Insensitivity for Matching Systems

Runhan Xie, Elene Anton, Kristen Gardner, Rhonda Righter

Queueing Systems (2025) 109:15
https://doi.org/10.1007/s11134-025-09943-4

Insensitivity for Matching Systems

Runhan Xie[1] · **Elene Anton**[2] · **Kristen Gardner**[3] · **Rhonda Righter**[1]

Received: 2 February 2024 / Revised: 18 April 2025 / Accepted: 26 April 2025 /
Published online: 2 June 2025

Abstract
We study matching queues motivated by applications in ride-sharing platforms, where items (e.g., drivers and riders) arrive over time and are matched based on a compatibility graph. Our model generalizes classical compatibility queueing models and queue-based ride-sharing models by allowing exogenous driver arrivals, queues for both drivers and riders, and arbitrary bipartite compatibility structures. We provide sufficient conditions under which the stationary distribution exhibits insensitivity to 1) service-time (drive-time) distributions and 2) the matching policies of drivers and riders, respectively. Using tools from Kelly networks and order-independent queues, we provide simple and unified proofs of insensitivity to service-time distributions and to matching policies for a broad class of models. Our policy insensitivity result requires a finite buffer; we demonstrate through counterexamples that such insensitivity may fail when the buffer is infinite. Our results also shed light on insensitivity in related redundancy systems, offering new insights into their stationary behavior.

Keywords Matching Queues · Insensitivity

✉ Runhan Xie
runhan_xie@berkeley.edu

Elene Anton
elene.anton@univ-pau.fr

Kristen Gardner
kgardner@amherst.edu

Rhonda Righter
rrighter@berkeley.edu

[1] Department of Industrial Engineering and Operations Research, University of California, Berkeley, USA

[2] Department of Computer Science, LIUPPA, Université de Pau et des Pays de l'Adour, Anglet, France

[3] Department of Computer Science, Amherst College, Amherst, USA

1 Introduction

In the study of queueing systems, we are often interested in finding the stationary distribution of the system state, which in turn can be used to compute various performance measures of interest. Under the assumptions that the arrival process is Poisson, the service-time distribution is exponential, and the service discipline is first-come first-served (FCFS), many queueing systems have elegant product-form stationary distributions. However, the exponentiality and FCFS assumptions are often unrealistic. Although relaxing these assumptions typically makes the analysis of the stationary distribution less tractable, some queueing systems are *insensitive* to the service-time distribution, meaning that they have the same stationary distribution under all service-time distributions with the same means. Other systems are *insensitive* to the scheduling policy or to the rule by which arriving jobs are assigned to idle servers when there are multiple such servers available (see [14] for classical examples such as the $M/G/\infty$ queue).

Recently, *matching queues* have attracted considerable attention. In a matching queue, an item that arrives to the system will depart when it is matched with some other item. Often, a bipartite graph defines a compatibility structure among item classes: Nodes in this graph represent item classes, and there is an edge between two nodes if items of the corresponding classes can be matched with each other. A key example of a matching system with a bipartite compatibility structure is a ride-sharing platform. Here, riders who seek to travel from one location to another must be matched to compatible drivers.

As we will see below, the details of the ride-sharing model can vary in terms of whether drivers or riders or both can wait in a queue, and, if so, whether the queue has buffer restrictions; whether or not drivers can enter the system exogenously and depart after completing a trip; and whether there are compatibility restrictions that limit which classes of drivers and riders can be matched. Our goal in this paper is to identify conditions under which the ride-sharing system exhibits insensitivity to the drive-time distribution and to the assignment rule that determines how to match arriving riders to available drivers (when drivers are allowed to queue). While we use the language of "drivers" and "riders" throughout this paper for ease of explanation, we note that our results apply to any bipartite matching system; classical parallel-server queues with compatibilities are a special case.

Adan and Weiss [1] considered a loss model with Poisson arrivals of jobs, job-server compatibilities, and exponential servers with heterogeneous rates. An arriving rider (job, in their terminology) is immediately matched with the compatible driver (server, in their terminology) that has been idle the longest if there is one (assign-longest idle server, ALIS, which is analogous to first-come first-served); otherwise, the rider is lost. The set of drivers is fixed, and they alternate between driving and waiting for a rider in the idle-driver queue. Adan and Weiss derive the stationary distribution of the set of idle drivers, listed in the order in which they became idle, and show that this distribution is insensitive to the drive-time distributions (service-time distributions, in their terminology). Their proof of insensitivity involves establishing a partial differential equation (PDE) that models the dynamics of the system under any drive-time distribution; they then show that the stationary distribution obtained

under the exponential assumption is the unique solution of the PDE. Haji and Ross [12] prove that, with the additional restriction that the compatibilities of riders to drivers are exchangeable, the stationary distribution is insensitive to both the drive-time distributions and the driver-assignment rule. They show drive-time insensitivity via the "method of stages," which approximates a general distribution using a mixture of exponential distributions. Both the PDE method [1] and the method of stages [12] for showing drive-time insensitivity are complicated. We generalize Adan and Weiss's result by allowing drivers to arrive to and depart from the system. We generalize Haji and Ross's result for insensitivity to the driver-assignment rule by allowing a queue for riders (jobs).

Banerjee et al. [4] consider a ride-sharing model in which drivers wait for riders under the first-come-first-matched (FCFM) discipline, but arriving riders who find no available drivers are lost. Riders and drivers arrive to the system according to independent Poisson processes. There are no compatibility restrictions; any driver may be matched with any rider (this corresponds to a complete bipartite compatibility graph). When a driver-rider pair is matched, the driver departs from the queue and drives the rider to their destination; the rates of completing these trips are heterogeneous. Upon completing a service (i.e., after dropping off a rider at their destination) a driver may choose either to depart from the system or to return to the driver queue to wait for a new rider. Banerjee et al. show that the steady-state distribution of the number of waiting drivers is insensitive to the drive-time distribution. We generalize their model by allowing arbitrary rider-driver compatibilities.

Gopalakrishnan et al. [11] consider the classic $M/M/k$ queue, but with heterogeneous service rates; in the language of drivers and riders, this corresponds to a system in which at most one driver of each class can wait in the idle-driver queue, and the rider queue is unbounded. They show that the stationary distribution is insensitive to the driver-assignment rule, among rules that depend only on the order in which drivers became idle. However, like Banerjee et al. [4], their model requires a complete bipartite compatibility graph between drivers and riders. We extend their result by allowing for exchangeable rider-to-driver compatibilities.

