Noname manuscript No.
(will be inserted by the editor)

The Cost of Collaboration

Kristen Gardner · Rhonda Righter

Received: date / Accepted: date

Abstract Multi-class multi-server systems, in which each job class can be served by a subset of servers, are common in a wide variety of applications. In such systems, a key question is whether to collaborate, allowing a job to be in service on multiple servers simultaneously, or not to collaborate. For systems with redundancy and exponential service times, this question is equivalent to the question of whether to cancel copies of a job once a copy starts service or once a copy completes service. While the collaborative and noncollaborative systems have been well-studied in steady-state, existing results have not allowed for a comprehensive comparison of performance in the two systems. In this paper, we use a combination of steady-state analysis and sample-path arguments to study the costs and benefits of collaboration. While intuition says that collaboration should yield substantial benefits when service times are not correlated, we find that, surprisingly, this is not the case. In systems that exhibit symmetry among servers or job classes, we can construct coupled sample paths so that collaboration indeed always outperforms noncollaboration. However, in all systems the benefit of collaboration is limited: we show that on sample paths with the same arrival and service processes, there is an upper bound on this benefit, in terms of the number of jobs in system. On the other hand, collaboration can be arbitrarily worse than noncollaboration on sample paths with the same input processes. Ultimately, both in steady-state and sample-path-wise, the two systems typically achieve similar performance,

K. Gardner
Department of Computer Science
Amherst College
E-mail: kgardner@amherst.edu

R. Righter
Department of Industrial Engineering and Operations Research
University of California, Berkeley
E-mail: rrighter@berkeley.edu

indicating that collaboration is never a big win and that, in many real-world systems, noncollaboration may be the better choice.

1 Introduction

Multi-class multi-server queueing models, in which each server is capable of processing only a subset of the possible job classes, have received a great deal of attention in the recent literature because of the breadth of applications of such models. For example, in large computing systems, data locality and server configurations may restrict the set of servers on which a job can run, or a job may be routed to multiple servers as a form of straggler mitigation to improve response time. In computing and other networks, jobs arriving to a node can be routed to nearest neighbors. In manufacturing and service systems with human operators, operators may be trained to handle different types of jobs, and they may have different levels of training.

There are two variants for serving jobs in these flexible-job, flexible-server systems: the collaborative model, in which a job can be in service at multiple servers simultaneously, and the noncollaborative model, in which a job can enter service on only one server. Both variants are of practical interest, as each is appropriate for modeling different applications. In call centers with skill-based routing, in which different agents are trained to handle different types of calls, the noncollaborative model is appropriate because one customer cannot be on the line with multiple agents simultaneously. Similarly, manufacturing systems that use worker cross training and work sharing to increase efficiency are sometimes best modeled by the noncollaborative model; one example of this setting is zone-based cross training in assembly lines. In contrast, systems such as regional organ transplant waitlists are best captured by the collaborative model; here, being "in service" corresponds to being next in line to receive an organ, and a patient can be in service in multiple geographical regions' waitlists at the same time. But in yet other applications, whether or not to collaborate is a design choice. For example, in some systems with human operators it may be possible for operators to work together to accomplish a task, and whether or not to allow this is a design choice. Our goals in this paper are to identify the circumstances under which collaboration is beneficial in such systems, and to characterize the benefit or cost of collaboration.

Our problem can also be viewed as a dispatching problem. For certain symmetric systems, joining the shortest of two randomly selected compatible queues has been shown to perform almost as well as choosing the shortest among all compatible queues [24]. An alternative dispatching rule, common in large cloud computing systems, is to send redundant copies of jobs to all compatible queues, or to a randomly chosen subset of compatible queues of size d. Here, we have a choice of whether to collaborate, allowing the job to run on multiple servers and cancelling all extra copies as soon as the first copy *completes* service, or not to collaborate, cancelling all extra copies as soon as the first copy *starts* service on some server. The Cancel-on-Start variant of re-

dundancy allows us to implement JSW (join the smallest work, or least waiting time), without knowing job sizes. This is equivalent to having a single FCFS (first-come-first-served) queue with noncollaborative service. (See, e.g., [1,3, 9].) An example in which Cancel-on-Complete has been shown to be helpful is in volunteer desktop grids for scientific computing (e.g., BOINC.Berkeley.edu or foldingathome.org). Here multiple copies of the same job run in the background on different independent volunteer systems until the output of one of them is sent back to the system controller. Again, our goal is to identify settings in which Cancel-on-Complete (i.e., collaboration) outperforms Cancel-on-Start (i.e., noncollaboration), and vice versa, assuming exponential service times.

In some cases, the relative performance of collaboration and noncollaboration is known or intuitively clear, especially when service times are exponential and i.i.d., which is the setting we consider in this paper. For example, collaboration and noncollaboration have the same stability region, assuming Poisson arrivals, exponential services, and service is FCFS [2,13]. In this paper, we have two metrics of interest: number of jobs in system and response time (the time from when a job arrives to the system until it completes service), the means of which are of course related through Little's Law. Intuitively, it seems that collaboration should always lead to lower response times and queue lengths because the collaborative system always keeps servers busy when there is compatible work in the system, whereas in the noncollaborative system servers may idle even if a compatible job is present. In some cases, this intuition is accurate. For example, if there is a single job class that can be served on all servers, then the collaborative system is equivalent to an M/M/1 queue with rate equal to the sum of the individual service rates. In contrast, the noncollaborative system is equivalent to an M/M/M system with M parallel servers. In this case, collaboration is easily seen to be better in terms of response time. Similarly, in light traffic (in the limit as the total arrival rate goes to zero), collaboration has lower mean response time because, in this regime, with high probability there is at most one job in the system, and if that job is compatible with multiple servers then its response time will be lower under collaboration. In addition, it has been observed numerically that in the redundancy(d) model, in which each arriving job is replicated to d randomly chosen homogeneous servers, Cancel-on-Complete (collaboration) leads to lower steady-state mean response times than Cancel-on-Start (noncollaboration) [9]; we prove that this is true for all parameter values and for appropriately coupled sample paths. In general, however, we will see that the story is more complex: collaboration can be worse than noncollaboration, and it is never much better.

1.1 Challenges and Contributions

We begin by considering the collaborative and noncollaborative systems in steady state. For both systems, under FCFS scheduling, the stationary distribution of the queue state satisfies a product form [13,17,28]. In principle, the

explicit product forms could be evaluated to determine the range of parameters for which collaboration yields lower mean number of jobs or mean response time. In practice, aggregating states to go from the stationary distribution to mean performance metrics is challenging. For the collaborative model, the stationary distribution only yields closed-form results for aggregate performance metrics in a few particular system structures, mostly notably "nested" systems [15] and "line" systems [11]. For the noncollaborative model, closed-form aggregate results have been elusive even in nested systems because of the need to compute "matching probabilities," the probabilities that a job of a particular class runs on each compatible server, and probabilities for various configurations of idle and busy servers.

– We derive, for the first time, closed-form expressions for the steady-state distributions and means of per-class response times and number of jobs in the noncollaborative model, for a small system called the W model that has three classes of jobs and two servers.

Our findings for the W model in steady state reveal that collaboration is *not* always better; indeed, it can be much worse than noncollaboration in terms of the number of jobs in the system.

But the W model is just one small example of a flexible-job, flexible-server system, and even for this simple model, the resulting expressions are complicated. Furthermore, the steady-state results for the noncollaborative system require restrictions on the system dynamics: to obtain a product form stationary distribution one must either keep track of the order in which servers become idle and assign arriving jobs that find multiple compatible idle servers according to the ALIS ("Assign to Longest Idle Server") policy [2], or one must determine appropriate assignment probabilities that depend on which servers are busy and on the arrival order of the jobs in service on those servers [28].

For most of this paper, we turn to sample path arguments to compare performance in the collaborative and noncollaborative systems. This approach has several benefits. First, it avoids the complexity of the analytical expressions for steady-state performance measures, allowing us to obtain results for larger systems. Second, sample path arguments allow us to obtain stronger comparisons between the two systems than simply comparing mean performance metrics, and they hold regardless of whether the system is stable. For example, we are able to say that, under appropriate conditions, collaboration is better not just in terms of steady-state mean response time, but also in terms of stochastically minimizing the number-in-system process. Finally, we are able to consider more general settings: our results do not require any particular assignment rule for idle servers in the noncollaborative model, nor do they require Poisson arrivals for either model, so they hold even when the steady-state distribution is unknown. We can also consider more general scheduling policies than FCFS. Our sample path arguments shed light on the conditions under which collaboration is better than noncollaboration—and on the limits of this benefit.

- We show that, with coupled arrival and potential service completion processes, the benefit of collaboration is bounded: the number of jobs at any time under noncollaboration exceeds the number without collaboration by at most M, the total number of servers.
- We show that, again with coupled arrivals and potential service completions, the cost of collaboration is unbounded: the number of jobs under collaboration can be arbitrarily larger than that under noncollaboration with non-zero probability.

For symmetric systems, in which either the jobs or the servers are stochastically indistinguishable from one another, we can show, for coupled systems, that collaboration is always beneficial (i.e., there are always fewer jobs under collaboration). Because of our results above for a complete coupling of both the timing and the particular job classes for arrivals and service completions, our result for symmetric systems requires us to use a different, novel, coupling approach, in which we directly couple the timing of all arrivals and potential service completions, but we periodically resample job classes, while maintaining the correct marginal distributions. This approach allows us to obtain comparisons between the collaborative and noncollaborative systems that are not possible using either steady-state analysis or direct coupling techniques.

- We show that, in symmetric systems, we can construct stochastically coupled sample paths, so that collaboration always leads to fewer jobs in the system than noncollaboration.
- A corollary is that collaboration is always better in the popular redundancy(d) model for job replication; numerical evidence for this result was previously provided [9].
- We show that collaboration is stochastically better under the *optimal scheduling policy* when the bipartite matching between jobs and servers has a nested structure.

Finally, we shift our focus to the metric of response time, rather than number in system, using insights gained from our sample-path analysis.

- We provide bounds on the difference in mean steady-state response time between the collaborative and noncollaborative systems.

Notably, these bounds hold for much more general settings than most steady-state results in the literature, because they do not rely on the assumptions required to obtain a product-form stationary distribution.

The remainder of this paper is organized as follows. In Section 2 we provide a brief overview of related work. Section 3 introduces our system model and some preliminary results that we will use throughout the rest of the paper. In Section 4 we study the collaborative and noncollaborative systems in steady-state, presenting new results for the noncollaborative system and comparing the mean number of jobs between the two systems. In Section 5 we study the number of jobs for collaborative and noncollaborative systems on sample paths, and, in particular, when they have exactly the same arrival and potential

service processes. In Section 6, we use an alternative coupling approach to show that, in symmetric systems, the number of jobs is always stochastically smaller under collaboration. We also briefly consider alternative scheduling policies. In Section 7 we return to steady-state and consider the metric of response time. Finally, in Section 8, we conclude.

2 Related Work

For an overview and references on product-form results for collaborative and noncollaborative models, see [17]. The redundancy(d) model, in which copies of an arriving job are sent to d randomly chosen servers, was studied by Gardner et al. [14] for the collaborative (cancel-on-complete, CoC) system, and by Ayesta, Bodas, and Verloop [9,10] for the noncollaborative (cancel-on-start, CoS) system; the latter papers also include numerical comparisons of the steady-state performance under collaboration (CoC) and noncollaboration (CoS). Collaborative and noncollaborative systems have also been considered in the operations management literature; generally these paper study optimal policies for assigning flexible servers to tandem queues, unlike our setting where all servers work in parallel. See [18] and [27] for overviews, and [19] for more recent work.

The above papers, and indeed most of the literature comparing collaborative and noncollaborative models, assume exponential and independent service times, as we do in this paper. In the non-exponential setting, Koole and Righter [23] showed that when all jobs are fully flexible and service times are new-worse-than-used (i.e., remaining service times for jobs that are in process are stochastically larger than jobs that have not started service), the departure process under collaboration is stochastically larger than under noncollaboration. Kim, Righter, and Wolff [22] showed the reverse when service times are new-better-than-used, (i.e., "new" jobs have stochastically longer remaining service times than "used" jobs) and when there are two servers or an infinite number of jobs. Joshi, Soljanin, and Wornell [20,21] did similar comparisons for the fully flexible job case where, for a job to be complete, $k \geq 1$ copies must complete service. There have also been comparisons of stability regions for the collaborative and noncollaborative models when copies of jobs are correlated across servers, and for service disciplines other than FCFS (see [6–8,25,26] and the references therein). Gardner et al. [16] introduce a model in which a job's service times are correlated across servers and study a policy where collaboration only occurs when servers otherwise would be idle. Some of the work in the operations management literature has maintained the exponential service time assumption while allowing non-additive service rates when servers collaborate [4,5,29].

