Stabilizing Queuing Networks with Model Data-Independent Control

Qian Xie and Li Jin *Member, IEEE*

***Abstract*—This note studies the stability of multi-class queuing networks under a class of centralized or decentralized model data-independent (MDI) control policies, which only depend on traffic state observation and network topology. Control actions include routing, sequencing, and holding. By constructing piecewise-linear test functions, we derive an easy-to-use criterion to check the stability of a multi-class network under a given MDI control policy. For stabilizable multi-class networks, we show that a centralized, stabilizing MDI control policy exists. For stabilizable single-class networks, we further show that a decentralized, stabilizing MDI control policy exists. In addition, for both scenarios, we explicitly construct throughput-maximizing policies and present numerical examples to illustrate the results.**

***Index Terms*—Multi-class queuing networks, Dynamic routing, Lyapunov function, Stability**

I. Introduction

Control on multi-class queuing networks has been studied in numerous contexts of transportation, logistics, and communication systems [1]–[3]. Most existing analysis and design approaches rely on full knowledge of model data, i.e., arrival and service rates, to ensure stability and/or optimality [4]. However, in many practical settings, such data may be unavailable or hard to estimate, and may be varying over time. Such challenges motivate the idea of *model data-independent (MDI)* control policies. MDI control policies select control actions, including routing, sequencing, and/or holding, according to state observation and network topology but independent of arrival/service rates. Such policies are easy to implement and, if appropriately designed, robust against modeling error or non-stationary environment. However, the stability of general open multi-class queuing networks with centralized or decentralized MDI control policies has not been well studied.

In this note, we consider stabilizing multi-class queuing networks with MDI control policies. We study acyclic open queuing networks with Poisson arrivals and exponential service times. Customers (jobs) are classified according to their origin-destination (OD) information. Service rates are independent of customer classes. A network is stabilizable if there exists a control policy that ensures positive Harris recurrence of the queuing process, whether the network is open-loop or closed-loop, centralized or decentralized [5]. By standard results on Jackson networks, stabilizability is equivalent to the existence of a (typically model data-dependent) stabilizing Bernoulli routing policy [6]. We assume that the class-specific arrival rates and the server-specific service rates are unknown to the controller. The main results are as follows:

1) An easy-to-use criterion to check the stability of a multi-class network under a given MDI control policy (Proposition 1).
2) For a multi-class network, a stabilizing centralized MDI control policy exists if and only if the network is stabilizable (Theorem 1).
3) For a single-class network, a stabilizing decentralized MDI control policy exists if and only if the network is stabilizable (Theorem 2).

Previous works on stability of queuing networks are typically based on full knowledge of model data [1], [7]–[9]. So far, the best-studied MDI control policy is the join-the-shortest-queue (JSQ) routing policy for parallel queues [10]–[13], which requires only the queue lengths and does not rely on model data [14]. When and only when the network is stabilizable, i.e., the demand is less than capacity (service rate), the JSQ policy guarantees the stability of parallel queues [15] and the optimality of homogeneous servers [11]. However, JSQ routing does not guarantee stability of more complex networks [14]. MDI routing for general networks has been numerically evaluated [16], but no structural results are available. Decentralized dynamic routing has been considered for single origin-destination networks [17] but not in MDI settings.

To design stabilizing MDI control policies, we first develop a stability criterion (Proposition 1) based on route expansion for queuing networks and explicit construction of a piecewise-linear test function. The expanded network is essentially a parallel connection of all routes from the set of origins to the set of destinations. With this expansion, we use insights about the behavior of parallel queues and of tandem queues to construct the test function and derive the stability criterion. The test function can be used to obtain a smooth Lyapunov function verifying a negative drift condition. The piecewise-linear test function technique was proposed by Down and Meyn [5]; however, their implementation relies on linear programming formulations to determine parameters of the test function, which depends on model data. We will extend this technique to the MDI setting using explicitly constructed test functions.

Based on the stability criterion, we design control policies in centralized and decentralized settings. First, for multi-class networks, we present a stabilizing centralized MDI control policy requiring dynamic routing and sequencing (Theorem 1). The control policy is obtained by minimizing the mean drift of the piecewise-linear test function, and the mean drift is guaranteed to be negative if and only if the network is stabilizable. Second, for single-class networks, we present a decentralized routing and holding policy that guarantees stability (Theorem 2). This result is closely related to the theory on the classical JSQ routing policy [14].

The rest of this paper is organized as follows. Section II defines the multi-class queuing network model. Section III presents the stability criterion based on route expansion and piecewise-linear test function. Section IV and Section V consider the control design problem in centralized and decentralized settings respectively. Section VI gives concluding remarks.

II. Multi-class queuing network

Consider an acyclic network of queuing servers with infinite buffer spaces. Let $\mathcal{N}$ be the set of *servers*. Each server n has an exponential *service rate* $\bar{\mu}_n$. The network has a set $\mathcal{S}$ of *origins* and a set $\mathcal{T}$ of *destinations*. Customers are classified according to their origins and destinations. That is, we can use an origin-destination (OD) pair $(S,T)\in\mathcal{C}$ to denote a *customer class*, or simply *class*. For notational convenience, classes (OD pairs) are indexed by $c=(S_c,T_c)$.

This note was submitted on November XX, 2020. This work is in part supported by US NSF Award CMMI-1949710, the C2SMART University Transportation Center, and the NYU Tandon School of Engineering.

The authors are with the Tandon School of Engineering, New York University, Brooklyn, NY 11201 USA (emails: qianxie@nyu.edu, lijin@nyu.edu).

Customers of class c arrive at S_c according to a Poisson process of rate $\lambda_c \geq 0$. We assume that service rates are independent of customer class.

