* $X_3 \leq_{\mathrm{lr}} X_1$ *and* $X_2 + X_3 \leq_{\mathrm{lr}} X_1$.

Proposition 7, which is proved in the Appendix, can be used together with Corollary 3 to obtain the optimal control policy when service times are uniformly distributed. For example, when all service times at stations 1, 2, and 3 are i.i.d. uniform random variables, then conditions in part (b) of Proposition 7 are satisfied, and hence Corollary 3 implies that an optimal policy gives priority to stations 3, 1, and 2 in that order. Similarly, when service times at stations 2 and 3 are independent and uniformly distributed over $[0, \theta]$, then the conditions in part (c) of Proposition 7 are satisfied if X_1 is uniformly distributed over $[a_1, b_1]$, where $a_1 \geq \theta$ and $b_1 \geq 2\theta$, and hence Corollary 3 implies that an optimal policy gives priority to stations 3, 2, and 1 in that order.

One can also get conditions similar to those in Proposition 7 for service times that have a gamma distribution. For example, when service times at stations 1, 2, and 3 are independent and gamma distributed with the same scale parameter but possibly different shape parameters of α_1, α_2, and α_3, respectively, then it can be shown that $X_3 \leq_{\mathrm{lr}} X_1 \leq_{\mathrm{lr}} X_2 + X_3$ if $\alpha_3 \leq \alpha_1 \leq \alpha_2 + \alpha_3$; see, for example, Table 1.1 in Müller and Stoyan [12]. Then, by part (b) of Corollary 3, there exists an optimal policy that gives priority to stations 3, 1, and 2 in that order.

5.3 Independent and identically distributed service times at all service stations

In this section we study the control problem over the set of non-preemptive policies for the general series-parallel network structure but under the assumption that service times at all service stations are i.i.d. We first show that in this setting, any policy can be improved (or at least its performance would not degrade) by replacing the service of a job with the service of another with fewer tasks remaining. We then show that it is sufficient to consider only those policies within Π_{NP} under which the server never idles whenever there is a job with fewer tasks remaining than the number of stations on the shortest branch of the network.

Proposition 8 *Suppose that the service times at all service stations are i.i.d. Then, a non-preemptive policy that serves a job with k tasks is at least as good as one that serves a job with j tasks remaining, where $k < j$, at any decision epoch in terms of maximizing the total number of departures from station 0 by time $t \geq 0$.*

Proposition 9 *Suppose that the service times at all service stations are i.i.d. If at a decision epoch there exists at least one job that has fewer tasks remaining than the total number of tasks in the shortest branch of the network, then idling is suboptimal in terms of maximizing $D_0^\pi(t)$ in the usual stochastic order for all $t \geq 0$ within Π_{NP}.*

Corollary 4 *Suppose that the service times at all service stations are i.i.d. If at a decision epoch there exists at least one job with k tasks remaining, where $k \leq \min_{j=1,\ldots,J} i_j$, then serving the job with the fewest number of tasks remaining maximizes $D_0^\pi(t)$ in the usual stochastic order for all $t \geq 0$ within Π_{NP}.*

Corollary 4 partially characterizes the optimal policy for networks with i.i.d. service times at all service stations and that have at least one branch that has a different number of stations than the others. On the other hand, it completely characterizes the optimal policy for networks with i.i.d. service times at all service stations and that have the same number of stations on each branch: The non-idling policy that gives priority to the job with the fewest number of tasks remaining maximizes $D_0^\pi(t)$ in the usual stochastic order for all $t \geq 0$ within Π_{NP}.

6 Numerical results

Our analytical results presented in Sects. 4 and 5 suggest that giving priority to the jobs that have the shortest remaining service time in some stochastic sense could be a good policy. Hence, in this section, we evaluate the *shortest-expected-remaining-service-time* policy when preemption is not permitted. (In accordance with the scheduling-theory nomenclature, we call this policy SEPT, which stands for "shortest expected processing time.") In particular, we compare the throughput under SEPT with the optimal throughput by means of a numerical study.

We first formally define SEPT as a server assignment policy for a general parallel-series system. Suppose at some decision epoch t, $N(t)$ jobs are in need of attention from the server. Let $S(n)$ be the total service time required for the nth job before it enters station 0, for $n = 1, 2, \ldots, N(t)$. SEPT ranks these $N(t)$ jobs in an ascending order of their $\mathrm{E}[S(n)]$ and prioritizes the one that ranks the first, and follows the same rule at each decision epoch then on.

We conducted a numerical study for the parallel system with three service stations and the two-branch-three-station system that are shown in Figs. 4 and 5, respectively. In all our experiments, service times at station i were independent and exponentially distributed with rate μ_i for $i = 0, 1, 2, 3$. We fixed the service rate of station 0 at $\mu_0 = 1$, and varied the service rates of other stations. We considered two subsets of experiments depending on the number of jobs circulating in the system, specifically, one subset with $B = 5$ and another with $B = 10$. We used the method of policy iteration to obtain the optimal policy for each scenario where the margin of error was set to 10^{-12}.