3 The Model and Preliminaries

Our system consists of J classes of jobs and M servers; let $\mathcal{J}$ and $\mathcal{M}$ denote the sets of all job classes and all servers respectively. Class-i jobs arrive to the system as an exogenous process with mean rate λ_i, $i = 1, \ldots, J$. The total arrival rate to the system is $\lambda = \sum_i \lambda_i$. Each arriving job is a class-i job with probability λ_i/λ. Most of our results will apply for this general arrival process; for the steady-state results we present in Section 4, we will require Poisson arrivals. Service times are exponential and i.i.d. with rate μ_m on server m, $m = 1, \ldots, M$; the total service capacity is $\mu = \sum_m \mu_m$. There is a bipartite graph structure indicating which servers can serve which job classes. Let $S_i = \{j : \text{server } j \text{ can serve class } i\}$ denote the set of servers compatible with job class i, and let $C_j = \{i : \text{server } j \text{ can serve class } i\}$ be the set of job classes compatible with server j. Denote by $R(A) = \{i : S_i \subseteq A\}$ the set of job classes i that *require* a server in set $A \subseteq \mathcal{M}$. Let $\mu(A) = \sum_{j \in A} \mu_j$ denote the total rate of all servers in set $A \subseteq \mathcal{M}$, and let $\lambda(B) = \sum_{i \in B} \lambda_i$ denote the total arrival rate of all job classes in set $B \subseteq \mathcal{J}$. For stability, we assume that for any set $A \subseteq \mathcal{M}$, we have $\lambda(R(A)) < \mu(A)$ (following the results of [2,13]).

When a job arrives to the system, it joins a central queue that stores all jobs in the system (including jobs that are in service as well as those that are not yet in service) in their order of arrival. Each server works through this central queue in First-Come First-Served (FCFS) order, skipping over any jobs with which it is not compatible. We consider two different models for service. In the **noncollaborative** model, a job can only enter service on a single server. That is, if the next job in the queue encountered by server j is of some class $i \in C_j$ and the class-i job is already in service on some other server $k \in S_i$, then server j must skip over the class-i job and proceed to the next compatible job. In the **collaborative** model, a job can be in service simultaneously at multiple servers. That is, in our above example, server j will begin processing the class-i job, which is now in service on both servers j and k. If a job is in service on multiple servers, it departs from the system as soon as it completes service on any one of those servers. Because service times are exponentially distributed, this is equivalent to the job being served at a single fast server with additive rate. We will refer to our models as the C (collaborative) and NC (noncollaborative) models.

We adopt the central queue view of the system because this view will be helpful in developing our proofs in the sections that follow. However, we note that the central queue system is equivalent to a distributed system in which each server has its own FCFS queue and when a class-i job arrives to the system it joins the queues at all servers in S_i. This distributed view of the system, often referred to as replication or redundancy, is common in computer systems applications. In the distributed view, the collaborative model is equivalent to "cancel-on-complete" redundancy, and the noncollaborative model is equivalent to "cancel-on-start" redundancy. Because of the equivalence between the central queue and distributed views of the system, all of the results that we obtain in the central queue view also apply to distributed redundancy systems.

For the NC system, the steady-state results that have been derived in earlier literature require an additional policy restriction. Specifically, when an arriving job finds multiple compatible idle servers, it must be assigned to one of those servers according to either (1) the assign-to-longest-idle-server (ALIS) policy, or (2), the random-assignment-to-idle-servers (RAIS) policy, which requires *specific* assignment probabilities. Our steady-state results in Section 4 will require this assumption. However, for most of the paper, our results do not require any particular assignment condition for jobs that find multiple compatible idle servers. Hence, our analytical approach allows us to study a more general version of the NC system than previously has been considered in the literature.

3.1 Preliminary Results

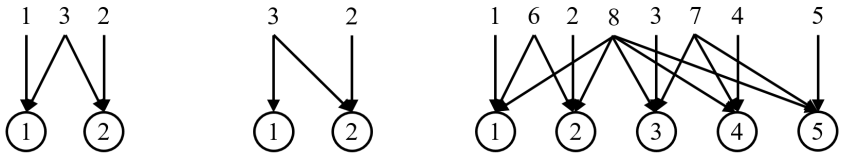
We begin with two preliminary results that allow us to relate performance metrics in the collaborative and noncollaborative systems. By considering sample paths we have the following lemma relating, at any time t, (1) the state of the collaborative *system*, i.e., the classes of jobs present in their order of arrival, and (2) the state of the noncollaborative *queue*, i.e., the classes of jobs present in the queue (not in service) in their order of arrival. Let $\gamma^C(t)$ and $\gamma_Q^{NC}(t)$ denote respectively the system state in the collaborative model and the queue state in the noncollaborative model, where by state we mean the sequence of classes of jobs in order of arrival. Note that the lemma does not require Poisson arrivals—arrivals may be arbitrary as long as they are independent of the state of the system—nor does it require a particular assignment condition for the NC system. The proof involves coupling the arrival sequences and potential service sequences for the C and NC systems and showing that, while all servers in the NC system remain busy, the system state of the former evolves identically to the queue state of the latter; we include the proof in Appendix A for completeness.

Lemma 1 *[1,17] Suppose all the servers are busy at time 0 in NC, and let* $\gamma^C(0) =_{st} \gamma_Q^{NC}(0)$*. Let* τ *be the first time after time 0 at which any server becomes idle in NC. Then* $\{\gamma^C(t)\}_{t=0}^{\tau} =_{st} \{\gamma_Q^{NC}(t)\}_{t=0}^{\tau}$.

A consequence of this lemma is that the queueing time of a job arriving to the noncollaborative system in steady state, conditioned on finding all the servers busy, is the same as the response time of a job arriving to the collaborative system in steady state.

We also have the following lemma for the conditional steady-state distributions for both the C and NC systems assuming Poisson arrivals; the result follows from the product form for the steady-state distributions.

Lemma 2 *[11,17] Assuming Poisson arrivals, the conditional steady-state distribution for the jobs in system for the collaborative model (and for the jobs in queue for the noncollaborative model), given that all servers in* $S \subseteq \mathcal{M}$



(a) The W model (b) The N model (c) A nested system

Fig. 1 The bipartite graph structures for (a) the W model, (b) the N model, and (c) an example of a nested system.

are idle, is the same as the corresponding steady-state distribution in a reduced system consisting of servers $\mathcal{M}\backslash S$ and job classes $\mathcal{J}\backslash\bigcup_{j\in S} C_j$, that is, a system in which the servers in S and all job classes compatible with those servers are removed.

Indeed, the Markov chain for the process, given a subset of servers remains idle, is the same as the Markov chain for a reduced system with those servers removed.

3.2 Specific System Structures

In the sections that follow, we focus on a few specific system structures to develop our initial results before generalizing to a wider range of systems. The **W model** (see Figure 1(a)) consists of $M = 2$ servers and $J = 3$ classes of jobs. Class-1 jobs are compatible with server 1 only ($S_1 = \{1\}$), class-2 jobs are compatible with server 2 only ($S_2 = \{2\}$), and class-3 jobs are compatible with both servers ($S_3 = \{1, 2\}$). When $\lambda_1 = 0$, the W model reduces to a special case called the **N model** (see Figure 1(b)); for consistency of notation between the W and N models we continue to label the job classes as 2 and 3, even though there are only two job classes in the N model. The W and N models are special cases of **nested** systems (see Figure 1(c)). In a nested system, for any two job classes i and j, either (1) $S_i \subset S_j$, (2) $S_j \subset S_i$, or (3) $S_i \cap S_j = \emptyset$. All nested systems have a fully flexible job class J, with $S_J = \{1, \ldots, M\}$. If we remove class J from the system, the system decomposes into two or more non-overlapping nested subsystems, each with its own fully flexible job class. These nested subsystems can be recursively decomposed, in turn, until we arrive at systems consisting of a single job class. In the example shown in Figure 1(b), if the fully flexible class 8 is removed the system decomposes into one nested system consisting of servers 1 and 2 and job classes 1, 2, and 6, and another nested system consisting of servers 3, 4, and 5 and job classes 3, 4, 5, and 7.

4 Steady-State Number of Jobs

If arrivals are Poisson, and, for the noncollaborative case, if the assignment of jobs to idle servers either is according to ALIS (assign to longest idle server), or is random (RAIS, random assignment to idle servers) with particular assignment probabilities, then we obtain a product form for the steady-state probabilities. See [17] for an overview and references. However, these probabilities are for the state described in terms of classes of jobs in order of arrival, and often are difficult to use to obtain formulas for performance measures such as response times and number of jobs for general systems. For an important subclass of bipartite compatibility systems with a nested structure, we do get simple formulas for steady-state number of jobs under collaboration, and these in turn give partial results under noncollaboration, using Lemma 1. In this section we focus on a special case called the W model; we review steady-state results for the number in system in the collaborative system and derive new closed-form results in the noncollaborative system. Note that, because jobs within a class are served FCFS, we can use distributional Little's law to transform our distributional results for per-class number in system to response-time distribution results.

We begin with some notation. Let N_i^X be the number of jobs of class i in steady state in our collaborative model, $i = 1, 2, 3$, $X = C, NC$, let $N^{M/M/1}(\lambda, \mu)$ be the number of jobs in an M/M/1 queueing system with arrival rate λ and service rate μ, and let $N_i^{M/M/1}(\lambda_i, \lambda, \mu)$ be the number of class-i jobs in a multiclass M/M/1 queue with class-i arrival rate λ_i, total arrival rate λ, and service rate μ. Recall that $N^{M/M/1}(\lambda, \mu) \sim geom(1 - \frac{\lambda}{\mu})$ and $N_i^{M/M/1}(\lambda_i, \lambda, \mu) \sim geom(1 - \frac{\lambda_i}{\mu - \lambda + \lambda_i}) \sim N_i^{M/M/1}(\lambda_i, \mu - \lambda + \lambda_i)$, where $Y \sim geom(p)$ represents a geometrically distributed random variable with $P(Y = y) = p(1-p)^y$, $y \in \{0, 1, \dots\}$. Finally, let $I(p)$ be a Bernoulli random variable with success probability p.

Proposition 1 *[13, 15, 17] For the W model in the collaborative system, the per-class number-in-system distributions are as follows:*

$$N_3^C \sim N^{M/M/1}(\lambda_3, \mu - \lambda_1 - \lambda_2) \sim N_3^{M/M/1}(\lambda_3, \lambda, \mu) \sim geom\left(1 - \frac{\lambda_3}{\mu - \lambda + \lambda_3}\right)$$

$$\begin{aligned} N_i^C &\sim N^{M/M/1}(\lambda_i, \mu_i) + I\left(\frac{\lambda_3}{\mu - \lambda + \lambda_3}\right) \cdot N_i^{M/M/1}(\lambda_i, \lambda, \mu) \\ &\sim geom\left(1 - \frac{\lambda_i}{\mu_i}\right) + I\left(\frac{\lambda_3}{\mu - \lambda + \lambda_3}\right) \cdot geom\left(1 - \frac{\lambda_i}{\mu - \lambda + \lambda_i}\right), \qquad i = 1, 2. \end{aligned}$$

The per-class mean numbers in system are:

$$E[N_3^C] = \frac{\lambda_3}{\mu - \lambda}$$
$$E[N_i^C] = \frac{\lambda_i}{\mu_i - \lambda_i} + \frac{\lambda_3}{\mu - \lambda + \lambda_3} \cdot \frac{\lambda_i}{\mu - \lambda}, \qquad i = 1, 2.$$

Of particular interest is the special case of the symmetric W, in which $\mu_1 = \mu_2 = \mu/2$ and $\lambda_1 = \lambda_2 = (\lambda - \lambda_3)/2$.

Corollary 1 *For the symmetric W model in the collaborative system, the class-i numbers in system, $i = 1, 2$, are given by*

$$E[N_1^C] = E[N_2^C] = \frac{\lambda - \lambda_3}{2(\mu - \lambda)} + \frac{\lambda - \lambda_3}{2(\mu - \lambda + \lambda_3)}.$$

We now develop the equations for steady-state number of class-i jobs in the noncollaborative W system under FCFS-ALIS, N_i^{NC}. We note that the expressions also hold under FCFS-RAIS, but the approach to find them under FCFS-RAIS is more complicated because of the need to derive appropriate assignment probabilities for assigning class-3 arrivals to idle server j when both servers are idle. While we give the first exact closed-form expressions for the per-class number-in-system distributions in the noncollaborative W model, the proofs, which we defer to the appendix, follow from results presented in [17].

Let P_j be the probability that server j is idle and the other server is busy, $j = 1, 2$, and let $P_\emptyset$ be the probability that neither server is idle. Let $P_j(i)$ denote the probability that server j is working on a class-i job, $j = 1, 2$, $i = 1, 2, 3$. The per-class numbers in system in the noncollaborative system are given in terms of these probabilities, which are derived in Lemmas 5 and 6 in Appendix B.