The *topology* of the network is characterized by *routes* between origins and destinations. We use $|r|$ to denote the number of servers on route r. Let $\mathcal{R}_c$ be the set of routes between S_c and T_c, and define $\mathcal{R} = \bigcup_{c\in\mathcal{C}} \mathcal{R}_c$. Below is an example network to illustrate the notations.

Example 1: Consider the Wheatstone bridge network in Fig. 1. Two classes of customers arrive at S_1 (resp. S_2) with $\lambda_1 > 0$ (resp.

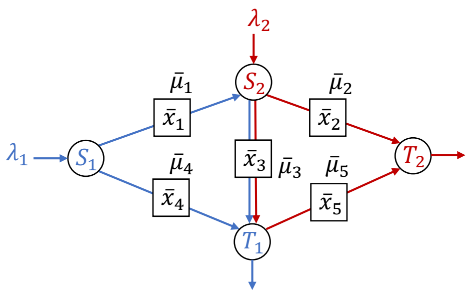

Fig. 1: A two-class queuing network.

$\lambda_2 > 0$). The set of servers is $\mathcal{N} = \{1, 2, \dots, 5\}$ and the set of OD-specific routes are

$$\mathcal{R}_1 = \{(1, 3), (4)\},\ \mathcal{R}_2 = \{(2), (3, 5)\}.$$

The *state* of the network is defined as follows. Let $\bar{x} = [\bar{x}_n^c]_{n\in\mathcal{N},c\in\mathcal{C}}$ be the vector of class-specific *customer numbers*, where $\bar{x}_n^c$ is the number of customers of class c in server n, either waiting or being served. Let $\bar{\mathcal{X}}$ be the space of $\bar{x}$. Let $\bar{y} = [\bar{y}_n]_{n\in\mathcal{N}}$ be the vector of classes of customers under service; i.e., $\bar{y}_n = c$ means that the customer being served in server n is of class c. Let $\bar{\mathcal{Y}}$ be the space of $\bar{y}$. Let $\bar{z} = [\bar{z}_n]_{n\in\mathcal{N}}$ be the vector of holding status. A server n is in *holding* status if a customer has finished service in server n but, instead of being discharged to a downstream server or destination, is held in server n, blocking the other customers in the buffer. $\bar{z}_n = 0$ (resp. $\bar{z}_n = 1$) means server n is not holding (resp. holding). Let $\bar{\mathcal{Z}}$ be the space of $\bar{z}$. Thus, the state of the network is $\bar{\phi} = (\bar{x}, \bar{y}, \bar{z})$. We use $\bar{\Phi}(t) = (\bar{X}(t), \bar{Y}(t), \bar{Z}(t))$ to denote the state of the Markov process at time t and $\bar{\mathbf{\Phi}} = \bar{\mathcal{X}} \times \bar{\mathcal{Y}} \times \bar{\mathcal{Z}}$ to denote the state space.

We consider three types of *control actions*, viz. routing, sequencing, and holding. We assume that all control actions are Markovian (i.e., depending on $\bar{\phi}$) and are applied at the instant of *transitions*, which include the arrival of a customer at an origin or the completion of service at a server. *Routing* refers to allocating an incoming customer to a server downstream to the origin or allocating a customer discharged by a server to another downstream server. *Sequencing* refers to selecting a customer from the waiting queue to serve. The default sequencing policy is the first-come-first-serve (FCFS) policy. For the multi-class setting, we consider preemptive-priority that can terminate an ongoing service and start serving customers from another class, while the customer with incomplete service is sent back to the queue. *Holding* refers to holding a customer who has completed its service in the server, blocking the other customers in the buffer from accessing the server.

Following [18], we say that a queuing network is *stable* if the queuing process is positive Harris recurrent, i.e., there exists a unique invariant measure ν on $\bar{\mathbf{\Phi}}$ such that for every measurable set $D \subseteq \bar{\mathbf{\Phi}}$ with $\nu(D) > 0$ and for every initial condition $\phi \in \bar{\mathbf{\Phi}}$

$$\Pr\{\tau_D < \infty | \Phi(0) = \phi\} = 1,$$

where $\tau_D = \inf\{t \geq 0 : \Phi(t) \in D\}$. For details about the notion of positive Harris recurrence for queuing networks, see [5], [7], [18]. Finally, we say that the network is *stabilizable* if a stabilizing control exists. One can check the stabilizability using the following result:

Lemma 1: An open acyclic queuing network is stabilizable if and only if there exists a vector $[\xi_r]_{r\in\mathcal{R}}$ such that

$$\begin{aligned} &\xi_r \geq 0, \quad \forall r \in \mathcal{R}, \\ &\lambda_c = \sum_{r\in\mathcal{R}_c} \xi_r, \quad \forall c \in \mathcal{C}, \\ &\sum_{r\in\mathcal{R}: n\in r} \xi_r < \bar{\mu}_n, \quad \forall n \in \mathcal{N}. \end{aligned}$$

The proof and implementation are straightforward.

III. Stability Criterion

In this section, we derive a stability criterion for multi-class networks. The techniques that we use include the route expansion of the original network and the explicit construction of a piecewise-linear test function based on the network topology. In Section III-A, we introduce the route expansion of the original network. In Section III-B, we apply a piecewise-linear test function to the expanded network to obtain a stability criterion (Proposition 1) for both the expanded and the original networks.

A. Route expansion

Route expansion refers to the construction of an *expanded network* based on the topology of *original network* (defined in Section II). The specific procedures are:

1) Place all routes $\mathcal{R}$ in the original network in parallel.
2) Add two-way connections between duplicates of servers in the original network.

For example, Fig. 2 shows the expanded network constructed from the original network in Fig. 1.