For the parallel system with three service stations, we let the service rate μ_i $(i = 1, 2, 3)$ take values from the set $\{0.5, 0.75, 1, 2, 5\}$. We considered four cases for $[p_1, p_2, p_3]$, namely $[0.2, 0.4, 0.4]$, $[0.4, 0.3, 0.3]$, $[0.6, 0.2, 0.2]$, and $[0.8, 0.1, 0.1]$. This resulted in 1,000 scenarios in total. The numerical results, which we do not present here, showed that SEPT is optimal for all 1,000 scenarios within the set of non-idling and non-preemptive policies.

For the two-branch-three-station system, we let μ_1 and μ_2 take values from the sets $\{0.5, 0.75, 1, 2, 5\}$ and $\{0.5\mu_1, 1.5\mu_1, 3\mu_1\}$, respectively, and set $\mu_3 = \mu_2$. We consider three levels for p_1, namely 0.25, 0.5, and 0.75. This resulted in 90 scenarios in total, for none of which our analytical results presented in earlier sections yield the optimal policy. For each scenario, we computed the throughput under SEPT, the optimal throughput (TH*), and the percentage deviation (PD) of the SEPT throughput from TH*. These results are presented in Table 1.

From Table 1, it can be seen that SEPT is optimal in 69 out of 90 scenarios and provides a performance that is very close to the optimal in others. A closer look also reveals that SEPT is suboptimal in scenarios where the expected service time of the single-station branch ($\mathrm{E}[X_1]$) is greater than the total expected service time for the branch with two service stations ($\mathrm{E}[X_2]+\mathrm{E}[X_3]$), and hence SEPT pools stations 2 and 3, and gives priority to the pooled station over station 1. Note that in these scenarios, the likelihood ratio order on total remaining service times does not hold, i.e., $X_2 + X_3$ is not smaller than X_1 in the likelihood ratio ordering. For example, consider the case where $(\mu_1, \mu_2, \mu_3) = (1, 3, 3)$; even though $\mathrm{E}[X_2]+\mathrm{E}[X_3] < \mathrm{E}[X_1]$, it can be shown that $X_2 + X_3$ is not smaller than X_1 in the likelihood ratio ordering. This shows that the likelihood ratio ordering conditions given in Corollary 3(c) cannot be replaced by orderings in expected values even when the objective is to maximize the long-run average throughput. Nevertheless, this numerical study also supports the use of SEPT by showing that it is optimal in the majority of the cases considered and near optimal in others. Over all 90 scenarios, the average and maximum percentage deviations from the optimal are only 0.00072 and 0.0127%, respectively.

7 Conclusion

In this paper, we studied the problem of dynamically assigning a single server to multiple stations in a finite-population parallel-series network of queues to maximize

Table 1 Performance of SEPT for the Markovian system with two branches and three stations under non-preemptive policies in terms of the percentage deviation (PD) from the optimal throughput (TH*)

μ_1	$\mu_2 = \mu_3$	$B = 5$						$B = 10$					
		$p_1 = 0.25$		$p_1 = 0.5$		$p_1 = 0.75$		$p_1 = 0.25$		$p_1 = 0.5$		$p_1 = 0.75$	
		PD ($\times 10^{-3}$)	TH*	PD ($\times 10^{-3}$)	TH*	PD ($\times 10^{-3}$)	TH*	PD ($\times 10^{-6}$)	TH*	PD ($\times 10^{-6}$)	TH*	PD ($\times 10^{-6}$)	TH*
0.5	$0.5\mu_1$	0	0.154	0	0.200	0	0.286	0	0.154	0	0.200	0	0.286
0.5	$1.5\mu_1$	0	0.400	0	0.429	0	0.462	0	0.400	0	0.429	0	0.462
0.5	$3.0\mu_1$	6.97	0.665	6.59	0.600	5.76	0.545	0.017	0.667	0.017	0.600	0.016	0.546
0.75	$0.5\mu_1$	0	0.231	0	0.300	0	0.429	0	0.231	0	0.300	0	0.429
0.75	$1.5\mu_1$	0	0.600	0	0.643	0	0.692	0	0.600	0	0.643	0	0.692
0.75	$3.0\mu_1$	12.7	0.999	11.3	0.899	8.98	0.817	0.786	1.000	0.729	0.900	0.680	0.818
1	$0.5\mu_1$	0	0.308	0	0.400	0	0.571	0	0.308	0	0.400	0	0.571
1	$1.5\mu_1$	0	0.800	0	0.857	0	0.922	0	0.800	0	0.857	0	0.923
1	$3.0\mu_1$	4.39	1.327	3.74	1.195	3.33	1.087	9.89	1.333	8.64	1.200	7.65	1.091
2	$0.5\mu_1$	0	0.615	0	0.800	0	1.141	0	0.615	0	0.800	0	1.143
2	$1.5\mu_1$	0	1.586	0	1.692	0	1.808	0	1.600	0	1.714	0	1.846
2	$3.0\mu_1$	0	2.484	0	2.270	0	2.085	576.0	2.665	491.0	2.400	326.0	2.182
5	$0.5\mu_1$	0	1.527	0	1.951	0	2.603	0	1.539	0	2.000	0	2.857
5	$1.5\mu_1$	0	3.272	0	3.379	0	3.481	0	3.993	0	4.271	0	4.585
5	$3.0\mu_1$	0	4.007	0	3.861	0	3.717	0	6.320	0	5.787	0	5.317

the long-run average reward/throughput. Under several scenarios, we were able to analytically show that assigning the server to a job that has the shortest remaining service (in some stochastic sense) is optimal. The intuitive explanation for such an optimal policy is that by pushing jobs out of service as fast as possible, this policy increases the rate of arrival of new jobs, and hence, reduces the idle time of the server. For example, in a series network with a single branch, the optimal policy gives priority to stations that are closest to the end of the line. For networks with parallel branches of stations, we identified stochastic ordering conditions among service times at different stations so that the optimal policy could be (partially) characterized as one that gives priority to jobs that are faster to complete. For example, for a network of two stations that are parallel, we showed that the faster station should receive priority over the other if the service times at these two stations are likelihood ratio ordered.