Proposition 2 *For the W model in the noncollaborative system, the per-class number-in-system distributions are as follows:*

$$\begin{aligned}
N_3^{NC} &\sim I(P_1(3)) + I(P_2(3)) + I(P_\emptyset) \cdot N_3^C \\
&\sim I(P_1(3)) + I(P_2(3)) + I(P_\emptyset) \cdot geom\left(1 - \frac{\lambda_3}{\mu - \lambda + \lambda_3}\right) \\
N_1^{NC} &\sim I(P_1(1)) + I(P_2) \cdot N^{M/M/1}(\lambda_1, \mu_1) + I(P_\emptyset) \cdot N_1^C \\
&\sim I(\lambda_1/\mu_1) + (I(P_2) + I(P_\emptyset)) \cdot geom\left(1 - \frac{\lambda_1}{\mu_1}\right) \\
&\quad + I(P_\emptyset) \cdot I\left(\frac{\lambda_3}{\mu - \lambda + \lambda_3}\right) \cdot geom\left(1 - \frac{\lambda_1}{\mu - \lambda + \lambda_1}\right) \\
N_2^{NC} &\sim I(\lambda_2/\mu_2) + (I(P_1) + I(P_\emptyset)) \cdot geom\left(1 - \frac{\lambda_2}{\mu_2}\right) \\
&\quad + I(P_\emptyset) \cdot I\left(\frac{\lambda_3}{\mu - \lambda + \lambda_3}\right) \cdot geom\left(1 - \frac{\lambda_2}{\mu - \lambda + \lambda_2}\right)
\end{aligned}$$

The per-class mean numbers in system are:

$$E[N_3^{NC}] = P_1(3) + P_2(3) + P_\emptyset \frac{\lambda_3}{\mu - \lambda}$$
$$E[N_1^{NC}] = \frac{\lambda_1}{\mu_1} + (P_2 + P_\emptyset)\frac{\lambda_1}{\mu_1 - \lambda_1} + P_\emptyset \frac{\lambda_3}{\mu - \lambda + \lambda_3}\frac{\lambda_1}{\mu - \lambda}$$
$$E[N_2^{NC}] = \frac{\lambda_2}{\mu_2} + (P_1 + P_\emptyset)\frac{\lambda_2}{\mu_2 - \lambda_2} + P_\emptyset \frac{\lambda_3}{\mu - \lambda + \lambda_3}\frac{\lambda_2}{\mu - \lambda}.$$

We now consider the special case of the symmetric W model, in which $\mu_1 = \mu_2 = \mu/2$ and $\lambda_1 = \lambda_2 = (\lambda - \lambda_3)/2$. Here the expressions for P_j, and $P_j(i)$ simplify nicely (see Corollary 10 in Appendix B), in turn yielding simple expressions for mean number in system and mean response time.

Corollary 2 *For the symmetric W model in the noncollaborative system, the class-i mean numbers in system, $i = 1, 2, 3$, are given by*

$$E[N_1^{NC}] = E[N_2^{NC}] = \frac{\lambda - \lambda_3}{2}\left(\frac{1}{\mu - \lambda} - \frac{1}{\mu + \lambda_3} + \frac{1}{\mu} + \frac{1}{\mu - \lambda + \lambda_3}\right)$$
$$E[N_3^{NC}] = \frac{\lambda_3}{\mu - \lambda} + \frac{\lambda_3(\mu - \lambda + \lambda_3)}{\mu(\mu + \lambda_3)}.$$

4.1 C versus NC Comparison

We are now ready to compare the per-class and overall mean number in system in the C and NC systems. We begin with the symmetric W model, where our closed-form expressions for mean numbers of jobs in both systems yield simple results for the difference between the two systems:

$$E[N_1^C] - E[N_1^{NC}] = E[N_2^C] - E[N_2^{NC}] = -\frac{2\lambda_3(\lambda - \lambda_3)}{\mu(\mu + \lambda_3)},$$
$$E[N_3^C] - E[N_3^{NC}] = -\frac{\lambda_3(\mu - \lambda + \lambda_3)}{\mu(\mu + \lambda_3)},$$
$$E[N^C] - E[N^{NC}] = -\frac{\lambda_3}{\mu + \lambda_3}.$$

All of these differences are strictly negative as long as there are flexible jobs, $\lambda_3 > 0$, meaning that all classes, and the system overall, have better performance in the collaborative system than in the noncollaborative system.

Figure 2 shows the difference in the mean number of jobs in the C and NC systems in the symmetric W model with $\mu = 2$. The only free system parameters in the symmetric W are λ and $p_3 = \lambda_3/\lambda$; we vary both of these. We observe that the magnitude of the difference in mean number of jobs between the two systems is at most 1/2; this also follows from our analytical results above, with $\lambda_3 = \lambda \to \mu = 2$. This maximum difference occurs when $p_3 = 1$, i.e., when all jobs are class-3. In this case, the C system collapses into

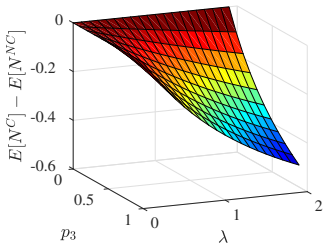
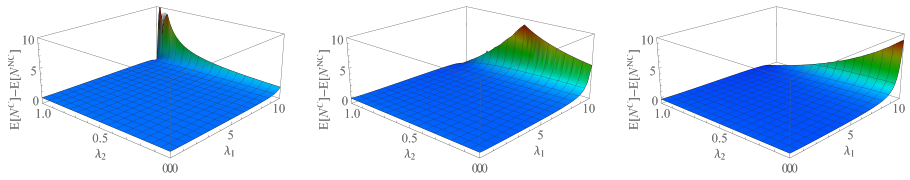


Fig. 2 $E[N^C] - E[N^{NC}]$ as a function of λ and p_3 (the fraction of jobs that are class-3) in the symmetric W model. Here $\lambda_1 = \lambda_2$ and $\mu_1 = \mu_2 = 1$.



(a) $\lambda_3 = 0.1$ (b) $\lambda_3 = 0.5$ (c) $\lambda_3 = 1$

Fig. 3 $E[N^C] - E[N^{NC}]$ as a function of λ_1 and λ_2 in the asymmetric W model. Here $\mu_1 = 1$, $\mu_2 = 10$, and we consider three different values of λ_3.

a single M/M/1 system with service rate μ, whereas the NC system becomes an M/M/2 with each server having rate $\mu/2$. At the other extreme, when $p_3 = 0$ the two systems are identical; both become two independent M/M/1 queues, one consisting of server 1 and class-1 jobs, and the other consisting of server 2 and class-2 jobs. In between these two extremes, the magnitude of the difference in mean number of jobs increases with p_3. In Section 6, we will show a stronger result: not only is collaboration better for the mean number of jobs, but, in symmetric systems, collaboration also is better on stochastically coupled sample paths (Theorem 4).

The story is quite different in the asymmetric W model. Because the expressions are long, we use our analytical results to present a numerical comparison. Figure 3 shows the difference in mean nunber of jobs between the C and NC asymmetric W systems, $E[N^C] - E[N^{NC}]$. We observe that the performance under collaboration is worse than under noncollaboration when the dedicated jobs are very unbalanced, i.e., when one dedicated class has very high load and the other dedicated class has very low load. How high or low the per-class loads need to be in order for C to be worse depends on λ_3 and on the relative server rates. The intuition here is that if one server is highly loaded with dedicated jobs (in the cases shown in Figure 3, server 2 and class 2), then the class-2 jobs are significantly harmed if class-3 jobs enter service on server 2. While this is a potential problem under both C and NC, when the class-2 load is high it will be rare for a class-3 job to enter service at server 2 before server 1. In the noncollaborative model, this means that the class-3 job will not run on server 2, whereas in the collaborative model the class-3 job could still enter service

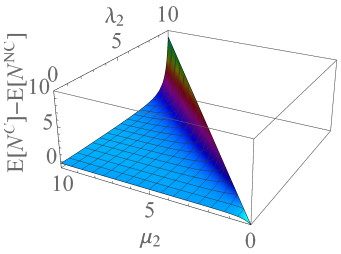


Fig. 4 Comparing $E[N]$ for C and NC in the asymmetric W. Here $\lambda_3 = \mu_1 = 1$, $\lambda_1 = 0$, and λ_2 and μ_2 vary.

on server 2, thereby hurting the class-2 jobs. In the next section, we will see how extra jobs can accumulate in the collaborative model on a sample path.

One of the cases in Figure 3 that exhibits the largest difference in mean number of jobs between the C and NC systems occurs when $\lambda_3 = \mu_1$, $\lambda_1 = 0$, and λ_2 is high. We now look at this special case in more detail and explore the effect of changing λ_2 and μ_2. Note that, because $\lambda_1 = 0$, we have now moved from the W model to the N model. Figure 4 shows our results; again, higher values indicate that the C system is worse than the NC system. The difference between C and NC is largest when $\lambda_2 \approx \mu_2$; a messy algebraic argument (with the aid of Mathematica) shows that as $\lambda_2 \to \mu_2$, this difference approaches $\frac{\mu_2}{\mu_1} - 1$, for $\mu_2 > 1$.

Proposition 3 *Let* $\lambda_3 = \mu_1$ *and* $\lambda_1 = 0$. *As* $\lambda_2 \to \mu_2$, $E[N^C] - E[N^{NC}] \to \frac{\mu_2}{\mu_1} - 1$.

Proposition 3 tells us that, by increasing $\lambda_2 \approx \mu_2$, the difference in the mean number of jobs in the C and NC systems can become arbitrarily high. Thus, for highly asymmetric systems, the cost of collaboration can be significant. As we will see, this is consistent with our sample-path number-of-jobs results. We will show that the benefit of collaboration is strictly bounded, while the cost can be arbitrarily large on a sample path (Section 5).

5 General Sample-Path Bounds

In this section we consider the difference between $N^C(t)$ and $N^{NC}(t)$, where $N^x(t)$ is the total number of jobs in system x, $x = C, NC$, at time t. We study this difference on sample paths with coupled arrivals and potential service completions, where a potential service completion (which occurs according to a Poisson process at rate $\mu = \sum \mu_i$) results in an actual job departure if the server is not idle. For our sample-path results, we can have a general exogenous arrival process, as long as each arrival is independently of class-i with probability λ_i/λ, and, for Theorem 2 below, we also assume that times between arrivals have nonzero probability of being arbitrarily small. We show that for coupled

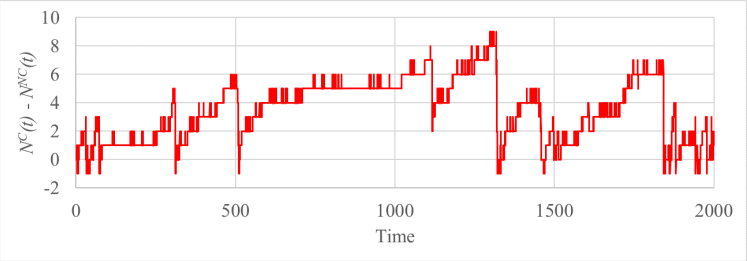


Fig. 5 Difference in the number of jobs in the C and NC systems over time. For the sample path shown, $\mu_1 = 1$, $\mu_2 = 10$, $\lambda_1 = 0$, $\lambda_2 = 9.5$, and $\lambda_3 = 1$.

arrival and potential service processes, the benefit of collaboration is always bounded with probability one while the cost is not bounded. In particular, on coupled sample paths, $\{N^{NC}(t) - N^{C}(t)\}_{t=0}^{\infty} \leq M$ with probability 1, where M is the number of servers, but, for any $K, t \geq 0$, $P(N^C(t) - N^{NC}(t) > K) > 0$. This behavior is illustrated in Figure 5 for an N model (which is equivalent to the W model with $\lambda_1 = 0$). The general intuition is that more flexible jobs can "block" later arriving less flexible jobs in collaborative systems, whereas later arriving jobs can "pass" earlier arrivals in noncollaborative systems, sometimes significantly reducing the number of jobs in those systems. At the end of the section we show a tighter bound on the benefit of collaboration for the W model.

We start with the proof of the upper bound of M on the benefit of collaboration, using the insight from Lemma 1 (which extends easily to the general arrivals case) that, when all servers are busy, the job queue in the noncollaborative system has the same coupled sample path as the system queue in the collaborative system (in addition to M extra jobs in service for the NC system). Let us therefore define the NCB (noncollaborative busy) model as a modified version of the noncollaborative, NC, model in which we keep all the servers busy by assigning a server that becomes idle and that does not find a waiting compatible job a compatible "dummy job." For $x = C, NC, NCB$, let $N^x(t)$ ($N_Q^x(t)$) be the total number of jobs in system x (in the queue for system x) at time t, and define $N_i^x(t)$ ($N_{Qi}^x(t)$) correspondingly for class i jobs. Let $\gamma_Q^x(t)$ denote the detailed queue state of system x at time t; the detailed state is of the form $(c_1, c_2, ..., c_n)$ for some $n \geq 0$ and tracks the classes of all jobs in the queue (i.e., not including the jobs in service) in the order in which they arrived. Let $\gamma_Q^x(t) \prec \gamma_Q^y(t)$ mean that the detailed queue state in system y at time t, $\gamma_Q^y(t)$, includes $\gamma_Q^x(t)$ as well as some number of interleaved "extra" jobs (so $N_{Qi}^x(t) \leq N_{Qi}^y(t)$ for all i). We will assume $\gamma_Q^{NC}(0) \prec \gamma_Q^{NCB}(0)$. Note that $N^{NCB}(t) - N_Q^{NCB}(t) = M \geq N^{NC}(t) - N_Q^{NC}(t)$.

Lemma 3 *Given $\gamma_Q^{NC}(0) \prec \gamma_Q^{NCB}(0)$, and coupled arrival and potential service completion processes, $\{\gamma_Q^{NC}(t)\}_{t=0}^{\infty} \prec \{\gamma_Q^{NCB}(t)\}_{t=0}^{\infty}$ with probability 1, so $\{(N_{Q1}^{NC}(t), N_{Q2}^{NC}(t), ..., N_{QJ}^{NC}(t))\}_{t=0}^{\infty} \leq \{(N_{Q1}^{NCB}(t), N_{Q2}^{NCB}(t), ..., N_{QJ}^{NCB}(t))\}_{t=0}^{\infty}$ with probability 1.*

Proof Our proof will be by induction on t; the result holds at time $t = 0$. Suppose $\gamma_Q^{NC}(t) \prec \gamma_Q^{NCB}(t)$ at some time $t \geq 0$. Let τ be the time of the next event after t. The system state does not change between time t and time τ, so the result continues to hold up until time τ. We will show that $\gamma_Q^{NC}(\tau) \prec \gamma_Q^{NCB}(\tau)$.