In this paper, we leverage the properties of Kelly networks [14] and order-independent queues [6, 7] (reviewed in Section 2) to develop simple proofs of both insensitivity to drive-time distributions and insensitivity to the rule by which riders are assigned to available drivers (policy insensitivity). We show that the stationary distribution is insensitive to the drive-time distribution in bipartite matching systems with arbitrary rider-to-driver compatibilities in which drivers arrive to the system exogenously, wait in a queue for compatible riders, and can return to the queue after completing a trip (Section 3); each of these features generalizes along at least one dimension the models of Adan and Weiss [1], Haji and Ross [12], and Banerjee et al. [4]. We show insensitivity to the policy by which riders are assigned to available waiting drivers in systems with a fixed set of drivers, exchangeable rider-to-driver compatibilities, and a queue in which riders can wait for compatible drivers (Section 4); again, these features generalize the models of Haji and Ross [12] and Gopalakrishnan et al. [11]. Tables 1 and 2 summarize these results. We then turn our attention to the model with exchangeable rider-to-driver compatibilities but with an infinite queue for drivers, and no queue for riders. That is, rather than a fixed set of nonhomogeneous

Table 1 Summary of results: drive-time distribution insensitivity (Section 3)

	Driver queue	Rider queue	Compatibilities	Driver arrivals
Adan and Weiss [1]	1 buffer slot per class	No queue	Arbitrary	Closed
Haji and Ross [12]	1 buffer slot per class	No queue	Exchangeable	Closed
Banerjee et al. [4]	Infinite buffer	No queue	Complete	Open
Our model	1 buffer slot per class or infinite buffer	No queue	Arbitrary	Open

Table 2 Summary of results: driver-assignment policy insensitivity (Section 4)

	Driver queue	Rider queue	Compatibilities	Driver arrivals
Haji and Ross [12]	1 buffer slot per class	No queue	Exchangeable	Closed
Gopalakrishnan et al. [11]	1 buffer slot per class	Infinite buffer	Complete	Closed
Our model	1 buffer slot per class	Any OI buffers (Def. 2.2)	Exchangeable	Closed

drivers, we have a set of driver classes, and we permit there to be more than one waiting driver per class.

We show that, surprisingly, a similar policy insensitivity result for assigning waiting drivers to arriving riders does *not* hold in this setting. A similar argument can be used to show that an earlier conjecture of policy insensitivity for assigning waiting jobs to servers in power-of-d redundancy systems does not hold (Section 4.2). Throughout, we discuss how our results for one- and two-sided matching models apply to related redundancy systems.

2 Preliminaries

In this section, we provide a brief introduction to order-independent queues, one- and two-sided matching queues, and Kelly networks. The models and results presented in this section will form the basis for our insensitivity results in the sections that follow.

2.1 Order-independent Queues

The order-independent (OI) queue, first introduced by Berezner, Kriel, and Krzesinski [6], is defined as follows.

Definition 2.1 Class-i customers arrive according to a Poisson process with rate α_i and wait in a queue in order of arrival. Denote the system state by $\vec{c}_n = (c_1, \ldots, c_n)$, where c_i is the class of the customer at position i of the queue, n is the number of customers, and let $\mathcal{C}$ denote the set of all possible states. The queue is called an *order-independent queue* if

1. The total service rate of the first $i \leq n$ customers in the queue in state $\vec{c}_n$ (henceforth denoted by $\beta(\vec{c}_i)$) is the same for all permutations of these i customers.

2. The marginal service rate of the i-th customer in the queue in state $\vec{c}_n$ is $\beta(\vec{c}_i) - \beta(\vec{c}_{i-1}) \geq 0$ and is independent of the customers behind it.
3. $\beta(c) > 0$ for any customer class c.

When a customer arrives to the system, either (i) it is *accepted*, in which case it joins the end of the queue, or (ii) it is *rejected*, in which case it departs from the system immediately without receiving service. Whether a customer is accepted or rejected is determined by the system's *acceptance region* [7], denoted by $\mathcal{S} \subseteq \mathcal{C}$. A class-$c$ customer that arrives when the system is in state $\vec{c}_n = (c_1, \ldots, c_n)$ will be accepted if $(\vec{c}_n, c) = (c_1, \ldots, c_n, c) \in \mathcal{S}$ and rejected otherwise. Define the indicator function

$$\mathcal{I}_{\mathcal{S}}(\vec{c}_n) = \begin{cases} 1, & \text{if } (\vec{c}_n) \in \mathcal{S} \\ 0, & \text{otherwise.} \end{cases}$$

Definition 2.2 Let $\mathcal{S} \subseteq \mathcal{C}$ denote a subset of the state space. We say that $\mathcal{S}$ is an *order-independent acceptance region* if it satisfies the following properties:

1. For all permutations $\sigma(\vec{c}_n)$ of the customers in $\vec{c}_n$, $\mathcal{I}_{\mathcal{S}}(\vec{c}_n) = \mathcal{I}_{\mathcal{S}}(\sigma(\vec{c}_n))$.
2. If $\vec{c}_n \in \mathcal{S}$, then $\vec{c}_{n-1} \in \mathcal{S}$.

Remark 2.1 We draw attention to three special cases.

- The acceptance region where $\mathcal{S} = \mathcal{C}$ corresponds to the infinite buffer setting in which customers are never rejected. This is the setting that we consider in Section 3.
- The acceptance region $\mathcal{S} = \{\vec{c}_n \ : \ n \leq k\}$ corresponds to a system with a fixed buffer capacity that can accommodate at most k customers, regardless of their classes.
- The acceptance region $\mathcal{S} = \{\vec{c}_n \ : \ c_i \neq c_j \ \forall i, j \in \{1, \ldots, n\}, i \neq j\}$ corresponds to a system in which at all times the queue may contain at most one customer of each class (i.e., each class has a single dedicated buffer slot). This is the setting that we consider in Section 4.

Theorem 2.1 *([6], Theorem 1) Consider an order-independent queue with an order-independent acceptance region $\mathcal{S}$, and let $\pi(\vec{c}_n)$ denote the stationary probability that the system is in state $\vec{c}_n \in \mathcal{S} \subseteq \mathcal{C}$. Assume*

$$G := 1 + \sum_{n=1}^{\infty} \sum_{\vec{c}_n \in \mathcal{S}} \prod_{i=1}^{n} \frac{\alpha_{c_i}}{\beta(\vec{c}_i)} < \infty.$$

Then the queue is stable and quasi-reversible and the stationary distribution satisfies:

$$\pi\ (\vec{c}_n) = \pi(\varnothing) \prod_{i=1}^{n} \frac{\alpha_{c_i}}{\beta\ (\vec{c}_i)} \tag{1}$$

for all $\vec{c}_n \in \mathcal{S}$, where $\pi(\varnothing) = 1/G$.

Henceforth, we will use OI queue to mean an order-independent queue with an order-independent acceptance region.

An important special case of the product-form result above is the setting in which the total service rate depends only on the number of waiting customers and not on their particular classes, i.e., $\beta(\vec{c}_n) = \hat{\beta}_n$ for all n and $\vec{c}_n$. An immediate example is a standard multi-class parallel-server queue without compatibilities and with c servers ($\hat{\beta}_k = \min\{k, c\}$ regardless of customer classes). In this case, the stationary probability of being in state $\vec{c}_n$,

$$\pi(\vec{c}_n) = \pi(\varnothing) \prod_{i=1}^{n} \frac{\alpha_{c_i}}{\hat{\beta}_i},$$

is the same for any permutation $\sigma(\vec{c}_n)$ of the customers in $\vec{c}_n$. For this special case with class-independent service rates, we can generalize the marginal position-based service rate of the OI queue, which will enable us to show our policy insensitivity result in Section 4.