Based on this theoretical support for policies that give priority to jobs with shortest remaining service times, we conducted a numerical study to test the performance of the shortest-expected-remaining-service-time policy, which is commonly called SEPT in scheduling literature. Our numerical results for parallel-series networks of three stations (with either two or three branches) show that SEPT is optimal in the majority of the cases considered, and when it is not optimal, its performance is very close to the optimal.

Acknowledgements The authors thank the associate editor and an anonymous referee for their comments on an earlier version of this article, which significantly improved it. The work of the first author was partially supported by the National Science Foundation under grants CMMI–1234212 and CMMI–1635574.

Appendix

In this appendix, we prove the analytical results presented in the main paper and provide supplemental material.

Proof of Proposition 1 Let $D^{\pi}_{0,j}(t)$ be the number of customers who request service from the jth branch after leaving station 0 during $[0, t]$ under policy π. This means that $\sum_{j=1}^{J} D^{\pi}_{0,j}(t) = D^{\pi}_{0}(t)$. Let also $C^{\pi}_{i,j}(t)$ denote the number of customers in station (i, j) at time t $(1 \leq i \leq i_j$ and $1 \leq j \leq J)$. Then, for $1 \leq i \leq i_j$ and $1 \leq j \leq J$, we have

$$C^{\pi}_{i,j}(t) = C^{\pi}_{i,j}(0) + D^{\pi}_{i-1,j}(t) - D^{\pi}_{i,j}(t),$$

which implies

$$D^{\pi}_{i,j}(t) = D^{\pi}_{0,j}(t) + \sum_{k=1}^{i} \left(C^{\pi}_{k,j}(0) - C^{\pi}_{k,j}(t) \right). \tag{4}$$

Hence, we have

$$R^{\pi} = \liminf_{t\to\infty} \sum_{j=1}^{J} \sum_{i=1}^{i_j} \frac{r_{i,j} \mathrm{E}\left[D_{0,j}^{\pi}(t)\right]}{t} + \liminf_{t\to\infty} \sum_{j=1}^{J} \sum_{i=1}^{i_j} r_{i,j} \sum_{k=1}^{i} \frac{\mathrm{E}\left[C_{k,j}^{\pi}(0) - C_{k,j}^{\pi}(t)\right]}{t}.$$

Since $0 \leq C_{k,j}^{\pi}(t) \leq B < \infty$ for all $t \geq 0$, $j = 1, \ldots, J$, and $k = 1, \ldots, i_j$, the last term in the above equation is zero, which leads to

$$\begin{aligned} R^{\pi} &= \liminf_{t\to\infty} \sum_{j=1}^{J} \sum_{i=1}^{i_j} \frac{r_{i,j} \mathrm{E}\left[D_{0,j}^{\pi}(t)\right]}{t} \\ &= \liminf_{t\to\infty} \sum_{j=1}^{J} \sum_{i=1}^{i_j} \frac{r_{i,j} p_j \mathrm{E}\left[D_{0}^{\pi}(t)\right]}{t} \\ &= T^{\pi} \sum_{j=1}^{J} \sum_{i=1}^{i_j} p_j r_{i,j}, \end{aligned}$$

where the second equation follows from the independent Bernoulli splitting of customers departing from station 0. □

Proof of Proposition 4 We will prove the result by means of a sample-path argument. As is commonly used in proofs for characterizing optimal policies within a set of preemptive policies, we consider a discrete-time version of the problem, where the discretization can be arbitrarily small. Let T be the number of periods left in this discrete-time problem. We will use induction on T to prove the result. When $T = 1$, all policies will result in the same number of departures from station 0, and hence the policy given in the statement of the proposition is trivially optimal. Now, suppose that the same policy is optimal for a finite-horizon problem with $T - 1$ periods. We will next show that it is also optimal for a problem with T periods.

Let π be a policy that does not follow the optimal policy defined in the proposition at time zero under the problem with T periods. From the inductive hypothesis, we can assume that π follows the optimal policy from time 1 on. We next construct a new policy γ that follows the optimal policy at time 0 and generates at least the same number of departures from station 0 as policy π along the entire sample path.