If the event at time τ is an arrival, then the arriving job either is appended to the end of the queue for both systems, so $\gamma_Q^{NC}(\tau) \prec \gamma_Q^{NCB}(\tau)$, or it enters service and leaves the queue immediately under NC but becomes an extra job under NCB, so $\gamma_Q^{NC}(\tau) \prec \gamma_Q^{NCB}(\tau)$.

If the event at time τ is a potential service completion and the same job, with class $c_j \in \gamma_Q^{NC}(t)$ for some j, is assigned to a server and leaves the queue in both systems, again $\gamma_Q^{NC}(\tau) \prec \gamma_Q^{NCB}(\tau)$.

If the event is a potential service completion and an extra job enters service under NCB but the server idles under NC because there is no compatible job, we still have $\gamma_Q^{NC}(\tau) \prec \gamma_Q^{NCB}(\tau)$.

If the event is a potential service completion and different jobs enter service in the two systems, then it must be that under NC a job with class $c_j \in \gamma_Q^{NC}(t)$ enters service, while under NCB an extra job of some class c enters service, where this class-c job arrived before the class c_j job. Then the extra class-c job disappears from the state under NCB and the class c_j job becomes an extra job. Again, $\gamma_Q^{NC}(\tau) \prec \gamma_Q^{NCB}(\tau)$.

From Lemma 1, we can couple the NCB and C processes, by starting with M jobs in service in the NCB system and $\gamma_Q^{NCB}(0) = \gamma^C(0)$, so that

$$\{(N_{Q1}^{NCB}(t), N_{Q2}^{NCB}(t), ..., N_{QJ}^{NCB}(t))\}_{t=0}^{\infty} = \{(N_1^C(t), N_2^C(t), ..., N_J^C(t))\}_{t=0}^{\infty} \text{ wp 1.}$$

Now suppose the collaborative and noncollaborative models have the same detailed states at time 0, $\gamma^{NC}(0) = \gamma^C(0)$, and let $\gamma^{NCB}(0)$ have M busy servers and $\gamma_Q^{NCB}(0) = \gamma^{NC}(0)$, so $\gamma_Q^{NC}(0) \prec \gamma_Q^{NCB}(0)$. Then, for coupled arrival and service processes, from the lemma above and Lemma 1,

$$\{N^{NC}(t)\}_{t=0}^{\infty} \leq \{N_Q^{NC}(t)+M\}_{t=0}^{\infty} \leq \{N_Q^{NCB}(t)+M\}_{t=0}^{\infty} = \{N^C(t)+M\}_{t=0}^{\infty} \text{ wp 1,}$$

That is, we have the following, where the argument for the second claim is similar because $|S_i|$ is an upper bound on the number of class i jobs in service for the NC system.

Theorem 1 *Given the same initial states, and coupled arrival and service processes,*

$$\left\{N^{NC}(t) - N^C(t)\right\}_{t=0}^{\infty} \leq M \text{ wp 1,}$$
$$\left\{N_i^{NC}(t) - N_i^C(t)\right\}_{t=0}^{\infty} \leq |S_i| \text{ wp 1.}$$

We now show that the cost of collaboration can grow arbitrarily large along sample paths for systems with coupled arrivals and services.

Theorem 2 *For any given bipartite matching structure that includes a pair of classes i and j with $S_i \cap S_j \neq \emptyset$, with coupled arrival and potential service processes, $\left\{N^C(t) - N^{NC}(t)\right\}_{t=0}^{\infty}$ can be arbitrarily large.*

To prove Theorem 2, we begin by studying the smallest nontrivial bipartite matching model, the N model (see Figure 1(b)). The full system state at time t is $\gamma^x(t)$, $x = C, NC$, where γ tracks the classes of all jobs in the system in their arrival order; note that this state includes jobs that are in service as well as jobs that are waiting in the queue for both the C and NC systems. We use a subscript to indicate the server to which a job has been assigned; this is required to avoid ambiguity in the NC system. For example, $\gamma^{NC}(t) = (3_2, 2)$ means that there are two jobs in the system, the first is a class-3 job that is in service on server 2, and the second is a class-2 job that is waiting in the queue. If $\gamma^{NC}(t) = (3_1, 2_2)$, then both jobs are in service (and, in terms of the future evolution of the system, the state is equivalent to $(2_2, 3_1)$).

We are now ready to show that, for the N model, system C can have arbitrarily many more jobs of either class than system NC, on sample paths with coupled arrivals and potential services.

Lemma 4 *For the N model, with coupled initial conditions and arrival and potential service processes, $\left\{N^C(t) - N^{NC}(t)\right\}_{t=0}^{\infty}$ and $\left\{N_i^C(t) - N_i^{NC}(t)\right\}_{t=0}^{\infty}$, $i = 2, 3$, all can be arbitrarily large. That is, for any $K, t \geq 0$, $P(N^{NC}(t) - N^C(t) > K) > 0$.*

Proof We provide a sample path, with coupled arrivals and potential service completions, that has nonzero probability of the difference exceeding K by time t. Let the number of events in the coupled arrival and service processes by time t be κ, where κ will be determined later. Note that the probability of κ events in $(0, t)$ is greater than 0 because of our assumption that interarrival times can be arbitrarily small.

Start with the same set of 4 jobs in the two systems: $\gamma^C(0) = (2_2, 3_1, 2, 2)$ and $\gamma^{NC}(0) = (2_2, 3_1, 2, 2)$. In the following, for notational simplicity, we redefine time in the argument of γ^x so that time i denotes the time of the i^{th} event (either an arrival or a potential service completion). Thus, we have redefined $\gamma^x(t)$ to be $\gamma^x(\kappa)$. Let the coupled event sequence consist of j service completions at server 2, followed by one service completion at server 1, one class-3 arrival, and $j + 1$ class-2 arrivals, $j = 2, 3, \ldots$. The beginning of this sample path proceeds as follows. There are three service completions, on servers 2, 2, and 1, so $\gamma^C(3) = (2_2, 2)$ and $\gamma^{NC}(3) = (2_2)$. Next, there are four arrivals, of classes 3, 2, 2, and 2, so $\gamma^C(7) = (2_2, 2, 3_1, 2, 2, 2)$ and $\gamma^{NC}(7) = (2_2, 3_1, 2, 2, 2)$. Next, there are four service completions on servers 2, 2, 2, and 1, so $\gamma^C(11) = (2_2, 2, 2)$ and $\gamma^{NC}(11) = (2_2)$. At this point we can see the pattern emerging: for each cycle j, after the j service completions, there is 1 class-2 job in system NC and there are j class-2 jobs in system C. We now let κ be large enough so that we have a build-up of $K + 1$ class-2 jobs in system C (and 1 class-2 job in system NC).

Note that C could also have arbitrarily more class-3 jobs than NC. Using the above initial state and the above sample path up to $j = k$, we reach a state consisting of k class-2 jobs in system C and one class-2 job in system NC. We then proceed with a coupled sample path consisting of $k - 1$ class-3 arrivals, followed by $k - 1$ service completions on server 2. At this point, the state in system C consists of one class-2 job followed by $k-1$ class-3 jobs, whereas the state in system NC consists of a single class-3 job that is in service at server 1.

We can use this same sample path for any subsystem that has the same structure as the N model. That is, if a system contains a pair of classes i and j and a pair of servers r and s such that $r \in S_i$, $r \notin S_j$, $s \in S_i$, and $s \in S_j$, then the above argument applies after we relabel i as class-3, j as class-2, r as server 1, and s as server 2. This is quite general: all that is required to have such a subsystem is that there exists a pair of job classes i and j such that $S_i \cap S_j \neq \emptyset$. This holds in *any* nondegenerate bipartite matching structure, which completes the proof of Theorem 2.

5.1 The W model and a tighter bound

The bound in Theorem 1 tells us that, for any bipartite matching structure, we can couple the C and NC systems so that the NC system has at most M more jobs than the C system. We now refine that bound for the W model (see Figure 1(a); recall that in the W model we have 3 classes of jobs and $M = 2$ servers, where class i jobs can only be served by server i, $i = 1, 2$, and class 3 jobs are compatible with both servers. For the W model, we will show that with coupled arrival and service processes, and the same initial state, the NC system can have at most one more job than the C system, and, if it does have such an extra job, then, excluding that job, the C system has at least as many jobs of each class as the NC system. This result generalizes a similar result for the N model given by Adan et al. [1]

Assume that $N_i^C(0) = N_i^{NC}(0)$ for $i = 1, 2, 3$, and that arrivals and potential service completions are coupled for both systems. Because our systems are FCFS, we can make the following observations about the coupled systems C and NC.

1. If both systems have the same number of class i jobs, then these jobs are the same set of jobs, i.e., they have the same arrival times in both systems.
2. If system x has fewer class i jobs than system x', $(x, x') = (C, NC)$ or (NC, C), then the additional class i jobs in x' must have arrived earlier than the other class i jobs, and the other class i jobs have the same arrival times in both systems.
3. If a class 3 job is present in system C, then all jobs, regardless of class, that arrived after it must also be present in C; they cannot start service until after the 3 job completes service. The same is not necessarily true in system NC.

Theorem 3 *For the W model, with coupled arrival and service processes and initial states, we have that with probability 1,* $\{N^C(t)\}_{t=0}^{\infty} \geq \{N^{NC}(t)-1\}_{t=0}^{\infty}$, $\{N_i^C(t)\}_{t=0}^{\infty} \geq \{N_i^{NC}(t)-1\}_{t=0}^{\infty}$ *for* $i = 1, 2, 3$, *and* $\{N_i^C(t)+N_3^C(t)\}_{t=0}^{\infty} \geq \{N_i^{NC}(t)+N_3^{NC}(t)-1\}_{t=0}^{\infty}$ *for* $i = 1, 2$.

We outline the key part of the argument here; the full proof is in Appendix C. The proof is by induction on the time t. If at time t NC has one more job than C, call it the tagged job, then from our observations above and the induction hypothesis, the tagged job must have arrived before any of the jobs in C, and other than the tagged job, all jobs and their arrivals are the same in both C and NC. This means that the tagged job must be in service in NC, and the other server must be serving the same job or class of job in both C and NC (if any). Therefore, if there is a departure in C, there must also be a departure in NC, so the difference in the numbers of jobs under C and NC cannot increase beyond the current (at time t) difference of 1.

6 The Benefit of Collaboration

We will generalize our results for mean steady-state number in system for the symmetric W model to show that collaboration is beneficial in more general *symmetric* systems in a stronger, stochastic sense. A system may be symmetric from the perspective of the servers or from the perspective of the job classes. Symmetry from the servers' perspective means that the servers are stochastically indistinguishable with respect to their service processes, the sets of job classes with which they are compatible, and the arrival processes of these job classes. The symmetric W model ($\lambda_1 = \lambda_2$ but λ_3 may be different, $\mu_1 = \mu_2$) exhibits this type of symmetry. Symmetry from the job classes' perspective means that the job classes are stochastically indistinguishable with respect to their arrival processes, the servers with which they are compatible, and the service processes of these servers. We will see that a symmetric M model ($\lambda_1 = \lambda_2$, $\mu_1 = \mu_3$ but μ_2 for the shared server may be different) exhibits this type of symmetry. The redundancy(d) structure, with all arrival and service rates the same, and commonly used in large-scale systems, exhibits both types of symmetry. Another example that satisfies both types of symmetry is the nearest neighbor model, common in bike share applications, where class i jobs can be served at server i or server $i+1$, $M{=}J$ with server $M+1$ defined to be server 1, and all arrival rates and service rates are the same.

We know from Theorem 2 that on sample paths that are directly coupled, so that the arrival process for each class and potential service process for each server are the same for both the C and the NC systems, we will have a nonzero probability of the number under C being arbitrarily larger. That is, even for symmetric systems, we cannot use a direct coupling to show that the job process in the C system is stochastically smaller than that of the NC system. To get such an ordering, in this section we use a novel coupling that leverages the symmetry in the system. To develop intuition for the distinction across

different couplings, first consider two W models, A and B, with $\mu_i^A = \mu_i^B$, $i = 1, 2$ and $\lambda_3^A = \lambda_3^B$, but $\lambda_1^A = 2$, $\lambda_2^A = 1$, $\lambda_1^B = 1$, $\lambda_2^B = 3$. Then, if we do the direct coupling of all arrival and service processes, the total number of jobs in model A can exceed the total number of jobs in model B, and vice versa, by an arbitrary amount with non-zero probability (i.e., in the sense of Lemma 4). However, if we relabel the job classes in model B, so that job class 1 is relabeled to be job class 2 and vice versa, and then couple the arrival and service processes, we will have that the number in model B is always at least as large as the number in model A, with probability 1. For our coupled arrivals of class 1 jobs, we assume potential arrivals at rate 3, but, with probability 1/3 there is no actual arrival in model A. For our arguments in this section, we will do a similar relabeling of classes of jobs, where, here, classes may be relabeled multiple times.