Definition 2.3 We say we have a *position only (PO) queue*, with $\vec{c}_n$ as the vector of customer classes in order of arrival, if the OI acceptance region conditions of definition 2.2 hold, and if the conditions of definition 2.1 are modified as follows:

1. The total service rate for all n customers in the queue in state $\vec{c}_n$ (denoted by $\hat{\beta}_n$) depends only on n.
2. The marginal service rate of the i-th customer in the queue in state $\vec{c}_n$, denoted by $\Delta_i(n)$, depends only on i and n and is such that $\Delta_i(n) \geq 0$ and $\sum_{i=1}^{n} \Delta_i(n) = \hat{\beta}_n$. ($\Delta_i(n)$ is otherwise arbitrary.)
3. $\hat{\beta}_1 > 0$.

For PO queues we can modify Berezner et al.'s proof to show that the stationary distribution does not depend on the marginal service rates, so all (possibly randomized) position-based service rules yield the same stationary distribution.

Corollary 2.1 *For a PO queue with any marginal service rates, for all $\vec{c}_n \in \mathcal{S}$,*

$$\pi(\vec{c}_n) = \pi(\varnothing) \prod_{i=1}^{n} \frac{\alpha_{c_i}}{\hat{\beta}_i}.$$

Proof Given $\vec{c}_n \in \mathcal{S}$ and $\Delta_i(n)$, $i = 1, ..., n$, we will show that

$$\pi(\vec{c}_n) = \pi(\varnothing) \prod_{i=1}^{n} \frac{\alpha_{c_i}}{\hat{\beta}_i} = \pi(\varnothing) \frac{\alpha_{c_n}}{\hat{\beta}_n} \prod_{i=1}^{n-1} \frac{\alpha_{c_i}}{\hat{\beta}_i}$$

satisfies the partial balance equations. First, because $\vec{c}_{n-1} \in \mathcal{S}$, the rate into state $\vec{c}_n$ due to an arrival equals the rate out of state $\vec{c}_n$ due to a service completion:

$$\alpha_{c_n} \pi(\vec{c}_{n-1}) = \alpha_{c_n} \pi(\varnothing) \prod_{i=1}^{n-1} \frac{\alpha_{c_i}}{\hat{\beta}_i} = \pi(\varnothing) \prod_{i=1}^{n} \frac{\alpha_{c_i}}{\hat{\beta}_i} \hat{\beta}_n = \hat{\beta}_n \pi(\vec{c}_n).$$

Also, for any state $\vec{c}_n$ such that $(\vec{c}_n, c) \in \mathcal{S}$, the rate into state $\vec{c}_n$ due to a class-c service completion equals the rate out of state $\vec{c}_n$ due to a class-c arrival:

$$\sum_{k=1}^{n+1} \Delta(k)\pi(c_1, \ldots, c_{k-1}, c, c_k, \ldots, c_n) = \sum_{k=1}^{n+1} \Delta(n+1)\pi(\varnothing) \frac{\alpha_c \prod_{i=1}^{n} \alpha_{c_i}}{\hat{\beta}_{n+1} \prod_{i=1}^{n} \hat{\beta}_i}$$

$$= \pi(\varnothing)\alpha_c \prod_{i=1}^{n} \frac{\alpha_{c_i}}{\hat{\beta}_i} = \alpha_c \pi(\vec{c}_n).$$

Observe that if $(\vec{c}_n, c) \notin \mathcal{S}$ then the probability of being in state $(\vec{c}_n, c)$ (or any permutation thereof) is 0, while the right-hand side of the corresponding balance equation is also equal to 0 because the arriving class-c job is lost; hence, the balance equation holds in this case as well. □

2.2 One-sided matching queue

Consider a ride-sharing platform where drivers and riders of different classes arrive according to independent Poisson processes with rate α_i for class-i drivers and rate β_i for class-i riders. There is a bipartite compatibility graph between the driver and rider classes. Arriving drivers wait in a queue in the order in which they arrived. An arriving rider matches with the first compatible driver in the queue (first-come-first-matched, FCFM) and leaves the system together with the driver; riders who find no compatible drivers are immediately lost. Meanwhile, drivers are accepted to the system based on an OI acceptance region $\mathcal{S}$, as described in Definition 2.2. We denote the system state as $\vec{a}_n = (a_1, \ldots, a_n)$, where a_i is the class of the driver at position i of the queue.

For a set of driver classes $\mathcal{D}$ and a set of rider classes $\mathcal{R}$, define $\alpha(\mathcal{D}) = \sum_{d\in\mathcal{D}} \alpha_d$ and $\beta(\mathcal{R}) = \sum_{r\in\mathcal{R}} \beta_r$ to denote, respectively, the total arrival rate of drivers with classes in $\mathcal{D}$ and of riders with classes in $\mathcal{R}$. Let $R(\mathcal{D})$ denote the set of rider classes compatible with at least one of the driver classes in $\mathcal{D}$, so that, given state $\vec{a}_n$, $R(\vec{a}_i)$ denotes the set of rider classes that are compatible with one or more of the driver classes included in $\vec{a}_i$. The total arrival rate of riders in $R(\vec{a}_i)$ is then $\beta(\vec{a}_i) = \sum_{k\in R(\vec{a}_i)} \beta_k$. Similarly, let $D(\mathcal{R})$ denote the set of driver classes compatible with at least one of the rider classes in $\mathcal{R}$.

Observe that the one-sided matching queue as described above is an order-independent queue, where drivers are customers and the arrival of a compatible rider corresponds to a "service" of a driver. It is easy to verify that $\beta(\vec{a}_n)$ satisfies the conditions of Definition 2.1. Therefore, the stationary distribution of the driver queue state, as well as its quasi-reversibility, follows immediately from Theorem 2.1 provided that the system is stable. Sufficient conditions for stability include (1) a finite acceptance region, (2) $\beta(\mathcal{D}) < \alpha(R(\mathcal{D}))$ for all subsets $\mathcal{D}$, or (3) drivers abandon the queue after an exponentially-distributed time with class-dependent rate γ_i. In the latter case, observe that we still have an order-independent queue, where the overall driver departure (service) rates are now given by $\tilde{\beta}(\vec{a}_n) = \sum_{k=1}^{n} \gamma_{a_k} + \beta(\vec{a}_n)$.

Theorem 2.2 *Consider a one-sided matching queue. Let $\pi(\vec{a}_n)$ denote the stationary probability that the system is in state $\vec{a}_n \in \mathcal{C}$. Assume*

$$G = 1 + \sum_{n=1}^{\infty} \sum_{\vec{a}_n \in \mathcal{S}} \prod_{i=1}^{n} \frac{\alpha_{a_i}}{\beta(\vec{a}_i)} < \infty.$$

Then the system is stable and quasi-reversible, and the stationary distribution is given by

$$\pi\ (\vec{a}_n) = \pi(\varnothing) \prod_{i=1}^{n} \frac{\alpha_{a_i}}{\beta\ (\vec{a}_i)},$$

where $\pi(\varnothing) = 1/G$.