Let C_1 be the job that policy π takes into service at time 0 and C_2 be the job that policy γ takes at time 0. (We here consider only the case where policy π serves a job at time 0; a similar argument can be used when π idles at time 0.) Suppose that job C_i has k_i tasks remaining (including the task at the current station at time 0) and t_i units of service time attained at the current station for $i = 1, 2$. We need to consider two cases:

Case (i) We first consider the case with $k_1 = k_2$ and $t_1 < t_2$. (The case with $k_1 = k_2$ and $t_1 = t_2$ will trivially hold.) Since π follows the optimal policy at time 1 and on, it will not serve C_1 again until C_2 is served. Let τ be the first time π takes C_2 into service. (We assume that $\tau \leq T$ but the results would trivially hold if the server never takes C_2 into service within the horizon.) We now construct γ such that it agrees with π at all decisions except that it serves C_2 at time 0 and C_1 at time τ, and thereafter serves C_1 or C_2 with priority to C_2 whenever π serves C_1 or C_2. Let S_i be the remaining service time of job C_i at the current station at time 0 under policy π and S_i' be the corresponding service time under policy γ. Suppose that we directly couple the service times of all jobs other than C_1 and C_2 under both policies, and couple service time of job $C_1[C_2]$ at station 0 under policy π and service time of job $C_2[C_1]$ under policy γ. Since service times are i.i.d. and ILR, and $t_1 < t_2$, we have $S_2 \leq_{\rm lr} S_1$ and $S_2' \leq_{\rm lr} S_1'$. Hence, we generate $m = \min\{S_1, S_2\}$ and $M = \max\{S_1, S_2\}$ from the appropriate distribution and use them for sample paths of both policies. If $m = M = 1$, then we have $S_1 = S_1' = S_2 = S_2' = 1$, and the two sample paths will couple at time $\tau + 1$, i.e., both sample paths will reach the same state at this time. Furthermore, we will have $D_0^\pi(t) = D_0^\gamma(t)$ for all $t = 0, 1, \ldots, T$. If $1 < m \leq M$, then we let $S_1 = S_1' = M$ and $S_2 = S_2' = m$ with probability $p = \Pr\{S_1 = M, S_2 = m | \min\{S_1, S_2\} = m, \max\{S_1, S_2\} = M\}$ and we let $S_1 = S_1' = m$ and $S_2 = S_2' = M$ with probability $1 - p$. In each case, the sample paths will couple at $\tau + 1$. Furthermore, we will again have $D_0^\pi(t) = D_0^\gamma(t)$ for all $t = 0, 1, \ldots, T$. If $1 = m < M$, then we let $S_1 = S_2' = M$ and $S_2 = S_1' = 1$ with probability $1 - p$, we let $S_1 = S_2' = 1$ and $S_2 = S_1' = M$ with probability $1 - p$, and we let $S_1 = S_1' = M$ and $S_2 = S_2' = 1$ with probability $2p - 1$. This generation works because $p \geq 1 - p$ by Theorem 1.C.24 of Shaked and Shanthikumar [17] and the likelihood ratio ordering between S_1 and S_2, and it will yield the correct probabilities, i.e., $\Pr\{S_1 = M, S_2 = 1 | \min\{S_1, S_2\} = 1, \max\{S_1, S_2\} = M\} = \Pr\{S_1' = M, S_2' = 1 | \min\{S_1', S_2'\} = 1, \max\{S_1', S_2'\} = M\} = p$. In the first two subcases, the interchange has no effect and the sample paths again couple at $\tau + 1$ with $D_0^\pi(t) = D_0^\gamma(t)$ for all $t = 0, 1, \ldots, T$. In the third subcase, if $k_1 > 1$, then the sample paths under both policies will again couple at $\tau + 1$ with no differences in the number of departures from station 0. However, if $k_1 = 1$, then job C_2 will enter station 0 at time 1 under policy γ and at time $\tau + 1$ under policy π. Let S_0 be the service time of C_2 at station 0 at this entrance to that station under both policies. Then, the two sample paths will couple at time $\tau + 1 + S_0$ and we will have

$$D_0^\gamma(t) = \begin{cases} D_0^\pi(t), & 0 \leq t < S_0 + 1, \\ D_0^\pi(t) + 1, & S_0 + 1 \leq t < \tau + 1 + S_0, \\ D_0^\pi(t), & t \geq \tau + 1 + S_0. \end{cases}$$

Case (ii) We now consider the case with $k_1 > k_2$. Let $\overline{S}_i$ be the total service time remaining at time 0 for job C_i until it enters station 0 for $i = 1, 2$ under policy π. Thus, we can write $\overline{S}_i = \sum_{j=1}^{k_i} S_j(i)$, where $S_j(i)$ is the service time of job C_i's jth task remaining for $i = 1, 2$. Note that the $S_j(i)$ are i.i.d. except for $S_1(1)$ and $S_1(2)$. Furthermore, since service times at all stations are ILR, by Theorem 1.C.9 in Shaked and Shanthikumar [17] (with the discussion that follows the theorem) and the facts that $k_1 > k_2$ and the $S_j(i)$ are non-negative, we have $\overline{S}_2 \leq_{\rm lr} \overline{S}_1$. Now, in the proof of Case (i), setting $k_1 = k_2 = 1$ and replacing S_i with $\overline{S}_i$ completes the proof. □

We defer the proof of Proposition 5 as it is based on an argument used in the proof of Proposition 6.

Proof of Proposition 6 We will use an inductive sample-path argument to prove the result. Suppose at time zero the server can only perform a total of $v \geq 1$ more tasks, at which point the server is shut off and the problem stops. Applying induction on v and letting $v \to \infty$ will complete the proof.