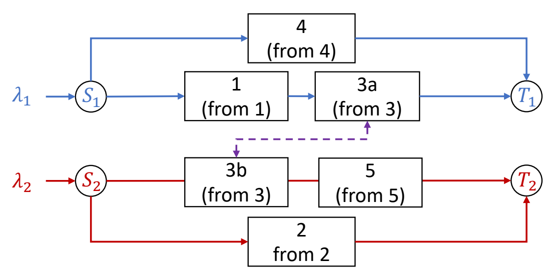

Fig. 2: Route expansion of the network in Fig. 1.

We call "servers" in the expanded network as *subservers*, since they are obtained by duplicating actual servers in the original network. Subservers are indexed by k, $c_k \in \mathcal{C}$ is the class index, $r_k \in \mathcal{R}$ is the route index, and $i_k \in \{1, 2, \dots, |r_k|\}$ is the numbering of subserver k on route r_k. We use $k \in r$ to refer to that subserver k is on route r. Let $\mathcal{K}$ be the set of all subservers and $\mathcal{K}_c$ be the set of subservers with $c_k = c$. We use $n_k \in \mathcal{N}$ to denote the actual server that corresponds to subserver k. In addition, let p_k (resp. s_k) denote the subserver immediately upstream (resp. downstream) to subserver k.

The *state* of the expanded network is $\phi = (x, y, z)$, where $x = \{x_k^c; k \in \mathcal{K}, c \in \mathcal{C}\}$ is the vector of number of class-c customers in subserver k, y_k is the class of the customer under service in k, and z_k is the holding status of k. The expanded state space is $\mathbf{\Phi} = \mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$, where

$$\mathcal{X} = \mathbb{Z}_{\geq 0}^{|\mathcal{C}|\times|\mathcal{K}|},\ \mathcal{Y} = \mathcal{C}^{|\mathcal{K}|},\ \mathcal{Z} = \{0, 1\}^{|\mathcal{K}|}.$$

Two subservers i and j are *duplicating* if $n_i = n_j$. Note that the service rates of duplicating subservers are coupled in the sense that for each server $n \in \mathcal{N}$, at a given time, at most one subserver such that $n_k = n$ can be actively serving customers, or *active*. This can be modeled as an *imaginary service rate control policy* such that the service rate $\mu_k(\phi)$ of subserver k satisfies

$$\sum_{k:n_k=n} \mu_k(\phi) \leq \bar{\mu}_n, \quad \forall \phi \in \mathbf{\Phi}.$$

Hence, the states of the expanded network and the states of the original network are related by

$$\bar{x}_n^c = \sum_{k\in\mathcal{K}:n_k=n} x_k^c, \quad k \in \mathcal{K}, \tag{1a}$$

$$\bar{y}_n = \sum_{k\in\mathcal{K}:n_k=n} \mathbb{I}_{\mu_k(\phi)>0} y_k, \quad k \in \mathcal{K}, \tag{1b}$$

$$\bar{z}_n = \sum_{k\in\mathcal{K}:n_k=n} z_k, \quad k \in \mathcal{K} \tag{1c}$$

for each $n \in \mathcal{N}$. In addition, we define

$$x_k := \sum_{c\in\mathcal{C}} x_k^c, \quad k \in \mathcal{K}.$$

The route expansion technique not only expands the network but also decomposes the state variables. Customers can move through the expanded network using two transition mechanisms. One is the *actual transition*, referring to moving a customer from i (or an origin) to j (or a destination) such that n_j (or the destination) is a downstream server to n_i (or the origin) in the original network. The other is the *imaginary transition* refers to moving a customer from one subserver to a duplicating subserver thereof. Imaginary transitions always occur instantaneously. Note that an actual transition corresponds to a transition in the original network, while an imaginary transition does not; this is also revealed in (1a)–(1c).

One can always map a control action in the expanded network to the original network. However, an MDI control policy may not exist on the state space of the original network; we do need an expanded state space for MDI control. In addition, we allow imaginary control actions in the expanded network, including *imaginary service rate control* and *imaginary switch*; see Section IV. Such imaginary actions only make sense in the expanded network and do not correspond to actual service rate control or switch in the original network.

B. Piecewise-linear test function

Consider a test function defined as follows.

1) For each class $c \in \mathcal{C}$ and each expanded state $x \in \mathcal{X}$, define

$$g_c(x) := \max_{\substack{K_c\subseteq\mathcal{K}_c:\\ \kappa\in K_c\Rightarrow p_\kappa\in\mathcal{K}_c}} \sum_{k\in K_c} a_k x_k,$$

where $a_k \in (0,1)$ is a parameter.

2) Define a piecewise-linear *test function*

$$V(x) := \max_{C\subseteq\mathcal{C}} \sum_{c\in C} b_c g_c(x),$$

where $b_c \in (0,1)$ is a parameter.

We call $V(x)$ the test function rather than the Lyapunov function, since strictly speaking, a smooth Lyapunov function should be developed based on the piecewise-linear test function to verify the Foster-Lyapunov stability criterion. Down and Meyn [5] showed that as long as a piecewise-linear test function can be determined, one can always smooth it to obtain a qualified C^2 Lyapunov function.

Remark 1: The test function may or may not be MDI.

Definition 1 (Dominance): Consider $x \in \mathcal{X}$.

1) A class $c \in \mathcal{C}$ is *dominant* if there exists $C \subseteq \mathcal{C}$ such that $c \in C$ and

$$V(x) = \sum_{\theta\in C} b_\theta g_\theta(\phi).$$

We also call such C the set of dominant classes.

2) A subserver k is *dominant* if $c_k = c$ is dominant and

$$k \in \underset{\substack{K_c\subseteq\mathcal{K}_c:\\ \kappa\in K_c\Rightarrow p_\kappa\in\mathcal{K}_c}}{\operatorname{argmax}} \sum_{k\in K_c} a_k x_k.$$

Let K_c be the set of dominant class-c subservers.