Remark 2.2 We note that the one-sided matching queue can also model multi-class heterogeneous parallel-server systems with bipartite compatibilities and with collaborative FCFS service. In such a system, jobs (customers) of each class arrive according to independent Poisson processes, and service times are exponential, possibly with server-dependent rates. At any given time, all of the jobs present in the system (including those in service) are listed in the queue in order of arrival, and each server is serving the first job with which it is compatible. A job is permitted to be in service at multiple servers simultaneously, in which case it receives service at the sum of those servers' rates. This model is in turn equivalent to a redundancy model in which copies of jobs are dispatched upon arrival to all compatible servers, operating under FCFS, and multiple copies of the same job can be in service at the same time. When any copy completes service, all other copies are canceled, and the job leaves the system. This is known as the "cancel-on-complete" protocol for redundancy systems.

Remark 2.3 The special case of a single-buffer space for each driver class in a one-sided matching queue captures the Adan-Weiss multi-class multi-server loss queue with heterogeneous servers. Here jobs correspond to riders, which are lost if they do not find an idle server, and waiting drivers correspond to idle servers; driver arrivals when the corresponding buffer is full correspond to "dummy services," which are lost. Here FCFM corresponds to "Assign Longest Idle Server (ALIS)" to an arriving compatible job.

2.3 Two-sided matching queues

A natural generalization of the one-sided matching model is the two-sided version, in which both drivers and riders are allowed to wait in the queue, waiting drivers are "served" by arriving compatible riders, and waiting riders are "served" by matching compatible drivers. Apart from the addition of a rider-side queue, the model remains the same as that described in Section 2.2. Our system state is now $(\vec{a}_n; \vec{b}_m)$, where $\vec{a}_n$ and $\vec{b}_m$ are, respectively, the classes of waiting drivers and riders in arrival order. Let $\mathcal{S}^D$ and $\mathcal{S}^R$ denote, respectively, the acceptance regions for the driver-side and the

rider-side; we assume that the acceptance regions on both sides are order-independent (Definition 2.2), so each side is an OI queue. We define the set of all admissible states as $\mathcal{C} = \{(\vec{a}_n; \vec{b}_m) \ : \ \vec{a}_n \in \mathcal{S}^D; \ \vec{b}_m \in \mathcal{S}^R; \ \forall a_i \in \vec{a}_n, a_i \notin R(\vec{b}_m)\}$.

Theorem 2.3 *([9], Theorem 3.13, [8], Theorem 3) Consider a two-sided matching queue. Let $\pi(\vec{a}_n; \vec{b}_m)$ denote the stationary probability that the system is in state $(\vec{a}_n; \vec{b}_m) \in \mathcal{C}$. Assume*

$$G = 1 + \sum_{m,n=1}^{\infty} \sum_{\left(\vec{a}_n;\vec{b}_m\right)\in\mathcal{C}} \left[\prod_{i=1}^{n} \frac{\alpha_{a_i}}{\beta(\vec{a}_i)} \prod_{j=1}^{m} \frac{\beta_{b_j}}{\alpha(\vec{b}_j)} \right] < \infty.$$

Then the system is stable and the stationary distribution satisfies

$$\pi\left(\vec{a}_n; \vec{b}_m\right) = \pi(\varnothing; \varnothing) \prod_{i=1}^{n} \frac{\alpha_{a_i}}{\beta\left(\vec{a}_i\right)} \prod_{j=1}^{m} \frac{\beta_{b_j}}{\alpha(\vec{b}_j)}$$

for all $\left(\vec{a}_n; \vec{b}_m\right) \in \mathcal{C}$, where $\pi(\varnothing; \varnothing) = 1/G$.

Remark 2.4 In Remark 2.2, we saw that the one-sided matching queue can model multi-class heterogeneous parallel-server systems with collaborative service. The two-sided matching queue can also model multi-class heterogeneous parallel-server systems with compatibilities, but now with *non-collaborative* FCFS service. That is, we have Poisson arrivals of jobs of different classes and exponential servers with different speeds, but now the job queue only stores those jobs that are *not* in service at a given time, again in order of arrival, represented by $\vec{a}_n$. At the same time, there is queue with a single-buffer space for each server that stores the idle servers in the order in which they became idle, represented by $\vec{b}_m$. Now "Assign Longest Idle Server (ALIS)" corresponds to FCFM or FCFS for the idle server queue. Busy servers and jobs in service do not appear in the state. For each server k, we have a Poisson arrival process of services at rate β_k, and if the buffer for server k is empty (i.e., server k is busy), then the server joins the idle server queue. If a class-k service arrival occurs when server k's buffer is full (the server is idle), then that service is lost; i.e., it is a "dummy" service. The non-collaborative parallel-server model is in turn equivalent a redundancy model in which copies of jobs are dispatched upon arrival to all compatible servers, operating under FCFS, but now we do not allow multiple copies of the same job to be in service at the same time: when any copy *starts* service, all other copies are canceled. This is known as the "cancel-on-start" protocol for redundancy systems.

2.4 Kelly networks and symmetric queues

A Kelly network [14] is a network of quasi-reversible multi-class queues with Poisson arrivals and probabilistic routing; this is a generalization of Jackson networks [13]. We remind the reader of several theorems in Kelly [14].

Theorem 2.4 *([14], Theorem 3.7 (i) and Theorem 3.12 (i)) For a stable open or closed Kelly network in steady state, the states of individual queues are mutually independent. Thus, the joint stationary distribution is the product of the stationary distributions of the individual queues.*

Constructing a Kelly network requires one to identify queues that are quasi-reversible; any such queue can be included as a node in a Kelly network. An important example that we will rely on in the sections that follow is a *symmetric multi-class queue*, defined as follows.

Definition 2.4 A *symmetric multi-class queue* is one in which the following properties hold, given n customers in the queue:

1. The service requirement of a customer is a random variable whose distribution may depend on the customer's class.
2. A customer that arrives to a system that already contains n customers is inserted into the queue at position $\ell \leqslant n+1$ with probability $\delta(\ell, n+1)$. All customers behind it move back by one position.
3. A proportion $\gamma(\ell, n)$ of the total service rate is directed to the customer in position $\ell, \ell \leqslant n$. When a customer leaves the system, all customers behind it move forward by one position.
4. $\gamma(\ell, n+1) = \delta(\ell, n+1), \ell \leqslant n+1$.

We let $\phi(i)$ denote the total service rate allocated to the first i customers in the queue, for $i \leq n$.

Important examples of symmetric queueing disciplines are processor-sharing, random order of service, and preempt-resume last-come-first-served. Infinite-server queues are also symmetric. In a multi-class symmetric queue, the stationary distribution has a product form and is insensitive to the service-time distribution.