Suppose π is a policy under which the server takes a job into service in station k with $k \neq (i_{j^*}, j^*)$ at a decision epoch where $v = 1$ and there exists a job available in station (i_{j^*}, j^*). (For convenience, we here abuse the notation and instead of using a 2-tuple to refer to a station we simply call it station k.) Since $v = 1$, such an event could only happen once in a sample path. Without loss of generality, we assume that this event takes place at time zero. Now, consider another policy γ that serves a job in station (i_{j^*}, j^*) at time zero. Note that once π and γ complete the respective tasks, there will be no other decisions left under either policy since $v = 1$. To generate the sample paths under policies π and γ, we directly couple the sojourn times of all jobs at station 0. If k is an intermediate station, i.e., it is not an end station that will immediately lead to an arrival to station 0, then it is easy to see that $D_0^{\gamma}(t) \geq D_0^{\pi}(t)$ for all $t \geq 0$ as (i_{j^*}, j^*) is an end station. Now consider the case where k is another end station. Let S_1' and S_2 be the service time of the job taken into service at time zero under policies γ and π, respectively. Since (i_{j^*}, j^*) is the fastest end station in likelihood ratio ordering (and thus in usual stochastic ordering), we can couple the service times such that $S_1' \leq S_2$ with probability one. This means that policy γ will have an earlier arrival at station 0 than policy π, and directly coupling the sojourn times of these two arrivals at station 0 will yield $D_0^{\gamma}(t) \geq D_0^{\pi}(t)$ for all $t \geq 0$. This concludes the proof that at a decision epoch where $v = 1$ and there exists a job at station (i_{j^*}, j^*), it is better to take this job into service than to take any other job. Having this result, a simple sample-path argument would also immediately lead to the suboptimality of idling at such a decision epoch. This completes the proof of the proposition for $v = 1$.

Now, suppose that the proposition holds when the server can serve at most $v - 1$ more tasks. We will next show that a policy, say policy π, which assigns the server to a station other than station (i_{j^*}, j^*) at a decision epoch where there is a job at station (i_{j^*}, j^*) and the maximum number of tasks that could be served is v, can be improved by serving the job in station (i_{j^*}, j^*) at that decision epoch. Again, without loss of generality, assume that time zero is the first time policy π starts working in station k, where $k \neq (i_{j^*}, j^*)$, despite the fact that there is a job in station (i_{j^*}, j^*). At the time of completion of service of this job in station k, the total number of tasks that the server can take on will drop to $v - 1$, and hence, by the inductive argument, we assume that policy π starts serving a job in station (i_{j^*}, j^*) at this time. Let S_2 be the service time of the job that the server picks from station k right before the server moves to station (i_{j^*}, j^*) to serve a job there with service time S_1 under policy π. We construct an alternative policy γ as follows: γ serves a job in station (i_{j^*}, j^*) with service time S_1' at time zero, switches to station k, and serves a job there with service time S_2'.

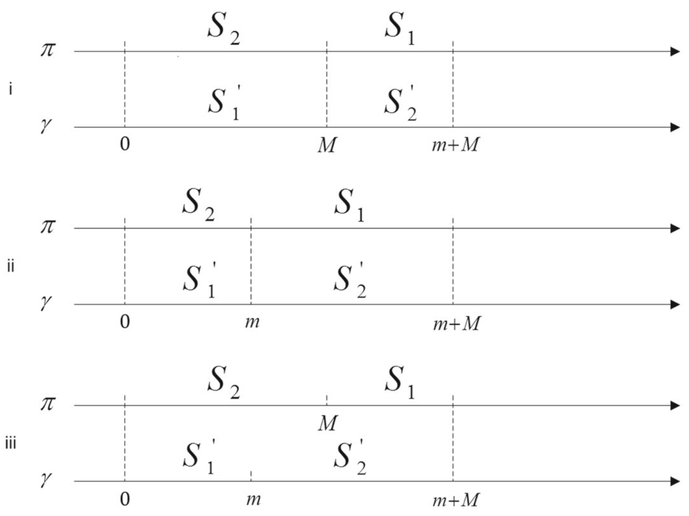

Fig. 6 Visual depiction of sample-path couplings used in the proof of Proposition 6

Case 1 (k is an intermediate station) We directly couple the service times of all jobs taken into service during $[0, \infty)$ under both policies, which yields $S'_1 = S_1$ and $S'_2 = S_2$. Let S_0 be the service time at station 0 for the job entering station 0 at time $S_2 + S_1$ under policy π and at time S'_1 under policy γ. We then let γ follow π during $[S_1 + S_2, \infty)$. This is possible because at least the same number of jobs are available to policy γ compared to policy π and idling is allowed. The system states for the two policies will become identical after time $S_2 + S_1 + S_0$. With this construction, it is then easy to see that $D_0^\gamma(t) \geq D_0^\pi(t)$ for all $t \geq 0$.

Case 2 (k is an end station) We cross couple S_1, S_2, S'_1, and S'_2 as follows. We first generate the minimum and maximum of S_1 and S_2, namely m and M, respectively, condition on their values, and use these values in both sample paths. Let $p = \Pr\{S_1 = M|m, M\} = \Pr\{S_2 = m|m, M\}$ and $q = \Pr\{S_1 = m|m, M\} = \Pr\{S_2 = M|m, M\} = 1 - p$. By Lemma 13.D.1(i) of Righter [14] (or Theorem 1.C.24 of Shaked and Shanthikumar [17]) and the likelihood ratio ordering between S_1 and S_2, we have $p < q$. Thus, we can let

(i) $S_1 = m$, $S_2 = M$, $S'_1 = M$, $S'_2 = m$, with probability p,
(ii) $S_1 = M$, $S_2 = m$, $S'_1 = m$, $S'_2 = M$, with probability p,
(iii) $S_1 = m$, $S_2 = M$, $S'_1 = m$, $S'_2 = M$, with probability $1 - 2p$.