3) A route $r \in \mathcal{R}_c$ is *dominant* if c is dominant and there exists $k \in r$ such that $k \in K_c$. Let R_c be the set of dominant class-c routes.

4) A subserver b is called a *bottleneck* if $c_b = c$ is dominant, $b \in K_c$ and $s_b \notin K_c$.

Remark 2: Essentially, a route or server is dominant if changes in its traffic state immediately affect the test function V.

A *regime* X of the piecewise-linear test function is a subset of $\mathcal{X}$ such that there exist $C^X \subseteq \mathcal{C}$, $R^X = \bigcup_{c\in C^X} R_c^X \subseteq \mathcal{R}$, and $K^X = \bigcup_{c\in C^X} K_c^X \subseteq \mathcal{K}$ such that C^X, R_c^X, K_c^X are dominant for each $x \in X$. That is, the test function is linear over X. Let $\mathscr{X}$ be the set of regimes; note that $\bigcup_{X\in\mathscr{X}} X = \mathcal{X}$.

Definition 2 (Mean velocity and drift): Consider a state $\phi = (x,y,z) \in \mathbf{\Phi}$.

1) The *mean velocity* at state ϕ is a function $\Delta : \mathbf{\Phi} \to \mathbb{R}^{|\mathcal{K}|}$ such that for each $k \in \mathcal{K}$,

$$\Delta_k(\phi) := \sum_{c\in\mathcal{C}} \lambda_c \pi_{S_c,k}^c(x) + \sum_{c\in\mathcal{C}} \mu_{p_k}(\phi)\pi_{p_k,k}^c(x) - \mu_k(x).$$

where $\pi_{S_c,k}^c$ (resp. $\pi_{p_k,k}^c$) is the probability that a class-c customer is routed from origin S_c (resp. subserver p_k) to subserver k.

2) Given $X \in \mathscr{X}$ such that $x \in X$, the *mean drift over* X is given by

$$D^X(\phi) := \sum_{c\in C^X} b_c \sum_{k\in K_c^X} a_k \Delta_k(\phi).$$

Remark 3: In our subsequent analysis, the mean drift $D^X(\phi)$ of the test function will play the role of the infinitesimal generator applied to a Lyapunov function; see [5] for the connection between the test function and the Lyapunov function.

The main result of this section is as follows:

Proposition 1: Consider a multi-class network under an expanded control policy. Suppose that there exist constants $\alpha \in (0,1)$, $\beta \in (0,1)$, $M < \infty$, and $\epsilon > 0$ such that, for each $\phi = (x,y,z)$ and each $X \in \mathscr{X}$ where $|x| = \sum_{k\in\mathcal{K}} x_k > M$ and $x \in X$,

$$\sum_{c\in C^X} b_c \sum_{k\in K_c^X} a_k \Delta_k(\phi) \leq -\epsilon. \tag{2}$$

Then, the network is stable.

Proof. Consider the test function $V(x)$. By its definition, if $x \in X$, we have

$$V(x) = \sum_{c\in C^X} b_c \sum_{k\in K_c^X} a_k x_k.$$

The mean drift is given by

$$\begin{aligned} D^X(\phi) &= \sum_{c\in C^X} b_c \sum_{k\in K_c^X} a_k \Delta_k(\phi) \\ &\overset{(2)}{\leq} -\gamma^{|C^X|-1}\epsilon \leq -\gamma^{|\mathcal{C}|-1}\epsilon, \quad \phi : |x| > M. \end{aligned}$$

One can then apply [5, Theorem 1] and [5, Lemma 5] to obtain the stability of the network. □

Compared to the existing result [5, Theorem 1], the main contribution of Proposition 1 is a parametrized construction of the test function. We use insights on network topology to significantly reduce the search space for the test function. For example, in the subsequent sections we will show that the parameters can be easily constructed. Furthermore, the proposed test function allows the development of structural results (Theorem 1–2).

IV. CENTRALIZED CONTROL FOR MULTIPLE CLASSES

In this section, we show that the "join-the-shortest-route (JSR)" policy, a joint routing and sequencing policy, is stabilizing if and only if the network is stabilizable.

Construct the test functions as follows.

1) For each class $c \in \mathcal{C}$, each route $r \in \mathcal{R}_c$, and each expanded state $x \in \mathcal{X}$, let

$$f_r(x) := \max_{k \in r} \alpha^{i_k - 1} \sum_{j: i_j \le i_k} x_j,$$

$$g_c(x) := \max_{R_c \subseteq \mathcal{R}_c} \beta^{|R_c|-1} \sum_{r \in R_c} f_r(x),$$

where $\alpha \in (0,1), \beta \in (0,1)$ are constant parameters.

2) The piecewise-linear *test function* is given by

$$V(x) := \max_{C \subseteq \mathcal{C}} \gamma^{|C|-1} \sum_{c \in C} g_c(x),$$

where $\gamma \in (0,1)$ is a constant parameter.

Let the parameters be such that

$$\alpha^{n-1} \ge \frac{1}{n}, \quad n = 1, 2, \dots, |\mathcal{K}|, \tag{3a}$$

$$\beta^{n-1} \ge \frac{1}{n}, \quad n = 1, 2, \dots, |\mathcal{R}|, \tag{3b}$$

$$\gamma^{n-1} \ge \frac{1}{n}, \quad n = 1, 2, \dots, |\mathcal{C}|, \tag{3c}$$

and follow the notions of dominance accordingly (see Definition 1). Note that such MDI parameters α, β, γ always exist. The control that we will consider in this subsection only depends on α, β, γ and is thus MDI. Specifically, we consider the JSR policy defined as follows:

Definition 3 (Join-the-shortest-route (JSR) policy):

1) (Routing) At an origin S, an incoming customer of class c is allocated to the route $r \in \mathcal{R}_c$ such that

$$f_r(x) = \min_{r' \in \mathcal{R}_c} f_{r'}(x).$$

In case of multiple minima, let i_r be such that

$$i_r = \min i_b \quad s.t.\ f_r(x) = \alpha^{i_b - 1} \sum_{\ell: i_\ell \le i_b} x_\ell.$$

Then, an incoming customer of class c is allocated to the route $r \in \mathcal{R}_c$ with the largest i_r, which is denoted by i_c^*. Further ties are broken uniformly at random.