Theorem 2.5 *([14], Theorems 3.8 and 3.10, [5] Theorem) Consider a symmetric multi-class queue in which customers of class $c \in C$ arrive as an independent Poisson process with rate λ_c and experience mean service-time m_c. If $G := \sum_{n=1}^{\infty} \prod_{i=1}^{n} \frac{\rho}{\phi(i)} < \infty$, where $\rho = \sum_{c \in C} \lambda_c m_c$, then the system is stable and the following properties hold:*

1. *The stationary distribution of the number of customers in the system is*

$$\pi(n) = \pi(0) \prod_{i=1}^{n} \frac{\rho}{\phi(i)} \tag{2}$$

   where $\pi(0) = 1/G$.
2. *The queue is quasi-reversible.*
3. *The stationary distribution (2) is insensitive to the service-time distribution.*

Corollary 2.2 *For a multi-class infinite-server queue under the assumptions of Theorem 2.5, the system is always stable and the joint stationary distribution of the numbers*

of customers of each class in the system $\pi(\vec{k}) = \pi(k_1, \ldots, k_{|C|})$ *is*

$$\pi(\vec{k}) = G \prod_{i=1}^{|C|} \frac{(\alpha_i m_i)^{k_i}}{k_i!}$$

where $G = \pi(\varnothing) = \prod_{i=1}^{|C|} e^{-\alpha_i m_i}$.

Proof The multi-class infinite-server queue is equivalent to a collection of $|C|$ independent single-class infinite-server queues. The result then follows with $\phi(i) = i$ for all i for each queue. □

3 Drive-time Distribution Insensitivity

We consider the following queueing network with compatibilities, which includes the loss system studied by Adan and Weiss [1] and Haji and Ross [12], as well as the ride-sharing models of Banerjee et al. [4] as special cases. Our model generalizes that in [4] by incorporating compatibilities between drivers and riders, thereby creating a more accurate model for real-world ride-sharing platforms and other matching models, by e.g., taking preferences into account. Let D and R denote the numbers of classes of drivers and riders respectively. Our model, illustrated in Figure 1, is a network of two nodes for drivers:

1. Node 1 is a one-sided matching queue with an infinite buffer. Drivers of class $i = 1, \ldots, D$ arrive from outside the network as an independent Poisson process with rate α_i and are allowed to wait for riders in the queue in order of arrival. Riders of class $j = 1, \ldots, R$ arrive as an independent Poisson process with rate β_j but are not allowed to queue. An arriving rider either matches with a waiting compatible driver in FCFM order or leaves immediately if there is no compatible driver. Drivers leave node 1 as soon as they are matched to a rider (corresponding to the service of the driver at node 1).
2. Node 2 is an infinite-server queue representing busy drivers, where the mean service (driving) time for class-i drivers is m_i, with rate $\mu_i = 1/m_i$.
3. A driver that is matched with a rider (served) at node 1 leaves node 1 (with their matched rider) and immediately enters node 2.
4. When a class-i driver completes service (finishes driving their rider) at node 2, they leave the system with probability p_i and return to node 1 to wait for another rider with probability $1 - p_i$.

We are now ready to establish insensitivity to the service-time distribution for node 2.

Theorem 3.1 *Let* $\vec{a} = (a_1, \ldots, a_n)$ *denote the state of node 1 (i.e., the classes of drivers in arrival order), and let* $\vec{k} = (k_1, \ldots, k_D)$ *denote the number of drivers of*

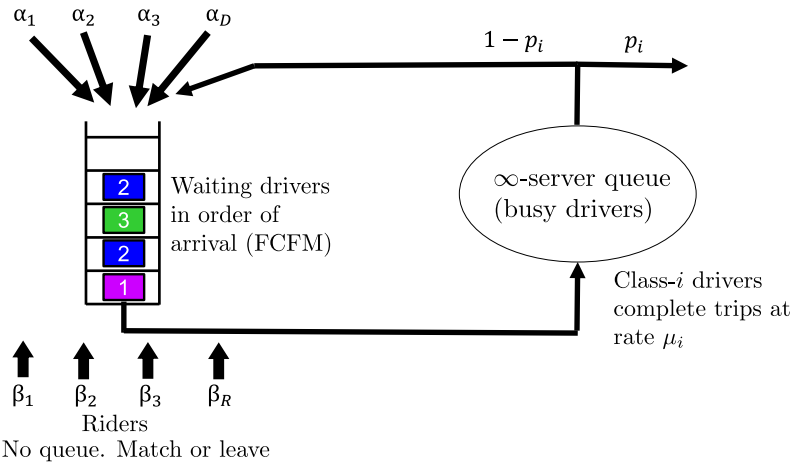

Fig. 1 Our model

each class at node 2. Let

$$G_1 = \sum_{n=1}^{\infty} \prod_{i=1}^{n} \frac{\alpha_{a_i}}{p_i \beta(\vec{a}_i)}, \quad G_2 = \prod_{i=1}^{D} e^{-\frac{\alpha_i}{p_i \mu_i}}.$$

Assume that $G_1 < \infty$. Then the system is stable and the stationary distribution of the joint state of the two nodes is

$$\pi(\vec{a}, \vec{k}) = \pi_1(\varnothing) \prod_{i=1}^{n} \frac{\alpha_{a_i}}{p_i \beta(\vec{a}_i)} \cdot \pi_2(\varnothing) \prod_{i=1}^{D} \left[\left(\frac{\alpha_i}{p_i \mu_i} \right)^{k_i} \frac{1}{k_i!} \right],$$

where $\pi_1(\varnothing) = 1/G_1$ and $\pi_2(\varnothing) = 1/G_2$ denote, respectively, the probabilities that node 1 and node 2 are empty. Moreover, the stationary distribution is insensitive to the drive-time distribution at node 2.

Proof Because node 1 implements FCFM, it is an order-independent queue and is quasi-reversible; the infinite-server queue at node 2 is also quasi-reversible. By Theorem 2.4 the stationary distribution of the system, $\pi(\vec{a}, \vec{k})$, is thus the product of the stationary distributions of the two nodes in isolation. By Theorem 2.5, the stationary distribution is insensitive to the drive-time (service-time) distributions at node 2. Note that, for both node 1 and node 2, the total rate at which class-i drivers arrive is α_i / p_i; the complete stationary distribution then follows from Theorem 2.5 and Corollary 2.2. □

Drive-time insensitivity for the ride-sharing model in Banerjee et al. [4] and service-time insensitivity for the loss model of Adan and Weiss [1]—which are themselves generalizations of the classical insensitivity result for the Erlang loss M/M/k/k model—are now special cases of Theorem 3.1.

Corollary 3.1 *([1], Theorem 3) The Adan-Weiss loss model has stationary distribution*

$$\pi\ (\vec{c}_m) = \pi(\varnothing) \prod_{i=1}^{m} \frac{\mu_{c_i}}{\lambda(\vec{c}_i)},$$

where $\vec{c}_m$ keeps track of the idle servers in the order in which they became idle and $\lambda(\vec{c}_i)$ is the total arrival rate of jobs that are compatible with servers in $\vec{c}_i$. Moreover, the stationary distribution is insensitive to the service-time distribution.