The sample path arguments that we use in this section follow a different approach than the standard coupling used in Section 5. In Section 5, we allowed both the C and NC systems to follow the same sequence of arrivals and potential service completions for the entire time horizon; this is the typical approach used in sample-path arguments. In this section, we use a slightly different approach: We will directly couple the *timing* of all arrivals and potential service completions, but we will periodically leverage the symmetry of the systems and policies under consideration to *relabel* job classes or servers in one of the two systems while maintaining the correct marginal distributions. At coupled service completion events, the symmetry property allows us either to relabel servers and/or job classes, or to resample the entire system state, to arrive at a stochastically equivalent state from which we can proceed. This approach allows us to obtain stochastic orderings between the C and NC systems, which are stronger than the steady-state comparisons of Section 4 and more general in terms of system structure.

We first investigate systems that are extensions of the W model, and then we consider the redundancy(d) system. We also show that collaboration is beneficial when we follow an optimal scheduling policy for nested, possibly asymmetric, systems. Our results hold for general arrival processes, as long as each arrival is of class i with probability λ_i/λ independently of all else, and the arrival process is independent of the state of the system and the collaboration policy.

6.1 W Model and Variants

We begin by considering the symmetric W model, in which $\lambda_1 = \lambda_2$ and $\mu_1 = \mu_2$, and which is symmetric from the servers' perspective. We have already shown that collaboration is better in terms of steady-state mean number of jobs. We now show that it is better in a stronger sample-path sense, and then generalize our results to several related system structures.

Theorem 4 *For the symmetric W model,* $\{N^C(t)\}_{t=0}^{\infty} \leq_{st} \{N^{NC}(t)\}_{t=0}^{\infty}$.

Proof We consider *admissible policies*, under which (1) each server maintains a FCFS service discipline, (2) if the first job in the system that is compatible with server 1 (respectively, server 2) is a class-3 job, either it can be assigned to server 1 (respectively, server 2) or it can be relabeled as a class-2 job (respectively, class-1 job), and (3) servers may idle even if there are compatible jobs in the queue. Our decisions regarding idling and collaboration are not permitted to depend on the state of the system or on the identity of the server; i.e., admissible policies preserve the symmetry of the system. We will show that any policy that does not collaborate when a class-3 job is at the head of the queue is stochastically dominated by a policy that collaborates at that time. We then argue that idling when there is work in the queue is also suboptimal.

Let π be an arbitrary admissible policy, and let $N(t)$ be the number of jobs in the system at time t under π. Suppose that at some time s the first job in the system is a class-3 job, which we will denote job a, and suppose (without loss of generality) that π assigns job a to server 1 only. Consider an alternative system operating under admissible policy π', with corresponding $N'(t)$, and let the two systems have coupled arrival processes and potential service processes. Let π' agree with π before time s, and suppose that π' assigns job a to both servers at time s. We will show that we can construct π' on (s, ∞) such that $\{N'(t)\}_{t=0}^{\infty} \leq \{N(t)\}_{t=0}^{\infty}$ w.p. 1. Note that this result holds for $t \in [0, s)$ because π and π' behave identically over this interval.

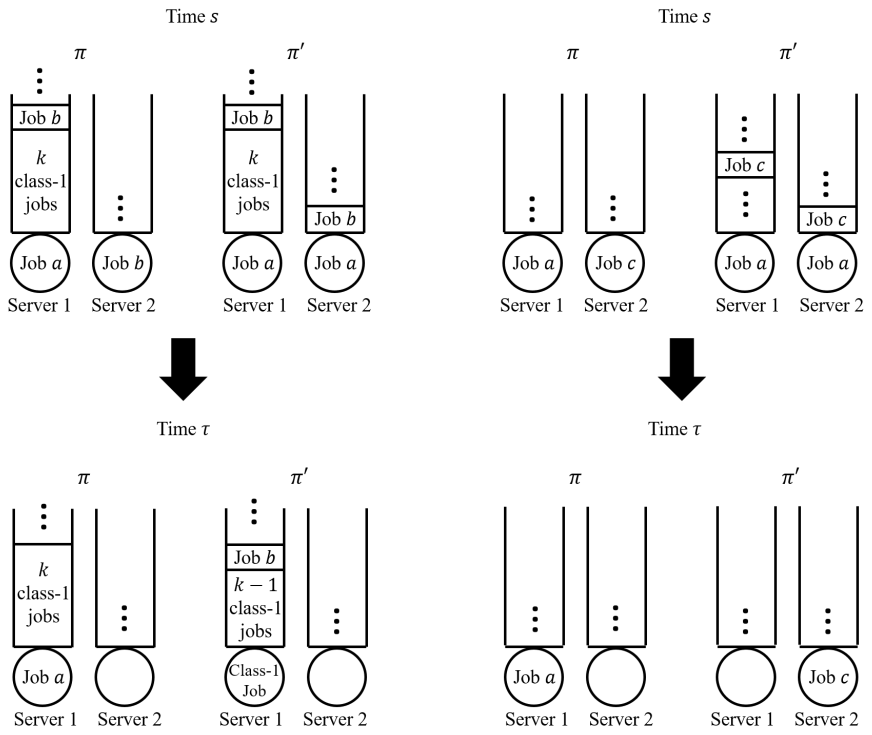
Let τ denote the time of the next event (arrival or potential service completion). If an arrival occurs at time τ, then $N(\tau) = N(s)+1 = N'(s)+1 = N'(\tau)$, so the result holds. Now suppose that τ is a potential service completion time. We will consider four cases for this service completion.

Case 1: potential service completion at server 1. Then job a departs both systems at time τ, and the two systems will be in the same state. Letting π' agree with π from time τ on, we have $\{N'(t)\}_{t=0}^{\infty} = \{N(t)\}_{t=0}^{\infty}$.

Case 2: potential service completion at server 2, server 2 is idle under π. Then job a will complete at time τ under π' but no job will complete under π. Let π' agree with π from time τ on, except that π' idles server 1 until job a completes under π; the system under π' has one fewer job than the system under π during this time. When job a completes under π, the two systems again have the same state, and let the two policies agree from that point on. Then $\{N'(t)\}_{t=0}^{\infty} \leq \{N(t)\}_{t=0}^{\infty}$.

Case 3: potential service completion at server 2, class-3 completion under π (see Figure 6(a)). Let job b be the class-3 job that completes at server 2 at time τ under π. Note between jobs a and b, there may be some number $k \geq 0$ of class-1 jobs that arrived, and no class-2 jobs. Also note that job a completes at server 2 under π', while under π job a remains and is treated as a class-1 job. Let π' treat job b as a class-1 job. Then, just after time τ, the two systems effectively have the same state, with $k+1$ class-1 jobs at the head of the queue (including one in service). Again, letting the two policies otherwise agree from time τ on, $\{N'(t)\}_{t=0}^{\infty} = \{N(t)\}_{t=0}^{\infty}$.

Case 4: potential service completion at server 2, class-2 completion under π (see Figure 6(b)). Let job c be the class-2 job that completes



(a) Case 3: class-3 completion at server 2 under π

(b) Case 4: class-2 completion at server 2 under π

Fig. 6 Illustrations of (a) case 3 and (b) case 4 of Theorem 4. Vertical ellipses indicate that we have not conditioned on the job(s) presented in a section of the queue.

at server 2 at time τ under π, and note that job a completes at server 2 under π'. Then, just after time τ, the two systems have the same state except that under π job a is present (and in service on server 1) and job c is not present, whereas under π' job a is not present and job c is. Observe that, at this point, π treats job a as a class-1 job from our definition of admissible policies. Let π' assign job c to server 2. We now relabel the servers under π' so that server 1 is now labeled server 2, and vice versa. We can do this without loss of generality because of the symmetry of our system and our policies, and because we have not conditioned on the state of the queue beyond the presence of job a under π and of job c under π'. Again letting π' agree with π from time τ on, we have $\{N'(t)\}_{t=0}^{\infty} = \{N(t)\}_{t=0}^{\infty}$.

At this point we have shown that π', which always collaborates when the first job in the system is a class-3 job, has stochastically fewer jobs than π at all moments in time. Note that π' may leave a server idle while there is a compatible job in the system. A similar argument shows that for any policy that idles at some time s a policy that does not idle at s can be constructed with a smaller number in system process. Repeating the collaboration and

nonidling arguments at every moment at which policy π either idles or does not collaborate yields the result of the theorem.

We now generalize our result for the symmetric W model to several related models, most of which generalize to arbitrarily many servers. The proofs all follow the same approach, with some additional cases required for each of the related models; we defer the details of these proofs to Appendix D.

Consider a model with M servers and $M+1$ customer classes, where class $M+1$ is fully flexible, so it can be served by any server, while classes $i = 1, ..., M$ are dedicated classes, i.e., $S_i = \{i\}$. We call this the extended W model. In the symmetric extended W model, we also assume that $\lambda_i = \lambda_j$ for $i, j = 1, ..., M$, and $\mu_i = \mu_j$ for $i, j = 1, ..., M$; that is, all non-flexible job classes have the same arrival rate, and all servers have the same rate.

Corollary 3 *For the symmetric extended W model,* $\{N^C(t)\}_{t=0}^{\infty} \leq_{st} \{N^{NC}(t)\}_{t=0}^{\infty}$.

We can also embed the extended W model in a larger model. Define servers $1, ..., M$, and classes $1, ..., M+1$ as an extended W subsystem, and suppose there are other servers $M+1, ..., M'$ and classes $M+2, ..., J$, such that the extra classes are either fully flexible classes for the extended W subsystem or they are incompatible with all the servers in the subsystem. That is, for $i = M+2, ..., J$, either $S_i \supset \{1, ..., M\}$ or $S_i \cap \{1, ..., M\} = \emptyset$. Now we consider only collaboration within the extended W subsystem. We let the policy for the additional servers and classes be arbitrary, and assume that the C and NC systems follow the same policy outside of the extended W subsystem.

Corollary 4 *For a symmetric extended W subsystem embedded in a larger system, and considering collaboration within the subsystem,* $\{N^C(t)\}_{t=0}^{\infty} \leq_{st} \{N^{NC}(t)\}_{t=0}^{\infty}$.

Another generalization of the symmetric W system is the nearest neighbor model. There are $J = 2M$ classes of jobs, classes $i = 1, 2, ..., M$ are dedicated classes, with $S_i = \{i\}$, and classes $i = M+1, ..., 2M$ are flexible classes, with $S_i = \{i, i+1 \bmod M\}$. In the symmetric version, all servers have the same speed, the dedicated job classes all have the same arrival rate, and the flexible job classes all have the same arrival rate.

Corollary 5 *For the symmetric nearest neighbor model,* $\{N^C(t)\}_{t=0}^{\infty} \leq_{st} \{N^{NC}(t)\}_{t=0}^{\infty}$.

Finally, we consider the symmetric M model, which consists of three servers and two job classes, where class-i jobs, $i = 1, 2$, are compatible with server i and server 3. In the symmetric version we let $\lambda_1 = \lambda_2$ and $\mu_1 = \mu_2$. Because this system is symmetric with respect to job classes, rather than with respect to servers, our coupling argument will involve at times relabeling the job classes, rather than the servers. Apart from recoupling from the jobs' perspective rather than the servers' perspective, the argument is again similar to the proof of Theorem 4.

Corollary 6 *For the symmetric M model,* $\{N^C(t)\}_{t=0}^{\infty} \leq_{st} \{N^{NC}(t)\}_{t=0}^{\infty}$.

6.2 Redundancy(d) Model

We now consider a fully symmetric system where all servers and all job classes have the same rates, and where the compatibility graph is based on "power-of-d" routing, under which each arrival is compatible with $d \geq 1$ randomly and uniformly chosen servers. In the redundancy(d) system the bipartite matching graph has degree d for all job class nodes, and there are $\binom{M}{d}$ job classes. Note that all job classes are stochastically indistinguishable, as are all servers. Also, because both our collaborative and noncollaborative policies follow FCFS, which does not distinguish among job classes or servers, the classes and servers remain stochastically indistinguishable as the system evolves. That is, at any time t in both the C and NC systems, any job present at time t is equally likely to have any class, all servers are equally likely to be busy, and if a server j is busy under the policy, the job it is serving is equally likely to be of any compatible job class. Let $N^C(t)$ and $N^{NC}(t)$ be the total number of jobs at time t in the C and NC systems respectively.

Theorem 5 *For the redundancy(d) system, if the collaborative and noncollaborative systems start in the same state at time 0, then* $\{N^C(t)\}_{t=0}^{\infty} \leq_{st} \{N^{NC}(t)\}_{t=0}^{\infty}$.

Proof The proof will proceed by coupling and forward induction; in fact, we will show a stronger result. We begin by assuming that at some arbitrary time t, $N^C(t) \leq N^{NC}(t)$ w.p. 1. We partition the set of $N^{NC}(t)$ jobs in the NC system into $N_1^{NC}(t) = N^C(t)$ "basic" jobs and $N_2^{NC}(t) = N^{NC}(t) - N^C(t) \geq 0$ "extra" jobs, and let $\mathcal{S}^C(t)$, $\mathcal{S}_1^{NC}(t)$, and $\mathcal{S}_2^{NC}(t)$ be the set of servers that are compatible with the corresponding sets of jobs. We further assume that at time t, $\mathcal{S}_1^{NC}(t) = \mathcal{S}^C(t)$ w.p. 1 (in addition to $N^C(t) \leq N^{NC}(t)$ w.p. 1); we will show that, with coupled arrival and potential service completion times, both conditions hold at time τ w.p. 1, where τ is the time of the next event (arrival or potential service completion).

For the base case, observe that by starting in the same state at time 0 we have $N^C(0) \leq N^{NC}(0)$ and $\mathcal{S}_1^{NC}(0) = \mathcal{S}^C(0)$ w.p. 1.