The coupling yields $S'_1 \leq S_2$ (and $S_1 \leq S'_2$) in all three cases. See Fig. 6 for a visual depiction of this coupling. In the first two cases all arrival times to station 0 under policies π and γ are identical. Furthermore, the system states for the two policies become identical after time $m + M$. Hence, policy γ can follow policy π thereafter. By directly coupling the service times of all jobs taken into service after time $m + M$, we have $D_0^\gamma(t) = D_0^\pi(t)$ for all $t \geq 0$ under cases (i) and (ii).

In case (iii), let S_0 be the service time at station 0 for the job entering station 0 at time S_2 under policy π and at time S'_1 under policy γ. We directly couple the service times of all jobs taken into service after $m + M$. Note that the system states for the two policies will become identical after time $M + \max\{m, S_0\}$. However, policy γ can follow π after $m + M$ because the same jobs or more will be available to policy γ compared to policy π and idling is allowed. We then have $D_0^{\gamma}(t) \geq D_0^{\pi}(t)$ for all $t \geq 0$. This completes the proof that it is better to take a job in station (i_{j^*}, j^*) into service than to take any other job at a decision epoch where the maximum number of tasks that can be served is v.

We finally prove that idling is suboptimal at a decision epoch (say, at time zero, without loss of generality), where the maximum number of tasks that can be served is v and there exists a job in station (i_{j^*}, j^*). Consider policy π that idles the server for I units of time at time zero. We already proved that if the server would be assigned to a job, then it is best to assign to one in station (i_{j^*}, j^*) (whenever there exists one) at a decision epoch where the maximum number of tasks that can be served is v. Hence, we assume that π assigns the server to a job at station (i_{j^*}, j^*) with service time S_1 at time I. Consider an alternative policy γ that serves a job in station (i_{j^*}, j^*) with service time S_1' at time zero and then starts idling for I units of time. We directly couple the service times of all jobs taken into service and sojourn times of all jobs at station 0 during $[0, \infty)$, which means that $S_1' = S_1$. Let S_0 be the service time at station 0 for the job entering station 0 at time S_1 under policy γ and at time $I + S_1$ under policy π. The system states under π and γ will become identical at $I + S_1 + S_0$. However, policy γ can follow π after $I + S_1$ because the same jobs or more will be available to policy γ compared to policy π and idling is allowed. This construction will yield $D_0^{\gamma}(t) \geq D_0^{\pi}(t)$ for all $t \geq 0$, which completes the proof. □

We need the following lemma to prove Proposition 5 and Corollary 1.

Lemma 1 *Suppose that station (i_{j^*}, j^*) has the highest priority under an optimal policy in Π_{NP} where $j^* \in \{1, 2, \ldots, J\}$. If $i_{j^*} > 1$, then, within Π_{NP}, it suffices to consider policies that pool the two tasks at stations $(i_{j^*} - 1, j^*)$ and (i_{j^*}, j^*) for all jobs that enter station $(i_{j^*} - 1, j^*)$ in order to maximize $D_0^{\pi}(t)$ in the usual stochastic order for all $t \geq 0$.*

Proof of Lemma 1 Note that the only source of jobs for station (i_{j^*}, j^*) is station $(i_{j^*} - 1, j^*)$ when $i_{j^*} > 1$. Hence, when seeking an optimal policy, it is sufficient to only consider policies that sequentially serve stations $(i_{j^*} - 1, j^*)$ and (i_{j^*}, j^*), i.e., policies under which the tasks at these two stations are essentially pooled, given that station (i_{j^*}, j^*) has the highest priority under an optimal policy in Π_{NP}. □

Proof of Proposition 5 When $J = 1$, using the arguments for the case where k is an intermediate station in the proof of Proposition 6, we can prove that the end station $(i_1, 1)$ should receive the highest priority. Then, by Lemma 1, we can consider only policies that pool the two stations at the end of the branch. Repeating this argument several times proves Proposition 5. □

Proof of Corollary 1 Applying Lemma 1 and Proposition 6 in Steps **S.3**a and **S.3**b of Algorithm A-1 proves Corollary 1. □

We need Definition 1 and Lemma 2 to prove Proposition 7.