Proof The Adan-Weiss loss system is equivalent to a closed version of our two-node model, where there is exactly one driver of each driver class. Servers in their model are our drivers, with node 1 representing the idle-server queue. Jobs in their model are our riders. Moreover, because the system is closed and consists of only two nodes, it suffices to track only the waiting drivers (idle servers) at node 1. □

Our analytical approach involves modeling a queueing system of interest by constructing an equivalent network with quasi-reversible nodes, some of which are symmetric. This approach can be easily adapted to model a variety of generalizations. For example, we can allow drivers to "abandon" from node 1, meaning that a driver leaves the system if they are not matched with a rider after an exponential time. In this case the node remains an order-independent queue, where the "service" rate now includes the rate of abandonment (see the discussion before Theorem 2.2). We can also add more nodes to the network; additional infinite-server nodes could model, e.g., driver breaks or trips to further destinations, while additional order-independent queues could model, e.g., alternative rider pickup locations. Another application of our model is to a cancel-on-complete redundancy system with compatibilities, which can be seen as an OI one-sided matching queue [9], where users alternate between thinking and waiting for their (redundant) jobs to complete. Theorem 3.1 gives us a product-form stationary distribution of such a system that is insensitive to think-time distributions.

4 Assignment-Policy Insensitivity

Assignment-policy insensitivity, referred to by Gopalakrishnan et al. [11] as "policy-space collapse," is the phenomenon where all position-based assignments of riders to drivers (or service completions to customers in an OI queue) lead to the same steady-state distribution.

Definition 4.1 A policy assigning arriving riders to compatible drivers (service completions to compatible customers) is *position-based* if the policy depends on the relative position (in order of arrival) of a driver (customer), not on its identity (except through its compatibilities).

Examples of position-based policies include first-come-first-matched, last-come-first-matched, random order of matching, etc. Non-position-based policies include class-based priority policies, such as highest-priority-customer-first or fastest-server-first, or state-based policies such as longest-queue first.

4.1 Limited driver buffers

We assume the one-sided matching model of Section 2.2, where riders do not queue, each driver class i has Poisson arrivals at rate α_i and there is a constant total Poisson arrival rate of riders, call it β_{total}; we impose a few additional requirements on the system. First, we assume that there is a buffer of size one for each of the D driver classes, such that if a driver of class i arrives and finds a class-i driver already in the queue, the newly arriving driver is lost. Note that this single-buffer assumption satisfies the conditions for an OI acceptance region, so the driver queue with exponential drive times is an OI queue. Also, in contrast to the model of Section 3, we consider this queue in isolation, not in a network.

Second, we assume that the compatibilities of riders with drivers are exchangeable, as defined below.

Definition 4.2 The compatibility (or class) of a randomly arriving rider with the D drivers can be represented as a binary vector $(X_1, \ldots, X_D)$, where $X_i = 1$ if the rider is compatible with driver i. We say that the compatibilities of riders with drivers are exchangeable if the compatibility vector of a randomly arriving rider, $(X_1, \ldots, X_D)$, is exchangeable, i.e., the joint probability mass function of $X_1, \ldots, X_D$ is the same for all permutations of the drivers. Equivalently, given that a randomly arriving rider is compatible with d drivers, each subset of d drivers is equally likely to be the compatible set.

Observe that the one-sided matching queue described above has the property that the total departure (service) rate for drivers (customers) when there are $n \leq D$ waiting drivers, each of a different class due to the single-buffer assumption, does not depend on the particular driver classes, i.e., $\beta(\vec{a}_n) \equiv \hat{\beta}_n = \beta_{total} P(\sum_{i=1}^{n} X_i > 0)$ depends only on n. Thus, for any position-based policy, we have a PO queue as defined in 2.3, and Corollary 2.1 holds.

Corollary 4.1 *For a one-sided queue with a single-buffer slot for each driver (server) class, and with exchangeable compatibilities of riders (jobs) with drivers (servers), for any position-based driver-assignment policy*

$$\pi(\vec{a}_n) = \pi(\varnothing) \prod_{i=1}^{n} \frac{\alpha_{a_i}}{\hat{\beta}_i}.$$

Furthermore, for a stable two-sided matching queue in which idle drivers and waiting riders can both queue, with one buffer slot for each driver, and an order-independent acceptance region $\mathcal{S}^R$ for riders, we have the following policy insensitivity result for assigning idle drivers. Recall that for two-sided matching queues the set of all admissible states is $\mathcal{C} = \{(\vec{a}_n; \vec{b}_m) : \vec{a}_n \in \mathcal{S}^D;\ \vec{b}_m \in \mathcal{S}^R;\ \forall a_i \in \vec{a}_n, a_i \notin R(\vec{b}_m)\}$ because waiting drivers must be incompatible with the waiting riders, and vice versa.

Corollary 4.2 *Consider a two-sided matching queue in which the compatibilities of riders with drivers are exchangeable, the buffer size for each driver class is 1, and waiting*

riders are assigned to arriving drivers in first-come-first-matched order. Assume

$$G = 1 + \sum_{n=1}^{D} \sum_{m=1}^{\infty} \sum_{\left(\vec{a}_n;\vec{b}_m\right)\in\mathcal{C}} \left[\prod_{i=1}^{n} \frac{\alpha_{a_i}}{\hat{\beta}_i} \prod_{j=1}^{m} \frac{\beta_{b_j}}{\alpha(\vec{b}_j)} \right] < \infty.$$

Then the stationary distribution of the system state is given by

$$\pi(\vec{a}_n;\vec{b}_m) = \pi(\varnothing;\varnothing) \prod_{i=1}^{n} \frac{\alpha_{a_i}}{\hat{\beta}_i} \prod_{j=1}^{m} \frac{\beta_{b_j}}{\alpha(\vec{b}_j)},$$

where $\pi(\varnothing;\varnothing) = 1/G$. Furthermore, this distribution is insensitive to the policy by which arriving riders are assigned to waiting drivers.

Proof For any driver-side state $\vec{a}_n$ the probability that a randomly arriving rider matches with one of the first i idle drivers is independent of the rider-side state $\vec{b}_m$. Furthermore, because the rider-to-driver compatibilities are exchangeable, this probability depends only on i and not on the specific identities of the drivers, hence $\beta(\vec{a}_i) = \hat{\beta}_i$ for any $\vec{a}_i$. The stationary distribution and policy insensitivity follow from Theorem 2.3 and an argument similar to that given in the proof of Corollary 2.1, where now the driver-side acceptance region depends on $\vec{b}_m$. □

Corollary 4.2 generalizes results in Haji and Ross [12] by allowing a rider (job) queue, and Gopalakrishnan et al. [11] by allowing rider-to-driver compatibilities.

This result is consistent with existing work that shows that the stationary distribution in the cancel-on-start system is the same under the assign-longest-idle-server (ALIS, analogous to first-come-first-matched) and random-assignment-to-idle-server (RAIS) policies [2, 15]. The RAIS result requires jobs to be assigned to servers according to a particular probability distribution that results from solving a network flow problem [15]; when the job-to-servers compatibilities are exchangeable, this distribution is uniform. Our result thus offers a generalization, in the case of exchangeable compatibilities, by allowing for any position-based server assignment policy rather than just ALIS and RAIS.

4.2 General buffer sizes: a counterexample

Thus far, we have shown that the one-sided matching queue with exchangeable compatibilities is insensitive to the (position-based) policy that assigns arriving riders to idle drivers, provided that the driver-side buffers are limited to one buffer slot per driver class. It is natural to ask whether this policy insensitivity result continues to hold when we relax the drivers' buffer constraint.