We now proceed to the inductive step. It is important to emphasize that at time τ^- (i.e., just before time τ), we assume *only* the aggregate coupling, that $N^C(\tau^-) = N^C(t) \leq N^{NC}(\tau^-) = N^{NC}(t)$ and the NC jobs are partitioned such that $\mathcal{S}_1^{NC}(\tau^-) = \mathcal{S}_1^{NC}(t) = \mathcal{S}^C(\tau^-) = \mathcal{S}^C(t)$ w.p. 1; we forget any earlier more detailed job class couplings. That is, if we require a more detailed coupling of job classes at any τ, we assume that these are *resampled*, given the aggregate coupling at time τ^-.

We have four cases corresponding to the different events that may happen at time τ.

Case 1: arrival. By letting the class of the arrival be the same in both systems, both conditions continue to hold at time τ.

Case 2: potential service completion of a server in $\overline{\mathcal{S}_2^{NC}(t) \cup \mathcal{S}^C(t)}$. This is a potential service completion at a server that is idle, so again, both conditions continue to hold at time τ.

Case 3: potential service completion of a server in $\mathcal{S}_2^{NC}(t) \cap \overline{\mathcal{S}^C}(t) \neq \emptyset$. In this case, the NC system necessarily has extra jobs, i.e., $N^C(t) < N^{NC}(t)$. Either no jobs leave in either system, or an extra job leaves in the NC system and no job leaves in the C system. Both conditions still hold.

Case 4: potential service completion of a server in $\mathcal{S}^C(t)$. By the nature of collaborative service, a potential service completion of a server in $\mathcal{S}^C(t)$ must result in a job completion in the C system, but not necessarily in the NC system. Because of the symmetry of the redundancy(d) system and the collaborative and noncollaborative policies, given $N^C(t) = N_1^{NC}(t)$ and $\mathcal{S}_1^{NC}(t) = \mathcal{S}^C(t)$, the jobs (basic jobs) in the C (NC) system are equally likely to be any of the job classes that are compatible with the servers in $\mathcal{S}^C(t)$. This allows us to sample the specific job classes present at time τ in the C and NC systems in order to maintain the coupling, as follows. If there is not a service completion in the NC system, then let one of the basic jobs in the NC system have the same class as the departing job in the C system, and relabel it as an extra job. Let the other basic jobs in the NC system be of the same class as the jobs in the C system. Then $N^C(\tau) = N^C(t) - 1 \leq N^{NC}(t) = N^{NC}(\tau)$, $N_2^{NC}(\tau) = N_2^{NC}(t) + 1$, and $\mathcal{S}_1^{NC}(\tau) = \mathcal{S}^C(\tau)$, w.p. 1. If there is a service completion and job departure in both systems, and if it is of an extra job in the NC system, then again let one of the basic jobs in the NC system have the same class as the departing job in the C system, and relabel it as an extra job (so it replaces the departing extra job). Letting the other basic jobs in the NC system be of the same class as the jobs in the C system, $N^C(\tau) = N^C(t) - 1 \leq N^{NC}(t) - 1 = N^{NC}(\tau)$, $N_2^{NC}(\tau) = N_2^{NC}(t)$, and $\mathcal{S}_1^{NC}(\tau) = \mathcal{S}^C(\tau)$, w.p. 1. Finally, suppose a basic job completes in the NC system. Then let the class of the completing job be the same for both systems, and, again coupling the classes of the other basic NC jobs with the jobs in the C system, we have $N^C(\tau) = N^C(t) - 1 \leq N^{NC}(t) - 1 = N^{NC}(\tau)$ and $\mathcal{S}_1^{NC}(\tau) = \mathcal{S}^C(\tau)$, w.p. 1.

Thus, in all cases, if $N^C(t) \leq N^{NC}(t)$ w.p. 1., then we can couple the systems so that $N^C(\tau) \leq N^{NC}(\tau)$ w.p. 1. We can repeat the argument at the next event time after τ to obtain the result of the theorem.

6.3 Optimal Scheduling in Nested Systems

For most of this paper we assume FCFS scheduling; we now briefly consider the Least Redundant First (LRF) policy and nested systems. Under LRF, at all times each server works on the job in its queue with the fewest compatible servers. That is, server s gives priority to jobs of class i over jobs of class $j \neq i$ if $i, j \in C_s$ and $|S_i| < |S_j|$. Note that this policy is well defined and provides a unique ordering among all job classes for each server due to the nested structure: it is not possible to have $i, j \in C_s$ and $|S_i| = |S_j|$ unless $i = j$. LRF which is known to be optimal in nested collaborative systems [15] and in the W model under noncollaboration [3]; the arguments of [15] easily extend this result to general nested systems under noncollaboration. We will show that

under LRF in nested systems, collaboration is better than noncollaboration. We first review the optimality results for LRF.

Let $N_i(t)$ be the number of class-i jobs in the system at time t, and let $N^i(t) = \sum_{j:S_j \subseteq S_i} N_j(t)$ be the total number of class-i jobs plus those jobs that have priority over class-i jobs at any server (i.e., the jobs that are less flexible than class-i jobs and share a server with class-i jobs). Note that $N_J(t)$ is the total number of jobs in the system. Finally, let $\vec{N}(t) = (N^1(t), N^2(t), \ldots, N^J(t))$. We have the following result from [15]. Note that the result does not require Poisson arrivals; we simply assume that arrivals are an arbitrary exogenous process.

Theorem 6 *[15] The preemptive non-idling LRF policy stochastically minimizes $\{\vec{N}(t)\}_{t=0}^{\infty}$ among all preemptive, possibly idling, policies for nested collaborative systems when service times are exponential and the arrivals form a general exogenous process.*

The proof of Theorem 6 is similar in flavor to the sample-path proofs of the previous section, and it easily extends to the noncollaborative case, assuming a nested system. It also extends to nested subsystems of larger systems. That is, we have the following results.

Corollary 7

(i) Non-idling LRF stochastically minimizes $\{\vec{N}(t)\}_{t=0}^{\infty}$, among all preemptive, possibly idling, noncollaborative *policies for nested systems with exponential service times and a general exogenous arrival process.*

(ii) Non-idling LRF maximizes the number of departures (job completions) by time t, for all t, and therefore also minimizes mean response time, for both collaborative and non-collaborative nested systems.

(iii) For any bipartite compatibility matching and for both collaborative and noncollaborative systems, if for some server s and two job classes i and j, $s \in S_i \subset S_j$, then server s should always give priority to class i over class j. Thus, dedicated classes that have only one compatible server should always have highest priority, and a fully flexible class (if any) should have lowest priority.

The sample-path/interchange arguments for parts (i) and (iii) are very similar to that in [15] and is omitted; part (ii) follows from (i) because $N^J(t) = N(t)$ and from Little's law.

Not surprisingly, if each server follows the optimal policy and serves its least flexible compatible job class at any particular time, then it also is optimal to assign as many servers as possible to that job class. That is, assuming LRF service in a nested system, collaboration is better than noncollaboration. Again, a sample path proof along the lines of our earlier proofs gives the result; we omit the details

Corollary 8 *The preemptive non-idling collaborative LRF policy stochastically minimizes $\{\vec{N}(t)\}_{t=0}^{\infty}$, among all policies in which preemption, idling, and collaboration are permitted, for nested systems with exponential service times and a general exogenous arrival process.*

6.4 The impact of flexibility

We briefly note here that for systems in which collaboration is beneficial, i.e., symmetric systems and optimally scheduled systems, more flexibility, in terms of increasing the proportion of jobs in classes with more compatible servers, is also beneficial. In asymmetric systems, more flexibility may not be more beneficial (see, e.g., [3,12,15]).

7 Response Time Bounds

Throughout most of this paper, our primary metric has been the number of jobs in the system. We now turn to the metric of response time, the time from when a job enters the system until it completes service and departs, revisiting the steady-state setting considered in Section 4.

Exact expressions for per-class and overall mean response time in the W model, for both the C and NC systems, follow directly from our expressions for $E[N^{NC}]$ in Section 4, by Little's Law: $E[T_i^x] = E[N_i^x]/\lambda_i$, $x = C, NC$. While the results for the C system were previously derived in [13], our results for the NC system are new. As was the case for mean number in system, the expressions for mean response time are quite long except in the special case of the symmetric W. For the symmetric W, we have:

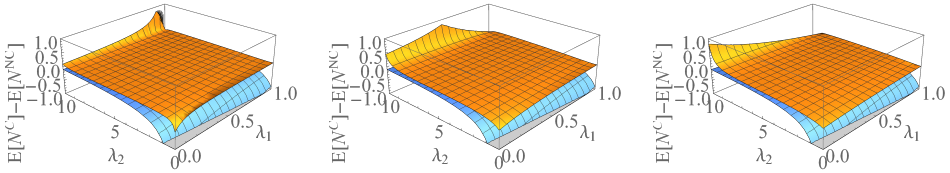
$$\begin{aligned}
E[T_1^C] = E[T_2^C] &= \frac{1}{\mu-\lambda} + \frac{1}{\mu-\lambda+\lambda_3} \\
E[T_3^C] &= \frac{1}{\mu-\lambda} \\
E[T_1^{NC}] = E[T_2^{NC}] &= \frac{1}{\mu-\lambda} - \frac{1}{\mu+\lambda_3} + \frac{1}{\mu} + \frac{1}{\mu-\lambda+\lambda_3} \\
E[T_3^{NC}] &= \frac{1}{\mu-\lambda} + \frac{\mu-\lambda+\lambda_3}{\mu(\mu+\lambda_3)}.
\end{aligned}$$

Comparing mean response time in the two systems, we find

$$\begin{aligned}
E[T_1^C] - E[T_1^{NC}] &= -\frac{\lambda_3}{\mu(\mu+\lambda_3)}, \\
E[T_3^C] - E[T_3^{NC}] &= -\frac{\mu-\lambda+\lambda_3}{\mu(\mu+\lambda_3)}, \\
E[T^C] - E[T^{NC}] &= -\frac{\lambda_3}{\lambda(\mu+\lambda_3)}.
\end{aligned}$$

As was the case for mean number in system, we again remark that collaboration is always better for the symmetric W, whereas collaboration can be worse in the asymmetric W.

In principle, one could obtain exact expressions for the probabilities of various combinations of idle and busy servers to get explicit mean response times in larger noncollaborative systems, but in general these would be too



(a) $\lambda_3 = 0.1$ (b) $\lambda_3 = 0.5$ (c) $\lambda_3 = 1$

Fig. 7 Comparing mean response time in the C and NC asymmetric W systems, where $\mu_1 = 1$, $\mu_2 = 10$, and λ_1 and λ_2 vary; we consider three different values of λ_3. Graphs show $E[T^C] - E[T^{NC}]$ (orange surface) and our bound on this difference (blue surface).

complicated to draw useful conclusions. In contrast, the sample-path bounds that we derive on the number of jobs in the system allow us to obtain bounds on the difference in mean steady-state response time, for any system structure. A corollary of Theorem 1 is that $E[N^{NC}] \leq E[N^C] + M$ and $E[N_i^{NC}] \leq E[N_i^C] + |S_i|$ where N^x (N_i^x), $x = C, NC$, is the steady-state number of jobs (of class i) in system. From Little's law, Theorems 1 and 3, and Lemma 1, we have the following.

Corollary 9 *Let* $\mu_{\min} = \min_j \mu_j$ *and* $\mu_{\min}^i = \min_{j \in S_i} \mu_j$.

(i) $E[T^{NC}] \leq E[T^C] + \frac{M}{\lambda}$ *and* $E[T_i^{NC}] \leq E[T_i^C] + \frac{|S_i|}{\lambda_i}$, $i = 1, ..., J$.
(ii) $E[T^{NC}] \leq E[T^C] + \frac{1}{\mu_{\min}}$.
(iii) $E[T_i^{NC}] \leq E[T_i^C] + \frac{1}{\mu_{\min}^i}$ *and* $E[T^{NC}] \leq E[T^C] + \sum_{i=1}^J \frac{\lambda_i}{\lambda \mu_{\min}^i}$.
(iv) If all servers have the same speed, $\mu_i \equiv \mu/M$, *then* $E[T^{NC}] \leq E[T^C] + \frac{M}{\mu}$, *and* $E[T_i^{NC}] \leq E[T_i^C] + \frac{|S_i|}{\mu}$, $i = 1, ..., J$.
(v) For the W model, $E[T^{NC}] \leq E[T^C] + \frac{1}{\lambda}$ *and* $E[T_i^{NC}] \leq E[T_i^C] + \frac{1}{\lambda_i}$, $i = 1, 2, 3$.

Note that, in the homogeneous servers case, the bound in case (v) is tighter than that of case (i).

Figure 7 shows the exact difference in mean response time between the C and NC systems, in the asymmetric W model, as well as our bound on this difference. We show the bound given in Corollary 9, which is tighter in this case. While the bound is not tight, it also is not very far off the exact results in most regimes; the bound is loosest when $\lambda_1 = \lambda_2 = 0$, and it becomes increasingly tight when λ_1 and λ_2 are high. In larger systems our bounds will have similar accuracy, indicating that collaboration cannot outperform noncollaboration by very much.