Definition 1 Y has a trapezoidal distribution with parameters $a \leq b \leq c \leq d$ if its probability density function $f(x)$ is given by

$$f(x) = \begin{cases} \frac{2(x-a)}{(b-a)(c+d-a-b)}, & \text{if } a \leq x \leq b, \\ \frac{2}{c+d-a-b}, & \text{if } b \leq x \leq c, \\ \frac{2(d-x)}{(d-c)(c+d-a-b)}, & \text{if } c \leq x \leq d, \\ 0, & \text{otherwise.} \end{cases}$$

Lemma 2 *Let X and Y be two continuous random variables, where X is uniformly distributed over $[a_1, b_1]$ and Y has a trapezoidal distribution with parameters $a \leq b \leq c \leq d$.*

(i) *If $a_1 \leq a$ and $b_1 \leq c$, then $X \leq_{\text{lr}} Y$.*
(ii) *If $a_1 \geq b$ and $b_1 \geq d$, then $Y \leq_{\text{lr}} X$.*

Proof of Lemma 2 (i) Let f and g be the probability density functions for X and Y, respectively. We will show that $f(x)g(y) \geq f(y)g(x)$ for all $x \leq y$, where $x, y \in [a_1, d]$. When $x \in [a_1, a)$, we have $g(x) = 0$, which implies that $f(x)g(y) \geq f(y)g(x)$ for all $y \in [a_1, d]$. The proof will be then complete once we show that $f(x)/g(x)$ is non-increasing for $x \in [a, d]$. For $x \in [a, b_1]$ (when $a \leq b_1$), $g(x)$ is non-decreasing by the condition that $b_1 \leq c$ and $f(x)$ is a constant, and hence, $f(x)/g(x)$ is non-increasing. For $x \in [b_1, d]$, $f(x)/g(x)$ is trivially non-increasing as $f(x)$ is zero, which completes the proof.
(ii) Follows directly from part (i), by noting that $-X$ is uniformly distributed on $[-b_1, -a_1]$ and $-Y$ has a trapezoidal distribution with parameters $-d \leq -c \leq -b \leq -a$. □

Proof of Proposition 7 (a) In Lemma 2, if $a = b$ and $c = d$, then Y has a uniform distribution on $[a, c]$, from which the result follows immediately.
(b) The first stochastic inequality is already proved in part (a). For the ordering between X_1 and $X_2 + X_3$, first note that $X_2 + X_3$ has a trapezoidal distribution with parameters $a \leq b \leq c \leq d$, where $a = a_2 + a_3$, $b = a_2 + a_3 + \min\{b_2 - a_2, b_3 - a_3\}$, $c = b_2 + b_3 - \min\{b_2 - a_2, b_3 - a_3\}$, and $d = b_2 + b_3$ by Proposition 2.1 in Korwar [11]. Since $a_1 \leq a_2 + a_3$ and $b_1 \leq b_2 + b_3 - \min\{b_2 - a_2, b_3 - a_3\}$, part (i) of Lemma 2 yields that $X_1 \leq_{\text{lr}} X_2 + X_3$.
(c) Conditions imply that $a_1 \geq a_3$ and $b_1 \geq b_3$, and hence by part (a), $X_3 \leq_{\text{lr}} X_1$. For the ordering between X_1 and $X_2 + X_3$, we know from the proof of part (b) that $X_2 + X_3$ has a trapezoidal distribution. Since $a_1 \geq a_2 + a_3 + \min\{b_2 - a_2, b_3 - a_3\}$ and $b_1 \geq b_2 + b_3$, part (ii) of Lemma 2 yields that $X_2 + X_3 \leq_{\text{lr}} X_1$. □

Proof of Proposition 8 We prove the result by a sample-path argument. Suppose that π is a policy that chooses to serve a customer that has j tasks remaining (say, customer C_1) at time $\tau_1 > 0$, although there exists a customer (say, customer C_2) that has $k < j$ tasks remaining. We will next construct a new policy γ that follows π during $[0, \tau_1)$.

Let t_i be the first time that job C_i enters station 0 after τ_1 for $i = 1, 2$. Let also $\underline{\tau} = \min\{t_1, t_2\}$ and $\overline{\tau} = \max\{t_1, t_2\}$, where $\tau_1 < \underline{\tau} < \overline{\tau}$. During $[\tau_1, \overline{\tau})$, let γ follow π exactly except that γ works on C_1 or C_2 with preference for C_2 whenever π works on

C_1 or C_2. We directly couple all service times at the service stations and station 0 that start at the same decision epoch under both policies during $[\tau_1, \overline{\tau}]$. Such a coupling would work because service times at all service stations are i.i.d. Let $\underline{S}_0$ be the service time (at station 0) of C_1 or C_2 that starts at time $\underline{\tau}$ and $\overline{S}_0$ be the service time (at station 0) of the other one that starts at time $\overline{\tau}$ under policy π. By the construction of policy γ, C_2 will enter station 0 earlier than C_1 under policy γ; denote this time by $\underline{\tau}'$. Note that we must have $\underline{\tau}' < \underline{\tau}$ due to the construction of policy γ and the fact that $k < j$. We couple the two sample paths such that $\underline{S}_0$ and $\overline{S}_0$ are the service times of C_2 and C_1 at station 0 under policy γ, respectively. (The service times of all other jobs at station 0 are directly coupled under π and γ.) At time $\overline{\tau}$, the same jobs or more will be available to the server under policy γ when compared with policy π. Hence, policy γ can follow policy π after $\overline{\tau}$ given that idling is allowed. The two sample paths will eventually couple at time $\max\{\underline{\tau} + \underline{S}_0, \overline{\tau}\}$. For such a coupling, we have $D_0^{\gamma}(t) \geq D_0^{\pi}(t)$, for all $t \geq 0$. □