The extension to infinite buffers is motivated by consideration of the cancel-on-complete redundancy-d system, in which each arriving job sends copies to some constant number d servers chosen uniformly at random. Recall that cancel-on-complete parallel-server systems satisfy the conditions for a one-sided OI queue,

where now jobs correspond to drivers, and service completions correspond to rider arrivals. Based on numerical results, Anton and Gardner [3] conjectured that the stationary distribution of the redundancy-d system is indeed insensitive to the job-side scheduling policy. On the other hand, Gast and van Houdt [10] recently showed that in the mean-field limit, the system's mean response time is *not* independent of the job-side scheduling policy.

Our policy's insensitivity results rely on having a single-buffer space per driver (job) class and on the assumption that the rider-to-driver (here, server-to-job) compatibilities are exchangeable; that is, for a fixed set of D driver (job) classes, the compatibility vector, $(X_1, \ldots, X_D)$, of a randomly arriving rider (server completing its service) is exchangeable. Such exchangeability is not true in general for the redundancy-d system. However, in the special case in which there are three servers and $d = 2$, we do have server-to-job exchangeability. Another property of redundancy-d systems is that the arrival rates of all classes of jobs (drivers) are the same. This assumption is not needed for Corollary 4.2, but one would expect it to be necessary with infinite buffers. We thus pose the following conjecture, which extends Corollary 4.2 to the setting in which the driver buffers are unlimited.

Conjecture 4.1 If the rider-to-driver compatibilities in a stable one-sided matching queue are exchangeable, and all driver classes have the same arrival rates, then the stationary distribution is insensitive to the policy by which arriving riders are assigned to waiting drivers, even with general buffers satisfying the OI acceptance region conditions for drivers.

We will show that, surprisingly, this conjecture does *not* hold.

To see this, we will consider the system typically referred to as the M-model (see Figure 2). The M-model has two driver classes (a and b) and three rider classes (1, 2, and 3), where class-1 riders are compatible with class-a drivers only, class-2 riders are compatible with both classes of drivers, and class-3 riders are compatible with class-b drivers only. Each driver class has arrival rate α, and each rider class has arrival rate β. The M-model compatibility structure satisfies the exchangeability condition; a randomly arriving rider's compatibility vector across the two driver classes is exchangeable, and is the simplest nontrivial such model. We assume that there are at least two buffer slots for each driver class, and, in the case of infinite buffers, we assume $2\alpha < 3\beta$ for stability. We will focus on two specific position-based driver-assignment policies: first-come-first-matched (FCFM) and random matching.

Consider the aggregated state space $\vec{k} = (k_a, k_b)$, where k_i denotes the number of class-i drivers in the system. (Note that for our restriction in the previous section, where there is one buffer slot per class, k_i must be either 0 or 1.) We will use $\pi^R(\vec{k})$ and $\pi^F(\vec{k})$ to denote the stationary probability of being in state $\vec{k}$ under random and FCFM respectively. Observe that if Conjecture 4.1 is true, then it must be the case that $\pi^R(\vec{k}) = \pi^F(\vec{k})$ for all states $\vec{k}$. Recall that the stationary distribution for the system under FCFM is given by Theorem 2.1 for the state space $\vec{c} = (c_1, \ldots, c_n)$, where we interpret drivers as being the "customers" in the OI queue. While in general it is combinatorially challenging to aggregate the stationary probabilities of the $\vec{c}$ states to obtain the probability of state $\vec{k}$, this computation is straightforward for individual

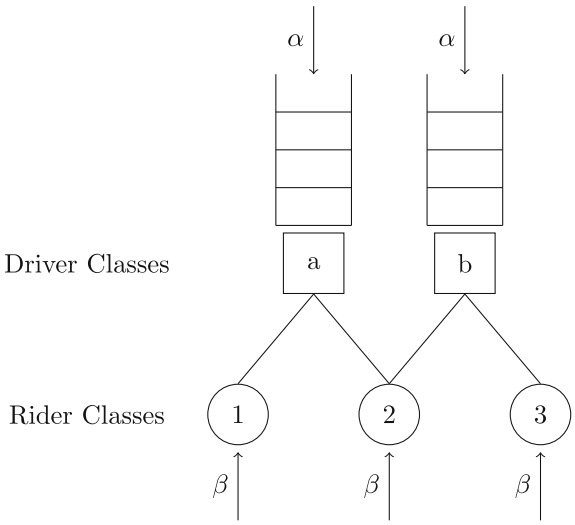

Fig. 2 Compatibility graph of the counterexample

states that are sufficiently small. For clarity in the equations that follow, we will use $\pi^F_{\vec{k}}$ and $\pi^F_{\vec{c}}$ to denote stationary probabilities in the $\vec{k}$ and $\vec{c}$ state spaces respectively.

Consider the state $\vec{k} = (1, 1)$. The balance equation for this state under random is as follows:

$$\pi^R_{\vec{k}}(1,1)(2\alpha+3\beta) = \left(\pi^R_{\vec{k}}(1,0)+\pi^R_{\vec{k}}(0,1)\right)\cdot\alpha + \left(\pi^R_{\vec{k}}(2,1)+\pi^R_{\vec{k}}(1,2)\right)\cdot\left(\beta+\frac{2}{3}\beta\right). \tag{3}$$

If Conjecture 4.1 is true, then the stationary probabilities $\pi^F(\vec{k})$, for FCFM, must also satisfy the above balance equation. We have, from Theorem 2.1:

$$\pi^F_{\vec{k}}(1,1) = \pi^F_{\vec{c}}(a,b)+\pi^F_{\vec{c}}(b,a) = \pi(\varnothing)\left(\frac{\alpha}{2\beta}\cdot\frac{\alpha}{3\beta}+\frac{\alpha}{2\beta}\cdot\frac{\alpha}{3\beta}\right) = \pi(\varnothing)\frac{\alpha^2}{3\beta^2} \tag{4}$$

$$\pi^F_{\vec{k}}(1,0) = \pi^F_{\vec{k}}(0,1) = \pi^F_{\vec{c}}(a) = \pi^F_{\vec{c}}(b) = \pi(\varnothing)\frac{\alpha}{2\beta} \tag{5}$$

$$\pi^F_{\vec{k}}(1,2) = \pi^F_{\vec{k}}(2,1) = \pi^F_{\vec{c}}(a,a,b)+\pi^F_{\vec{c}}(a,b,a)+\pi^F_{\vec{c}}(b,a,a) = \pi(\varnothing)\left(\frac{\alpha}{2\beta}\cdot\frac{\alpha}{2\beta}\cdot\frac{\alpha}{3\beta}+\frac{\alpha}{2\beta}\cdot\frac{\alpha}{3\beta}\cdot\frac{\alpha}{3\beta}+\frac{\alpha}{2\beta}\cdot\frac{\alpha}{3\beta}\cdot\frac{\alpha}{3\beta}\right) = \frac{7\alpha^3}{36\beta^3}. \tag{6}$$

We now substitute (4)-(6) into (3) to verify whether the balance equation holds. On the left-hand side of (3), we have

$$\pi^F_{\vec{k}}(1,1)(2\alpha+3\beta) = \pi(\varnothing)\frac{\alpha^2}{3\beta^2}(2\alpha+3\beta) = \pi(\varnothing)\left(\frac{\alpha^2}{\beta}+\frac{2\alpha^3}{3\beta^2}\right). \tag{7}$$

On the right-hand side of (3), we have

$$\pi(\varnothing)\left(2\alpha\cdot\frac{\alpha}{2\beta}+2\cdot\frac{5\beta}{3}\cdot\frac{7\alpha^3}{36\beta^3}\right) = \pi(\varnothing)\left(\frac{\alpha^2}{\beta}+\frac{35\alpha^3}{54\beta^2}\right). \tag{8}$$

Comparing (7) and (8), we see that the term $\frac{\alpha^2}{\beta}$ appears on both sides of the balance equation; meanwhile, the term $\frac{\alpha^3}{\beta^2}$ is multiplied by different factors on the left- and right-hand sides. Hence, $\pi^F_{\vec{k}}(1,1)$ does *not* satisfy the balance equation for $\pi^R_{\vec{k}}(1,1)$, and so Conjecture 4.1 cannot be true.