In the other direction, we see in Figure 7 that collaboration also never performs much better than noncollaboration, with respect to mean response time. Recall from Section 4 that the largest difference in mean number occurred when the system was very highly loaded. In this regime, large differences in mean number in system correspond to relatively modest differences in mean

response time, due to the fact that we divide by λ when applying Little's Law. The consequence is that, in most settings, the C and NC systems generally yield very similar response times: collaboration cannot be much better, and it appears never to be much worse. When we consider the relative benefit of collaboration, $(E[T^C] - E[T^{NC}])/E[T^C]$, the difference in performance between the two systems disappears almost entirely. Here, there is a material difference only when both λ_1 and λ_2 are very close to 0.

8 Conclusion

In this paper we compare the performance of the collaborative and noncollaborative models with respect to two metrics: number of jobs in system and steady-state mean response time. To enable these comparisons, we make several methodological contributions. First, we derive the first exact, closed-form analysis of mean number in system and mean response time in the noncollaborative W system in steady state. While previous papers provided the foundations necessary for these results, our synthesis of prior results allows us to directly and exactly compare steady-state performance metrics between the C and NC systems. Second, to prove the benefit of collaboration in symmetric systems we develop nonstandard coupling arguments that involve direct coupling of the number in system over time, but resampling or relabeling of particular jobs at different time points, by exploiting the symmetry of the systems under consideration.

Our results indicate that, while collaboration is always better in symmetric systems, the benefit of collaboration is bounded: in *any* system, collaboration cannot outperform noncollaboration by a great deal. On the other hand, we show that the collaborative system can have arbitrarily more jobs than the noncollaborative system, on coupled sample paths. Turning to mean response time, we find that the collaborative and noncollaborative systems tend to have very similar performance; the relative benefit or cost of collaboration is quite small. These findings have important implications for configuring systems in which collaboration is a design choice. In these systems, flexibility of jobs and servers—or, equivalently, redundantly dispatching jobs to multiple distributed servers—provides a benefit if either (1) queueing times differ substantially at different servers, or (2) an individual job's service times differ substantially at different servers. Both the collaborative and noncollaborative models (equivalently, cancel-on-complete and cancel-on-start, in redundancy system) gain the first benefit, but only collaboration (cancel-on-complete) has the potential to gain from the second point. Our results hold for i.i.d. exponential service times; here one job truly can have very different service times on multiple servers, making this setting particularly advantageous to collaboration. Yet even in this setting, our results indicate that collaboration cannot offer too much of a benefit over noncollaboration. In more realistic settings, in which a job's service times are correlated across servers, any potential benefit of collaboration will be even smaller. Furthermore, collaboration has additional costs

that we do not model; for example, it may be more difficult to cancel a job that is in service on multiple servers than a job that is waiting in the queue. We conclude, therefore, with a recommendation: in most real-world settings, noncollaboration will be the better choice.

Acknowledgements We thank the anonymous reviewers for their valuable feedback.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Adan, I. J. B. F., I. Kleiner, R. Righter, and G. Weiss. (2018) FCFS parallel service systems and matching models. Perf. Eval. 127: 253-272.
2. Adan, I., and Weiss, G. (2014) A skill based parallel service system under FCFS-ALIS—steady state, overloads, and abandonments. Stochastic System, 4(1):250-299.
3. Akgun, O., R. Righter, and R. Wolff. (2013) Partial flexibility in routeing and scheduling. Adv. Appl. Prob., 45: 637-691.
4. Andradottir, S., H. Ayhan, and D.G. Down. (2011). Queueing system with synergistic servers. Operations Research 59: 772-780.
5. Andradottir, S., H. Ayhan, and D.G. Down. (2013). Optimal assignment of servers to tasks when collaboration is Inefficient. Queueing Systems 75: 79-110.
6. Anton, E., U. Ayesta, M. Jonckheere, and I.M. Verloop. (2020). On the stability of redundancy models. Operations Research, Forthcoming.
7. Anton, E., U. Ayesta, M. Jonckheere, and I.M. Verloop. (2019) Redundancy with processor sharing servers. Performance Evaluation Review 47: 15-17.
8. Anton, E., U. Ayesta, M. Jonckheere, and I.M. Verloop. (2020). Improving the performance of heterogeneous data centers through redundancy. Proc. ACM Meas. Anal. Comput. Syst. 4: 1-29.
9. Ayesta, U., T. Bodas, and I.M. Verloop. (2018) On a unifying product form framework for redundancy models, IFIP Performance.
10. Ayesta, U., T. Bodas, and I.M. Verloop. (2018) On redundancy-d with cancel-on-start a.k.a Join-shortest-work (d), MAMA Workshop, SIGMETRICS.
11. Bonald, T., C. Comte, and F. Mathieu (2019). Performance of balanced fairness in resource pools: A recursive approach, ACM SIGMETRICS Perform. Eval. Rev. 46: 125-127.
12. Cadas, A.,J. Doncel, J.-M. Fourneau, and A. Bušić, Flexibility can hurt dynamic matching system performance, Preprint. https://arxiv.org/pdf/2009.10009.pdf
13. Gardner, K., S. Zbarsky, S. Doroudi, M. Harchol-Balter, E. Hyytia, and A. Scheller-Wolf. (2016). Queueing with redundant requests: exact analysis. Queueing Systems 83:227-259.
14. Gardner, K., M. Harchol-Balter, A. Scheller-Wolf, M. Velednitsky, and S. Zbarsky. (2017). Redundancy-d: The power of d choices for redundancy. Operations Research 65:4, 1078-1094.
15. Gardner, K., M. Harchol-Balter, E. Hyytia, and R. Righter. (2017). Scheduling for efficiency and fairness in systems with redundancy. Performance Evaluation 116:1-25.
16. Gardner, K., M. Harchol-Balter, A. Scheller-Wolf, and B. van Houdt. (2017). A better model for job redundancy: Decoupling server slowdown and job size. IEEE/ACM Transactions on Networking, 25: 3353-3367.
17. Gardner, K., and R. Righter. (2020). Product Forms for FCFS Queueing Models with Arbitrary Server-Job Compatibilities: An Overview. Queueing Systems, 96: 3-51.

18. Hopp, W. J. and M. P. van Oyen. (2004). Agile workforce evaluation: A framework for crosstraining and coordination."IIE Transactions 36: 919-940.
19. Isik, T., S. Andradottir, and H. Ayhan. (2016) Optimal control of queueing systems with non-collaborating servers. Queueing Systems, 84: 79-110.
20. Joshi, G., E. Soljanin, G. Wornell. (2015). Queues with redundancy: Latency-cost analysis, ACM SIGMETRICS Perf. Eval. Rev. 43: 54–56.
21. 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 2, https://doi.org/10.1145/3055281.
22. Kim, Y., R. Righter and R. Wolff. (2010) Grid scheduling with NBU service times, Operations Research Letters, 38: 502-504.
23. Koole, G. and R. Righter. (2008) Resource Allocation in Grid Computing, Journal of Scheduling 11: 163-174.
24. Mitzenmacher, M. (2001). The power of two choices in randomized load balancing. IEEE Transactions on Parallel and Distributed Systems, 12(10): 1094-1104.
25. Raaijmakers, Y., S. Borst, and O. Boxma. (2019) Redundancy scheduling with scaled Bernoulli service requirements. Queueing Systems 93: 67–82.
26. Raaijmakers, Y., S. Borst, and O. Boxma. (2020) Stability of redundancy systems with processor sharing. VALUETOOLS '20: Proceedings of the 13th EAI International Conference on Performance Evaluation Methodologies and Tools 120–127.
27. Van Oyen, M.P., E.G.S. Gel, and W.J. Hopp. (2001) Performance opportunity for workforce agility in collaborative and noncollaborative work systems. IIE Trans. 33: 761-777.
28. Visschers, J., Adan, I. J. B. F., and Weiss, G. (2012). A product form solution to a system with multi-type customers and multi-type servers. Queueing Systems 70: 269–298.
29. Wang, X., S. Andradottir, and H. Ayhan. (2015). Dynamic server assignment with task dependent server synergy. IEEE Transactions on Automatic Control, 60: 570-575.

A Proof of Lemma 1

Lemma 1. *[1,17] Suppose all the servers are busy at time 0 in NC, and let* $\gamma^C(0) =_{st} \gamma_Q^{NC}(0)$*. Let* τ *be the first time after time 0 at which any server becomes idle in NC. Then* $\{\gamma^C(t)\}_{t=0}^{\tau} =_{st} \{\gamma_Q^{NC}(t)\}_{t=0}^{\tau}$.

Proof Let $\gamma^C(0) = \gamma_Q^{NC}(0)$. We will couple the arrival sequences and potential service completions for the C and NC systems after time 0; with this coupling we will show by induction that $\{\gamma^C(t)\}_{t=0}^{\tau} = \{\gamma_Q^{NC}(t)\}_{t=0}^{\tau}$ wp 1.

Assume that $\gamma^C(t) = \gamma_Q^N C(t)$. Suppose that the next event is an arrival. Then the states for both C and NC are augmented by appending the class of the new arrival, and the result continues to hold. Now suppose that the next event is a potential service completion on server s. If this completion results in a departure from system C, then the first job in the state γ^C of any class $i \in C_s$ is removed. In system NC, this completion results in the corresponding job in the *queue* being able to start service, and thus being removed from the state γ_Q^{NC}; a job also departs from the system in NC, but this is not reflected in the queue state. If, on the other hand, the potential service completion on server s does not result in a departure in system C, then server s was idle at time t in system C. This means that none of the jobs in $\gamma_C(t)$, and hence also in $\gamma_Q^{NC}(t)$, are compatible with server s, and so when the next completion occurs on server s in system NC, server s will become idle. At this point, we have reached time τ.

B Additional Steady-State Results

Let $P_{1,2}$ (respectively, $P_{2,1}$) be the probability that both servers are idle and that server 1 (respectively, server 2) has been idle longer.

Lemma 5 *In the W model in the noncollaborative system, the probabilities* $P_\emptyset, P_1, P_2, P_{1,2}, P_{2,1}$ *are given by*

$$\begin{aligned}
P_\emptyset &= \pi^{ALIS}(\emptyset,\emptyset)\left(\frac{\mu_1}{\mu_1-\lambda_1}\right)\left(\frac{\mu_2}{\mu_2-\lambda_2}\right)\left(\frac{\mu-\lambda+\lambda_3}{\mu-\lambda}\right)\\
P_1 &= \pi^{ALIS}(\emptyset,\emptyset)\frac{\mu_1\mu_2}{(\mu_2-\lambda_2)(\lambda_1+\lambda_3)}\\
P_2 &= \pi^{ALIS}(\emptyset,\emptyset)\frac{\mu_1\mu_2}{(\mu_1-\lambda_1)(\lambda_2+\lambda_3)}\\
P_{1,2} &= \pi^{ALIS}(\emptyset,\emptyset)\left(\frac{\mu_1}{\lambda_1+\lambda_3}\right)\left(\frac{\mu_2}{\lambda}\right)\\
P_{2,1} &= \pi^{ALIS}(\emptyset,\emptyset)\left(\frac{\mu_2}{\lambda_2+\lambda_3}\right)\left(\frac{\mu_1}{\lambda}\right),
\end{aligned}$$

where $P_\emptyset + P_1 + P_2 + P_{1,2} + P_{2,1} = 1$.

Proof We derive the probabilities $P_\emptyset, P_1, P_2, P_{1,2}, P_{2,1}$ for the noncolloborative (ALIS) W model by relating them to probabilities for the collaborative model, as in [17]. Let $\pi^{ALIS}(\emptyset,\emptyset)$ be the stationary probability that both servers are busy and no jobs are waiting in the queue under the noncollaborative ALIS policy, let $\pi^C(\emptyset)$ be the stationary probability that the system is empty for the W model under collaboration, and let $\pi_j^C(\emptyset)$ be the probability that the collaborative system is empty when the system consists only of server j and job class j, $j = 1, 2$. From Lemma 2, the latter probabilities are the same as the conditional probability that the collaborative W system is empty given that the other server (not server j) is idle. Then, by Theorem 4.5 in [17], $\pi_j^C(\emptyset) = 1 - \frac{\lambda_j}{\mu_j} = \frac{\mu_j-\lambda_j}{\mu_j}$, $j = 1, 2$, and

$$\pi^C(\emptyset) = \left(\frac{\mu_1-\lambda_1}{\mu_1}\right)\left(\frac{\mu_2-\lambda_2}{\mu_2}\right)\left(\frac{\mu-\lambda}{\mu-\lambda+\lambda_3}\right).$$

Once we have obtained $\pi_j^C(\emptyset)$, $j = 1, 2$, and $\pi^C(\emptyset)$, the result then follows from Corollary 3.12 of [17].

We now consider the special case of the symmetric W model, in which $\mu_1 = \mu_2 = \mu/2$ and $\lambda_1 = \lambda_2 = (\lambda - \lambda_3)/2$. In this case, class-3 jobs are equally likely to enter service on either server, so $P_1(3) = P_2(3) = \lambda_3/\mu$, and the expressions for $P_\emptyset$, P_1, P_2, $P_{1,2}$, and $P_{2,1}$ simplify nicely.