Proof of Proposition 9 Suppose policy π is a policy that idles at time $\tau_1 > 0$, when there exists at least one job that has fewer tasks remaining than the total number of tasks in the shortest branch of the network. Without loss of generality, assume that τ_1 is the first such decision epoch. Let C_1 be the job that has the fewest number of tasks remaining at time τ_1 and k be the number of tasks it has remaining, where $k \leq \min_{j=1,\ldots,J} i_j$. Let τ_2 be the first time after τ_1 at which the server starts working on a job. Based on Proposition 8, we need to only consider the case where policy π starts working on job C_1 at time τ_2. Note that job C_1 will still be the job with the fewest number of tasks remaining at time τ_2 because any job that might have arrived during $[\tau_1, \tau_2)$ will have at least $\min_{j=1,\ldots,J} i_j$ tasks remaining. Using similar arguments as in our earlier sample-path results, we can construct a new policy γ that follows π during $[0, \tau_1)$ but serves job C_1 at time τ_1 such that $D_0^{\gamma}(t) \geq D_0^{\pi}(t)$ for all $t \geq 0$. □

We finally provide a counter example to the conjecture that serving the job with the shortest service time in the likelihood ratio ordering maximizes the number of departures in a parallel network of more than two stations under non-preemptive and non-idling policies.

Example 1 Consider a parallel network of three service stations with deterministic service times of i units of time at service station i, for $i = 1, 2, 3$, and $B \geq 3$. Suppose that at a decision epoch, say at time zero, there are no jobs at station 1 but a single job at each of stations 2 and 3, and the rest of the $B - 2$ jobs are at station 0. Suppose also that under this sample path the first departure from station 0 among the $B - 2$ jobs initially there takes place at time 3 and enters station 1. Under the non-idling and non-preemptive shortest-expected-service-time policy, the server will start serving at station 2 at time zero and then at station 3 at time 2. Hence, by time 4, only a single job would have entered station 0. Now consider another policy that serves the job at station 3 at time zero, and then takes the job at station 1 into service at time 3 right after it arrives. Under this alternative policy, two jobs enter station 0 by time 4, which implies that this policy could have a larger number of departures from station 0 than the shortest-expected-service-time policy for some $t \geq 4$.

References

1. Ahn, H.-S., Righter, R.: Dynamic load balancing with flexible workers. Adv. Appl. Probab. **38**(3), 621–642 (2006)
2. de Véricourt, F., Jennings, O.B.: Nurse staffing in medical units: a queueing perspective. Oper. Res. **59**(6), 1320–1331 (2011)
3. Dshalalow, H.J.: Frontiers in Queueing: Models and Applications in Science and Engineering. CRC Press, Boca Raton (1997)
4. Haque, L., Armstrong, M.J.: A survey of the machine interference problem. Eur. J. Oper. Res. **179**(2), 469–482 (2007)
5. Haverkort, R.B.: Performance of Computer Communication Systems: A Model-Based Approach. Wiley, New York (1998)
6. Hopp, W.J., Iravani, S.M.R., Shou, B.Y., Lien, R.: Design and control of agile automated conwip production lines. Nav. Res. Logist. **56**(1), 42–56 (2009)
7. Iravani, S.M.R., Kolfal, B.: When does the cμ rule apply to finite-population queueing systems? Oper. Res. Lett. **33**(3), 301–304 (2005)
8. Iravani, S.M.R., Krishnamurthy, V., Chao, G.H.: Optimal server scheduling in nonpreemptive finite-population queueing systems. Queueing Syst. **55**(2), 95–105 (2007)
9. King, J.B.P.: Computer and Communication System Performance Modelling. Prentice-Hall, Upper Saddle River (1990)
10. Koole, G., Righter, R.: Optimal control of tandem reentrant queues. Queueing Syst. **28**(4), 337–347 (1998)
11. Korwar, R.M.: On stochastic orders for sums of independent random variables. J. Multivar. Anal. **80**, 344–357 (2002)
12. Müller, A., Stoyan, D.: Comparison Methods for Stochastic Models and Risks. Wiley, Chichester (2002)
13. Palesano, J., Chandra, J.: A machine interference problem with multiple types of failures. Int. J. Prod. Res. **24**(3), 567–582 (1986)
14. Righter, R.: Scheduling. In: Shaked, M., Shanthikumar, J.G. (eds.) Stochastic Orders and Their Applications, Chapter 13, pp. 381–432. Academic Press, New York (1994)
15. Righter, R., Shanthikumar, J.G.: Extremal properties of the fifo discipline in queueing networks. J. Appl. Probab. **29**(4), 967–978 (1992)
16. Schrage, L.: A proof of the optimality of the shortest remaining processing time discipline. Oper. Res. **16**(3), 687–690 (1968)
17. Shaked, M., Shanthikumar, J.G.: Stochastic Orders. Springer, New York (2007)
18. Stecke, K.E., Aronson, J.E.: Review of operator/machine interference models. Int. J. Prod. Res. **23**(1), 129–151 (1985)
19. Sztrik, J.: Finite source queuing systems and their applications: A bibliography. http://irh.inf.unideb.hu/user/jsztrik/research/fsqreview.pdf (2001)
20. Takagi, H.: Stochastic Analysis of Computer and Communication Systems. North-Holland, Amsterdam (1990)
21. Van Oyen, M.P., Gel, E.G.S., Hopp, W.J.: Performance opportunity for workforce agility in collaborative and noncollaborative work systems. IIE Trans. **33**(9), 761–777 (2001)