While it is surprising that Conjecture 4.1 is false, given earlier empirical results suggesting that FCFM and random yield near-identical performance [3], we can develop some intuition for this result by looking more closely at the stationary distribution under FCFM for the $\vec{c}$ states. Consider a system state with two class-a drivers and one class-b driver. The stationary probability of the permutation (a,a,b) is $\frac{\alpha^3}{2^2\cdot 3\beta^3}$, whereas the permutations (a,b,a) and (b,a,a) each have stationary probability $\frac{\alpha^3}{2\cdot 3^2\beta^3}$. Because not all permutations are equally likely to occur—and, in particular, because the permutation (a,a,b) occurs with higher probability—a class-a driver is chosen with probability greater than $2/3$.

Remark 4.1 Note that if there is a single-buffer slot for each driver class, then the balance equation for state $\vec{k}=(1,1)$ under random is as follows:

$$\pi^R_{\vec{k}}(1,1)\cdot 2\alpha = \pi^R_{\vec{k}}(0,1)\cdot\alpha + \pi^R_{\vec{k}}(1,0)\cdot\alpha.$$

One can readily verify that the stationary probabilities $\pi^F(\vec{k})$, for FCFM, also satisfy the above balance equation. Hence, consistent with our results in Section 4.1, this is not a counterexample when there is a single-buffer slot per driver class.

While the counterexample given above applies to the M-model, a very similar, though more complicated, argument holds for the redundancy-d system with three servers and $d=2$. We can thus conclude that the redundancy-d system is *not* insensitive to the job-side scheduling policy with general job buffers. This result definitively disproves the conjecture of [3], and is consistent with the results of [10].

5 Conclusion

In this paper, we study matching systems, framed in the context of ride-sharing, to identify circumstances under which the system exhibits insensitivity to either the

drive-time distribution or the *assignment policy* by which arriving riders are matched to available drivers. In the one-sided case, wherein drivers are permitted to wait in the queue but riders are not, we show that the stationary distribution is insensitive to the drive-time distribution. We then show that if there is only one buffer space per driver class, then the stationary distribution is insensitive to the assignment rule, even in the two-sided case where riders can queue. Our results generalize those obtained in earlier work by Adan and Weiss [1], Haji and Ross [12], Banerjee et al. [4], and Gopalakrishnan et al. [11]. From a methodological perspective, our contribution is the derivation of simple proofs that leverage Order Independent queues and Kelly networks.

In addition to our insensitivity results, we also present a counterexample illustrating that policy insensitivity does *not* hold when the driver-side buffers have arbitrary (or infinite) capacity, even in the one-sided case where riders do not queue, and even when all driver classes have the same arrival rate; in particular, we show that the stationary distribution does not coincide under FCFM and random assignment. It is natural to ask, then, whether there is an ordering between these two policies: does FCFM always outperform random, as the work of Gast and van Houdt [10] would suggest? Resolving this question remains an open problem at present, and is a major focus of our ongoing work in this space.

Acknowledgements Thanks to Nicolas Gast and Benny van Houdt for helpful discussions about Conjecture 4.1, and to the anonymous reviewers for their valuable feedback on an earlier version of this paper. Part of this work was conducted while Elene Anton was a postdoctoral researcher at Eindhoven University of Technology (TU/e).

Funding Elene Anton's research was partially supported by NWO through the Gravitation project NETWORKS under grant No 024.002.003 and received funding from the European Union's Horizon 2020 research and innovation program under the Marie Sk?odowska-Curie grant agreement No 101034253. Runhan Xie's research and Rhonda Righter's research were partially supported by the Ronald W. Wolff Chancellor's Chair.

References

1. Adan, I., Weiss, G.: A loss system with skill-based servers under assign to longest idle server policy. Probability in the Engineering and Informational Sciences **26**(3), 307–321 (2012)
2. Adan, I., Weiss, G.: A skill based parallel service system under fcfs-alis-steady state, overloads, and abandonments. Stochastic Systems **4**(1), 250–299 (2014)
3. Anton, E., Gardner, K.: The stationary distribution of the redundancy-d model with random order of service. ACM SIGMETRICS Performance Evaluation Review **51**(2), 9–11 (2023)
4. Banerjee, S., Riquelme, C., Johari, R.: Pricing in ride-share platforms: A queueing-theoretic approach. Econometric & Statistical Methods - Special Topics eJournal, Econometrics (2015)

5. Baskett, F., Chandy, K.M., Muntz, R.R., Palacios, F.G.: Open, closed, and mixed networks of queues with different classes of customers. Journal of the ACM (JACM) **22**(2), 248–260 (1975)
6. Berezner, S., Kriel, C., Krzesinski, A.E.: Quasi-reversible multiclass queues with order independent departure rates. Queueing Syst. **19**, 345–359 (1995)
7. Berezner, S., Krzesinski, A.E.: Order independent loss queues. Queueing Syst. **23**(1–4), 331–335 (1996)
8. Comte, C.: Stochastic non-bipartite matching models and order-independent loss queues. Stochastic Models **38**(1), 1–36 (2022)
9. Gardner, K., Righter, R.: Product forms for fcfs queueing models with arbitrary server-job compatibilities: an overview. Queueing Systems **96**(1–2), 3–51 (2020)
10. Gast, N., van Houdt, B.: Approximations to study the impact of the service discipline in systems with redundancy. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, to appear, (2024)
11. Gopalakrishnan, R., Doroudi, S., Ward, A.R., Wierman, A.: Routing and staffing when servers are strategic. Operations Research **64**(4), 1033–1050 (2016)
12. Haji, B., Ross, S.M.: A queueing loss model with heterogeneous skill based servers under idle time ordering policies. Journal of Applied Probability **52**, 269–277 (2015)
13. Jackson, J.R.: Networks of waiting lines. Operations research **5**(4), 518–521 (1957)
14. Kelly, F. P.: *Reversibility and stochastic networks*. Cambridge University Press, (2011)
15. Visschers, J., Adan, I., Weiss, G.: A product form solution to a system with multi-type jobs and multi-type servers. Queueing Systems **70**(3), 269–298 (2012)