Corollary 10 *In the symmetric W model in the noncollaborative system, we have*

$$P_\emptyset = \frac{\lambda}{\mu}\frac{\lambda+\lambda_3}{\mu+\lambda_3}$$
$$P_1 = P_2 = \frac{\lambda}{\mu}\frac{\mu-\lambda}{\mu+\lambda_3}$$
$$P_{1,2} = P_{2,1} = \frac{(\mu-\lambda)(\mu-\lambda+\lambda_3)}{2\mu(\mu+\lambda_3)}.$$

Lemma 6 *In the W model in the noncollaborative system, the probabilities that server j, $j = 1, 2$, is working on a class-i job, $i = 1, 2, 3$, are given by*

$$P_j(j) = \lambda_j/\mu_j, \qquad j = 1, 2$$
$$P_1(3) = P_\emptyset + P_2 - P_1(1)$$
$$P_2(3) = P_\emptyset + P_1 - P_2(2).$$

Proof $P_j(j) = \lambda_j/\mu_j$, $j = 1, 2$ follows from Little's law. The overall probability that server 1 is busy is $P_\emptyset + P_2$ and therefore the probability that server 1 is busy with a class-3 job is $P_1(3) = P_\emptyset + P_2 - P_1(1)$; similarly, $P_2(3) = P_\emptyset + P_1 - P_2(2)$.

C Proof of Theorem 3

Theorem 4. *For the W model, we can couple the systems so that, with probability 1, $\{N_i^C(t)\}_{t=0}^\infty \geq \{N_i^{NC}(t)-1\}_{t=0}^\infty$ for $i = 1, 2, 3$, $\{N_i^C(t)+N_3^C(t)\}_{t=0}^\infty \geq \{N_i^{NC}(t)+N_3^{NC}(t)-1\}_{t=0}^\infty$ for $i = 1, 2$, and $\{N^C(t)\}_{t=0}^\infty \geq \{N^{NC}(t)-1\}_{t=0}^\infty$.*

Proof We use induction on t. Suppose all the inequalities hold at time t, and consider the time of the next event, τ. (The result is immediately true for $t = 0$).

We first show that $N_i^C(\tau) \geq N_i^{NC}(\tau) - 1$ for $i = 1, 2$. This also follows from Theorem 1, but we give an alternative, more direct, proof here. Without loss of generality, we consider $i = 1$. The inequality continues to hold at time τ if (1) τ is an arrival time, (2) τ is a potential service completion time at server 2, or (3) τ is a potential service completion time at server 1 and there is a class-1 job in service at server 1 in both systems; in all of these cases the number of class-1 jobs evolves in the same way at time τ in both systems. Now consider the case where a class-1 job is in service at time t in NC but not in C. Then $N_1^C(\tau) = N_1^C(t) \geq N_1^{NC}(t)-1 = N_1^{NC}(\tau)$, and again, the result continues to hold. Similarly, if a class-1 job is in service at time t in C but not in NC, and $N_1^C(t) > N_1^{NC}(t) - 1$, then at time τ we have $N_1^C(\tau) = N_1^C(t) - 1 \geq N_1^{NC}(t) - 1 = N_1^{NC}(\tau) - 1$, and the result continues to hold. Finally, we consider the case where a class-1 job is in service at time t in C but not in NC, and $N_1^C(t) = N_1^{NC}(t)-1$. We will show by contradiction that this case cannot occur. By observation 2, all class-1 jobs present in C are also

present in NC, and the "extra" class-1 job in NC arrived earlier than all other class-1 jobs in either system. By the induction hypothesis, $N_3^C(t) \geq N_3^{NC}(t)$, and again by observation 2 all class-3 jobs present in system NC at time t are also present in system C. Because the extra job has departed in system C, it must have arrived earlier than all class-3 jobs present in system C at time t, and hence it arrived earlier than all class-3 jobs present in system NC at time t. Therefore, the extra job must be in service on server 1 in system NC, contradicting the assumption that system NC does not have a class-1 job in service.

Now we show that $N_i^C(\tau) + N_3^C(\tau) \geq N_i^{NC}(\tau) + N_3^{NC}(\tau) - 1$ for $i = 1$, without loss of generality. As before, the only case that is not straightforward is the case in which (i) $N_1^C(t) + N_3^C(t) = N_1^{NC}(t) + N_3^{NC}(t) - 1$, (ii) τ is a potential service completion time, and (iii) the server that completes at time τ is serving a class-1 or class-3 job in system C but not in system NC. From (i), we know that system NC has one extra class-1 or class-3 job that is not present in system C. From the inductive hypothesis and observations 1 and 2, we have that, not counting the extra job, systems C and NC have the same sets of class-1 and class-3 jobs, and system C has a superset of the class-2 jobs present in system NC. From (iii), we have that either server 1 must be idle in NC and not in C, or server 2 must be serving a class-3 job in C and not in NC. We will show by contradiction that neither of these cases is possible. If server 1 is idle in NC, then the extra job must be a class-3 job that is in service on server 2, the extra job must be the only class-1 or class-3 job in system NC, and there must be no clas-1 or clas-3 jobs in system C; this contradicts (iii). Now suppose that server 2 is serving a class-3 job in C but not in NC. By observation 3, this class-3 job must be the earliest arrival of all class-2 and class-3 jobs in system C. By the inductive hypothesis and observations 1 and 2, this class-3 job must also have arrived before all other class-3 and class-3 jobs in system NC, with the possible exception of the extra job (if the extra job is class-3). Thus, in system NC, either server 2 is serving the same class-3 job as in system C, or server 2 is serving the extra job. In either case, a class-3 job is in service in system NC, contradicting (iii).

Next we show $N_3^C(\tau) \geq N_3^{NC}(\tau) - 1$. Again, all cases are straightforward except for the case where $N_3^C(t) = N_3^{NC}(t) - 1$ and the event at time τ is a service completion. In this case, the extra job in system NC is a class-3 job. By the induction hypothesis and observations 1 and 2, excluding the extra job, systems C and NC have the same sets of class-3 jobs, and system C contains a superset of the class-1 jobs and class-2 jobs present in system NC. If a class-3 job is in service on at least one of the servers in system C, then we can use a similar argument to that in the previous case to show that, in system NC, the same class-3 job and the extra job are in service. Hence, after a service completion at time τ, we still have $N_3^C(\tau) \geq N_3^{NC}(\tau) - 1$.

Finally, we show $N^C(\tau) \geq N^{NC}(\tau) - 1$. Again, all cases are straightforward except for the case where $N^C(t) = N^{NC}(t) - 1$ and the event at time τ is a service completion. In this case, system NC has one extra job, and, from the induction hypothesis and observations 1 and 2, all the other jobs are the same

in both systems. We need only show that if a server is idle in system NC, then that same server also is idle in system C; without loss of generality, we will show this for server 1. Assume server 1 is idle in system NC. Then the extra job is not a class-1 job and there are no other class-1 jobs in either system. Moreover, if the extra job is a class-3 job, it is the only class-3 job in system NC, and therefore there are no class 3 jobs in system C; thus, server 1 also is idle in system C. If the extra job is a class-2 job, then it must have arrived before any class-3 jobs in both systems (by observation 3), and before any other class-2 jobs in system NC, so it is in service on server 2 in system NC. Thus, server 1 being idle in system NC means that there are no class-3 or class-1 jobs in system NC, and therefore also in system C, so server 1 must also be idle in system C.

D Proofs for Section 6

Corollary 1. *For the symmetric extended W model,* $\{N^C(t)\}_{t=0}^{\infty} \leq_{st} \{N^{NC}(t)\}_{t=0}^{\infty}$.

Proof The proof is basically the same as that of Theorem 4; we just describe the differences. Assume the definitions above, with job a being a fully flexible job at the head of the queue at time s, and where policy π assigns job a to just one server. The argument above shows that a policy π' that assigns job a to two of the available servers can be constructed that will improve upon π. The argument can be repeated to show that assigning job a to n servers will be better than assigning it to $n-1$.

Corollary 2. *For a symmetric extended W subsystem embedded in a larger system, and considering collaboration within the subsystem,* $\{N^C(t)\}_{t=0}^{\infty} \leq_{st} \{N^{NC}(t)\}_{t=0}^{\infty}$.

Proof Again the argument is very similar to that of the symmetric W model. Define π, π', $N(t)$, $N'(t)$, job a, times s and τ, and servers 1 and 2 as in the proof of Theorem 4.

If the event at time τ is an arrival, we have $\{N'(t)\}_{t=0}^{\infty} = \{N(t)\}_{t=0}^{\infty}$ by the same argument as in the proof of Theorem 4. If the event at time τ is a potential service completion (1) at server 1, (2) at server 2 and server 2 is idle under π, (3) at server 2 and server 2 is serving a job of the same class as job a under π, or (4) at server 2 and server 2 is serving a dedicated job, then, using the same argument as in the proof of Theorem 4, we have $\{N'(t)\}_{t=0}^{\infty} \leq \{N(t)\}_{t=0}^{\infty}$.

For the extended W model, we have one additional case, in which there is a potential service completion at server 2 at time τ and under π a job of class $i > M+1$ completes service. Let us call the job that completes under π job d. Now the states just after time τ under the two policies are the same except that π still has job a while π' still has the more flexible job d. Let π' agree with π from time τ on, treating job d the same way π treats job a. Then $\{N'(t)\}_{t=0}^{\infty} = \{N(t)\}_{t=0}^{\infty}$.

Corollary 3. *For the symmetric nearest neighbor model,* $\{N^C(t)\}_{t=0}^{\infty} \leq_{st} \{N^{NC}(t)\}_{t=0}^{\infty}$.

Proof Here again the argument is as in the proof of Theorem 4, with the same definitions, except that, as in the proof of Corollary 4, we have an additional case for the potential service completion at time τ: that server 2 completes service and the job it is serving under π (which we call job e) is flexible, but is of a different class than job a. Then the states under π and π' will be the same just after time τ except that job a is present under π and job e is present under π'. Note that π has chosen to serve job a on server 1 only, so that henceforth π will treat a as a dedicated job on server 1. Let π' assign job e to server 2 only, so that it treats job e as a dedicated job for server 2. Then let us swap the labels of servers 1 and 2 under π' and let π' agree with π from time τ on. As in the proof of Theorem 4, we can swap the server labels without loss of generality because of the symmetry of our system and policies, and because we have not conditioned on the state of the queue beyond the first job in the queue and the total number of jobs. Again we have $\{N'(t)\}_{t=0}^{\infty} = \{N(t)\}_{t=0}^{\infty}$.

Corollary 4. *For the symmetric M model,* $\{N^C(t)\}_{t=0}^{\infty} \leq_{st} \{N^{NC}(t)\}_{t=0}^{\infty}$.

Proof The proof is similar to that of Theorem 4; we define π, π', $N(t)$, and $N'(t)$ in the same way. Suppose that π and π' have behaved identically up until some time s, and let job a be the first job of its class in the system at time s (without loss of generality, assume class 2). We will consider two cases: the case in which, at time s, π assigns job a to server 3 only and π' assigns job a to both servers 2 and 3, and the case in which, at time s, π assigns job a to server 2 only and π' assigns job a to both servers 2 and 3. In both cases, we assume that π' agrees with π for the assignment to server 1. Let τ be the time of the next event after time s. If τ is either an arrival or a potential service completion on server 1, in both of the above cases we continue to have $\{N^C(t)\}_{t=0}^{\infty} = \{N^{NC}(t)\}_{t=0}^{\infty}$.

We now consider the case in which π assigns job a to server 3 and π' assigns job a to both servers 2 and 3. We have three cases for the event at time τ. First, suppose τ is a potential service completion time on server 3. Then job a departs under both policies, and, letting π' agree with π from time τ on, we have $\{N^C(t)\}_{t=0}^{\infty} = \{N^{NC}(t)\}_{t=0}^{\infty}$. Second, suppose τ is a potential service completion time on server 2, and server 2 is idle under π. Then at time τ, job a departs under π' but not under π. Let π' idle server 3 while π schedules job a on server 3, and let the two policies agree otherwise. Then $\{N'(t)\}_{t=0}^{\infty} \leq \{N(t)\}_{t=0}^{\infty}$. Third, suppose τ is a potential service completion time on server 2, and server 2 is serving a class 2 job under π; call this job b. Then, just after time τ, the system state is the same under π and π' except that job a is the first class-2 job under π and job b is the first class-2 job under π'. Let π' treat job b the same way π treats job a (assigning it to server 3 only) and otherwise agree with π, and again $\{N'(t)\}_{t=0}^{\infty} = \{N(t)\}_{t=0}^{\infty}$.

We now consider the case in which π assigns job a to server 2 and π' assigns job a to both servers 2 and 3 (and the two policies agree on the assignment at

server 1). We have three cases for the event at time τ. First, if τ is a potential service completion time at server 2, then job a departs under both policies and $\{N^C(t)\}_{t=0}^{\infty} = \{N^{NC}(t)\}_{t=0}^{\infty}$. Second, if τ is a potential service completion time at server 3 and either server 3 is idle under π or server 3 is serving a class-2 job under π, then we can use the argument above to obtain $\{N'(t)\}_{t=0}^{\infty} \leq \{N(t)\}_{t=0}^{\infty}$. Third, suppose that τ is a potential service completion time at server 3 and π assigns a class-1 job, which we call job b, to server 3 at time s. The argument above shows that π should also assign job b to server 1, hence we can assume that π assigns job b to both server 1 and server 3. Hence we know that, at time s, there is at least one job of each type under both policies, that π collaborates on the first class-1 job, and that π' collaborates on the first class-2 job. Because the system and policies are symmetric, and because we have not conditioned on the system state beyond the presence of jobs a and b, we can let π' swap the labels of the two job classes. Letting π' agree with π from time τ on, we obtain $\{N'(t)\}_{t=0}^{\infty} = \{N(t)\}_{t=0}^{\infty}$.