2) (Imaginary service rate control) Let $\mathcal{K}_n$ be the set of subservers corresponding to server n and let $\mathcal{B}$ be the set of bottlenecks for a given x. Then, a subserver $k \in \mathcal{K}_n$ is activated if $k \in \mathcal{B}$. If multiple subservers are in $\mathcal{K}_n \cap \mathcal{B}$, then activate the subserver k^* such that

$$i_{k^*} = \min i_k \quad s.t.\ k \in \mathcal{K}_n \cap \mathcal{B};$$

ties are broken uniformly at random. Note that such control actions are equivalent to class-based sequencing in the original network.

The main result of this section is the following:

Theorem 1 (Stability of JSR policy): The JSR policy stabilizes a multi-class network if and only if the network is stabilizable.

Note that one can easily check the stabilizability and maximize the throughput of a network with Lemma 1.

In the rest of this section, we apply Theorem 1 to study the stability of the Wheatstone bridge network under the JSR policy (Subsection IV-A) and then prove this theorem (Subsection IV-B).

A. Numerical Example

Consider the network in Fig. 1 and suppose that $\lambda_1 = \lambda_2 = \lambda = 1$ and $\bar{\mu}_n = \mu = 1$ for $n = 1, 2, 4, 5$ and $\bar{\mu}_3 = \frac{1}{4}$. This example is for illustrating the route expansion and the test function construction.

Note that under the above model parameters, the decentralized JSQ policy is destabilizing. To see this, $\bar{\mu}_1 = \bar{\mu}_4$ implies that on average, class-1 customers are evenly distributed between server 1 and server 4. Thus, the average departure rate of class-1 customers from server 1 is $\frac{1}{2}$, which exceeds the service rate of server 3. Therefore, the queue at server 3 is unstable. The main reason that the JSQ policy is destabilizing is the ignorance of downstream congestion. As $\bar{X}_3(t)$ gets large, a reasonable action is to allocate fewer class-1 customers to server 1. However, the myopic nature of the JSQ policy disallows such far-sighted decisions.

An alternative centralized stabilizing routing policy can be the JSR policy:

1) A class-1 customer arriving at S_1 is routed to server 1 if $\bar{X}_1^1(t) + \bar{X}_3^1(t) < \bar{X}_4^1(t)$, to server 4 if $\bar{X}_1^1(t) + \bar{X}_3^1(t) > \bar{X}_4^1(t)$, and uniformly at random otherwise.
2) A class-2 customer arriving at S_2 is routed to server 3 if $\bar{X}_3^2(t) + \bar{X}_5^2(t) < \bar{X}_2^2(t)$, to server 2 if $\bar{X}_3^2(t) + \bar{X}_5^2(t) > \bar{X}_2^2(t)$, and uniformly at random otherwise.
3) The dominant class has a higher priority.

That is, when customers are routed at S_1, the decision is based on not only the local state ($\bar{X}_1(t)$ and $\bar{X}_4(t)$), but also the state further downstream ($\bar{X}_3(t)$); this mechanism resolves the myopic problem of the JSQ policy. Note that this policy is MDI.

The expanded network is shown in Fig. 2. Each block in the figure represents a subserver. In particular, subservers 3a and 3b are decomposed from server 3; the other servers are remained. Solid arrows correspond to actual transitions in an original network, while dashed arrows correspond to imaginary transitions between duplicating subservers.

In the expanded network, a customer can move along both solid and dashed arrows. The color of an arrow shows which class can move along it: blue means class (S_1, T_1), red means (S_2, T_2), and purple means both. For ease of presentation, we label (S_1, T_1) as class 1 and (S_2, T_2) as class 2. For example, a customer of class (S_1, T_1) can visit subservers 4, 1, 3a, 3b and the destination T_1.

We generalize the JSR policy in the expanded network as follows.

1) A class-1 customer arriving at S_1 is routed to subserver 4 if $X_4(t) < X_1(t) + X_{3a}(t)$, to subserver 1 if $X_4(t) > X_1(t) + X_{3a}(t)$, and uniformly at random otherwise.
2) A class-2 customer arriving at S_2 is routed to subserver 3b if $X_{3b}(t) + X_5(t) < X_2(t)$, to subserver 2 if $X_{3b}(t) + X_5(t) > X_2(t)$, and uniformly at random otherwise.
3) If subserver 3a (resp. 3b) is dominant but server 3 is serving a class-2 (resp. class-1) customer, we switch the customer being served in 3b (resp. 3a) with a customer in 3a (resp. 3b). That is, for an imaginary switch between $3a$ and $3b$ at time t, we have $Y_{3a}(t) = Y_{3b}(t_-)$ and $Y_{3b}(t) = Y_{3a}(t_-)$. Consequently, 3a (resp. 3b) will be actively serving the switched customer. Such

imaginary transitions happen as soon as such a configuration occurs and do not affect the value of the test function.

By Theorem 1, the network can be stabilized by the JSR policy if and only if

$$\lambda_1 < 2,\ \lambda_2 < 2,\ \lambda_1 + \lambda_2 < \frac{9}{4}.$$

We use the following parameters for the test function:

$$\alpha = \beta = \gamma = \frac{3}{4},\ \epsilon = \Big(\frac{3}{4}\Big)^5.$$

One can verify that the above parameters satisfy (3a)–(3c) and Proposition 1 by considering the following cases:

1) Only one route is dominant. In this case, an incoming customer is always allocated to a non-dominant route, leading to non-positive contribution to the mean drift:
$$D^X(\phi) \le -\gamma\beta\alpha\mu = -\Big(\frac{3}{4}\Big)^3 \le -\epsilon.$$
2) Two routes with different OD pairs are dominant. This case is analogous to the previous case:
$$D^X(\phi) \le -\gamma\beta\alpha\mu = -\Big(\frac{3}{4}\Big)^3 \le -\epsilon.$$
3) Two routes with the same OD pair or more than two routes are dominant. In such cases, the mean drift satisfies
$$D^X(\phi) \le \gamma\beta^3(\lambda - \mu - \alpha\mu) = -\Big(\frac{3}{4}\Big)^5 \le -\epsilon.$$

Consequently, the network is stable under the MDI JSR policy.

B. Proof of Theorem 1

When analyzing the mean drift, we consider two parts: external arrivals and internal transmission. We first show that any internal transmission does not positively contribute to the mean drift and then show that any positive contribution from external arrivals can always be compensated by internal transmissions.

1) Internal transmissions: Note that under the JSR policy, every job remains on the route assigned to the job when it enters the network. Hence, internal transmissions only occur between subservers on the same route.

Given $\phi = (x, y, z)$, consider an internal transmission from subserver k to subserver j; this implicitly requires $x_k \ge 1$. The definition of dominance ensures that if j is dominant, then so is k. Hence, we need to consider the following cases:

1) If k and j are both dominant, the transmission leads to zero contribution to the mean drift $D^X(\phi)$ for all X such that $x \in X$.
2) If k is dominant and j is non-dominant, the transmission leads to the following contribution to the mean drift:
$$-\alpha^{i_k-1}\mu_k(\phi) \le 0.$$

Hence, internal transmissions never lead to positive contribution to the mean drift.

2) External arrivals: Given $x \neq 0$, consider a regime $X \in \mathscr{X}$ such that $x \in X$. For each $c \in C$, the JSR policy ensures that if there exists a non-dominant route in $\mathcal{R}_c$, then an incoming customer must be allocated to a non-dominant route in $\mathcal{R}_c$, leading to non-positive contribution to the mean drift. Hence, we only need to consider dominant classes c such that every route in $\mathcal{R}_c$ is dominant, i.e. $R_c^X = \mathcal{R}_c$. Let C^* be the set of such classes; hence $C^* \subseteq C$. The part of the mean drift associated with $c \in C^*$ satisfies

$$D_c^X(\phi) \le \gamma^{|C|-1}\beta^{|\mathcal{R}_c|-1}\Big(\alpha^{i_c^*-1}\lambda_c - \sum_{b\in\mathcal{B}^X:c_b=c}\alpha^{i_b-1}\mu_b(\phi)\Big) \tag{4}$$

over any regimes of the piecewise-linear test function, where i_c^* is given in Definition 3.

The right side of (4) is based on the following result.

Lemma 2: For $x \neq 0$, a bottleneck b must be non-empty, i.e.

$$x_b \ge 1. \tag{5}$$

Proof. First, consider the case that $i_b \ge 3$. Since b is a bottleneck, we have

$$\alpha^{i_b-1}\sum_{k\in r_b:i_k\le i_b} x_k \ge \alpha^{i_b-2}\sum_{k\in r_b:i_k\le i_b-1} x_k,$$

which implies

$$x_b \ge \frac{1-\alpha}{\alpha}\sum_{k:i_k\le i_b-1} x_k.$$

Since $x \neq 0$ and r_b is dominant, we have

$$\sum_{k:i_k\le i_b-1} x_k > 0.$$

The above two inequalities jointly imply (5). Finally, the proof for b such that $i_b \le 2$ is analogous to the above. □

To analyze the right side of (4), we need the following result.

Lemma 3: Under the JSR policy, for each $\phi \in \mathbf{\Phi}$, let $\mathcal{B}$ be the set of bottlenecks. Then

$$\sum_{b\in\mathcal{B}^X}\mu_b(\phi) = \sum_{n_b:b\in\mathcal{B}^X}\bar{\mu}_{n_b}. \tag{6}$$

Proof. The statement of this lemma is equivalent to that every bottleneck should be active, unless a duplicate thereof is also a bottleneck and is active. This is ensured by the imaginary service rate control of the JSR policy and Lemma 2. □

Then, to show the sufficiency, note that the total partial mean drift associated with C^* is

$$\begin{aligned}
&\sum_{c\in C^*} D_c^X(\phi)\\
&\le \gamma^{|C|-1}\sum_{c\in C^*}\beta^{|\mathcal{R}_c|-1}\Big(\alpha^{i_c^*-1}\lambda_c - \sum_{b\in\mathcal{B}^X:c_b=c}\alpha^{i_b-1}\mu_b(\phi)\Big)\\
&\le \gamma^{|C|-1}\sum_{c\in C^*}\beta^{|\mathcal{R}_c|-1}\alpha^{i_c^*-1}\Big(\lambda_c - \sum_{b\in\mathcal{B}^X:c_b=c}\mu_b(\phi)\Big)\\
&\overset{(6)}{=} \gamma^{|C|-1}\sum_{c\in C^*}\beta^{|\mathcal{R}_c|-1}\alpha^{i_c^*-1}\Big(\lambda_c - \sum_{n_b:b\in\mathcal{B}^X,c_k=c}\bar{\mu}_{n_b}\Big)\\
&\le \gamma^{|C|-1}\sum_{c\in C^*}\beta^{|\mathcal{R}_c|-1}\alpha^{i_c^*-1}\Big(\lambda_c - \sum_{n\in\mathcal{N}_c}\bar{\mu}_n\Big),
\end{aligned}$$

where $\mathcal{N}_c$ is the min-cut of the original network of class c. If the network is stabilizable, by Lemma 1, we have

$$\lambda_c - \sum_{n\in\mathcal{N}_c}\bar{\mu}_n < 0.$$

Hence, noting that internal transmissions lead to non-positive contributions to the mean drift, we have

$$D^X(\phi) \le \sum_{c\in C^*} D_c^X(\phi) < 0,$$

which implies stability.

Finally, the necessity is apparent: if a network is not stabilizable, then there exists no MDI control that can stabilize the network.

V. Decentralized control for a single class

For a single-class network, we can drop the class index and use x_k to denote the number of customers in subserver k. Note that such network has a single origin and a single destination. Again we can do route expansion on such network.

We consider a decentralized MDI control policy as follows.

Definition 4 (JSQ with artificial spillback): The JSQ with artificial spillback (JSQ-AS) policy is as follows:

1) (Routing) A discharged customer is routed to the shortest downstream queue, with ties broken uniformly at random.
2) (Holding) For each subserver k, any customer who has finished the service will be held if and only if $X_{s_k}(t) \geq X_k(t)$.
3) (Imaginary switch) If a dominant subserver k is inactive but its non-dominant duplicate k' is active, and both are not in the holding status, we switch the customer being served in k' with a customer in k.

The JSQ-AS policy is decentralized in the sense that control actions on subserver k only depend on local traffic information: number of customers in duplicate subservers $\{x_{k'} : n_k = n_{k'}\}$ and that in immediate downstream subservers $\{x_{s_{k'}} : n_k = n_{k'}\}$. A key characteristic of such policies is that congestion information can propagate through the network via the forced holding: if a subserver becomes congested (i.e. x_k gets large), the congestion will propagate to the upstream subservers in a cascading manner ("artificial spillback"). Importantly, such artificial spillback does not undermine throughput like the natural spillback caused by the limited buffer size. The reason is that although congestion can propagate, the queue size in any downstream subserver is not upper-bounded. Artificial spillback is the main difference between the JSQ-AS policy and the classic JSQ policies. Note that although the JSQ-AS policy is constructed based on the expanded network, its actions can always be converted to the ones in the original network. Importantly, the decentralized control in the expanded network must also be decentralized in the original network. Also note that the imaginary switch has no impact on the original network or the test function.

The main result of this section is as follows:

Theorem 2 (Stability of JSQ-AS policy): For the route expansion of a single-class network, the JSQ-AS policy is stabilizing if and only if

$$\lambda < \bar{\mu}^{mc}, \tag{7}$$

where $\bar{\mu}^{mc}$ is the min-cut service rate of the original network.

This theorem implies that JSQ-AS policy is also a throughput-maximizing policy since we allow any throughput that satisfies (7).

In the rest of this section, we apply Theorem 2 to study the stability of the Wheatstone bridge network under the JSQ-AS policy (Subsection V-A) and then prove this theorem (Subsection V-B).

A. Numerical Example

Consider the original network with route expansion in Fig. 3. Again suppose that $\lambda = 1$, $\bar{\mu}_n = \frac{3}{4}$ for $n = 1, 2, \cdots, 5$. Similarly, with the above parameters, the JSQ policy is destabilizing since the queue at server 5 is unstable. However, in the decentralized setting, the control actions can only depend on the local state, say the routing decision at the origin can be based on $\bar{X}_1(t)$ and $\bar{X}_4(t)$, but not $\bar{X}_5(t)$. A remedy is to introduce the holding policy (artificial spillback) to the JSQ policy so that the downstream congestion can be relieved and the local state can somehow reflect the states further downstream.

In the expanded network, server 1 is decomposed into subserver 1a and 1b, server 5 is decomposed into subserver 5a and 5b. The states in the original network and those in the expanded network satisfy $\bar{X}_1(t) = X_{1a}(t) + X_{1b}(t)$ and $\bar{X}_5(t) = X_{5a}(t) + X_{5b}(t)$.

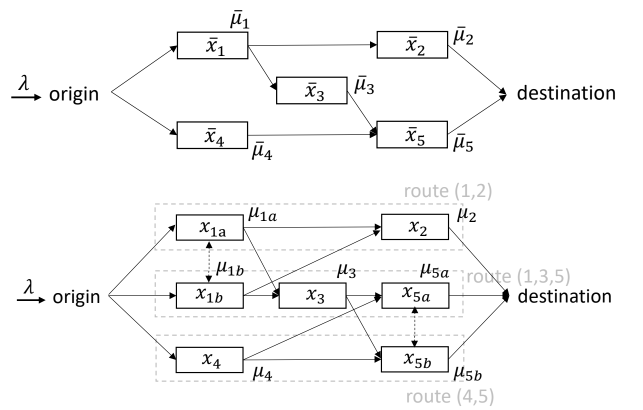

Fig. 3: A single-class queuing network and its expanded network.

The initial states of the expanded network can be not unique. Say the initial queue size of server 1 is 2, then the initial queue sizes of subserver 1a and 1b can be 2, 0 or 1, 1 or 0, 2 respectively. Then the states are updated based on the model and our JSQ-AS policy. For example, the routing decision at the origin is based on $X_{1a}(t)$, $X_{1b}(t)$ and $X_4(t)$ rather than $\bar{X}_1(t)$ and $\bar{X}_4(t)$; a customer who has just finished the service at server 3 will be held if $X_{5a}(t) \geq X_3(t)$, once released, it will be routed to the shorter downstream queue by comparing $X_{5a}(t)$ and $X_{5b}(t)$.

B. Proof of Theorem 2

To prove Theorem 2, we use the following explicitly constructed piecewise-linear test function

$$V(x) := \max_{\substack{K \subseteq \mathcal{K}: \\ \kappa \in K \Rightarrow p_\kappa \in \mathcal{K}}} \left\{ \frac{1 + (|K| - 1)\delta}{|K|} \sum_{k \in K} x_k \right\},$$

where $\delta > 0$ is a small value.

The following lemmas characterize several key properties of the JSQ-AS policy, which is the basis for the proof.

Lemma 4: Under the holding policy, the process $\{X(t); t \geq 0\}$ admits an invariant set $\mathcal{Q} \subseteq \mathcal{X}$ given by

$$\mathcal{Q} := \{x \in \mathcal{X} : x_{s_k} \leq x_k, k \in \mathcal{K}\}.$$

Since we consider the long-time stability of the network, it suffices to consider the states in an invariant set. The above result indicates that in the invariant set $\mathcal{Q}$, the queue size of any subserver is upper-bounded by the queue size of its immediate upstream subserver. The proof is straightforward.

In the rest of this subsection, we consider the regime $X \subseteq \mathcal{Q}$ containing x.

Lemma 5: A bottleneck can not be in the holding status.

Proof. Otherwise, the bottleneck must have at least one downstream subserver. By Lemma 4, $x_{s_k} \geq x_b$. Since b is a bottleneck, we have

$$\frac{1 + (|K^X| - 2)\delta}{|K^X| - 1} \sum_{k \in K^X : k \neq b} x_k \leq \frac{1 + (|K^X| - 1)\delta}{|K^X|} \sum_{k \in K^X} x_k, \tag{8}$$

which implies

$$(1 - \delta) \sum_{k \in K^X} x_k \leq |K^X| [1 + (|K^X| - 2)\delta] x_b. \tag{9}$$

Since $x_b \leq x_{s_k}$, we have

$$(1 - \delta) \sum_{k \in K^X} x_k < |K^X| [1 + |K^X| \delta] x_{s_k},$$

which is equivalent to

$$\frac{1+(|K^X|-1)\delta}{|K^X|}\sum_{k\in K^X} x_k < \frac{1+|K^X|\delta}{|K^X|+1}\Big(\sum_{k\in K^X} x_k + x_{s_k}\Big),$$

contradicting with the fact that subserver b is dominant and subserver s_k is non-dominant. □

Remark 4: A bottleneck b must be non-empty, i.e. $x_b > 0$. This can be directly derived from (9).

Lemma 6: Let k_r^1 be the first subserver on route r, then either the route with the smallest $x_{k_r^1}$ is non-dominant or every route is dominant.

Proof. If there is only one route, then that route must be dominant. Now assume there are at least two routes and route $\hat{r}$ has the smallest $x_{k_r^1}$, i.e. $\forall r \in \mathcal{R}$, $x_{k_{\hat{r}}^1} \le x_{k_r^1}$. Suppose $k_{\hat{r}}^1 \in K^X$ and $\exists\, r \in \mathcal{R}$ s.t. $k_r^1 \notin K^X$. Note that by Lemma 4, $x_b \le x_{k_{\hat{r}}^1} \le x_{k_r^1}$, then from (8) we have

$$\frac{1+(|K^X|-1)\delta}{|K^X|}\sum_{k\in K^X} x_k < \frac{1+|K^X|\delta}{|K^X|+1}\Big(\sum_{k\in K^X} x_k + x_{k_r^1}\Big),$$

contradicting with our supposition. Therefore, we can conclude that either $\hat{r}$ is non-dominant or every route in $\mathcal{R}$ is dominant. □

Lemma 7: If $\phi \in \mathbf{\Phi}$ makes every route $r \in \mathcal{R}$ dominant, then we have

$$\sum_{k\in\mathcal{B}^X}\mu_k(\phi) = \sum_{n:n=n_k,k\in\mathcal{B}^X}\bar{\mu}_n \tag{10}$$

Proof. The statement is equivalent to that a bottleneck should be active, unless a duplicate thereof is dominant and active.

To show this, note that once there exists an inactive bottleneck k and a non-dominant but active duplicate subserver k', the imaginary switch mechanism will move the customer being served in k' to k and move one customer in k to k'. This is allowed since both k and k' contain at least one customer due to the fact that a customer being served in k' and the bottleneck k must be non-empty. □

Similar to the proof of Theorem 1, we first analyze the internal transmissions and then external arrivals.

1) Internal transmissions: In the proof of Theorem 1, we have already discussed the case where internal transmissions between subservers are on the same route. However, unlike the JSR policy, the JSQ-AS policy allows internal transmissions between subservers on different routes. Hence, we also need to consider the internal transmission from subserver k to subserver j where $r_k \neq r_j$.

The definition of dominance ensures that if k is non-dominant, so is s_k. According to the routing policy, $x_{s_k} \ge x_j$. Let ℓ be the first non-dominant subserver on route r_k and b be the bottleneck on route r_j. If j is non-dominant, then by Lemma 4, we have $x_\ell \ge x_{s_k} \ge x_j \ge x_b$. Now from (8) we can obtain

$$\frac{1+(|K^X|-1)\delta}{|K^X|}\sum_{k\in K^X} x_k < \frac{1+|K^X|\delta}{|K^X|+1}\Big(\sum_{k\in K^X} x_k + x_\ell\Big),$$

contradicting with the definition of dominant subservers.

Thus, it cannot be the case that k is non-dominant and j is dominant, which implies that any internal transmission does not positively contribute to the mean drift.

2) External arrivals: According to Lemma 6, if a non-dominant route exists, then the routing policy guarantees that an arriving customer must be routed to the first subserver on a non-dominant route r; this leads to non-positive contribution to the mean drift. Otherwise, every route is dominant. Then by Lemma 5 and Remark 4, for any $x \in \mathcal{Q}$ ($x \neq 0$), the drift satisfies

$$\begin{aligned} D^X(\phi) &\le \frac{1+(|K^X|-1)\delta}{|K^X|}\Big(\lambda - \sum_{b\in\mathcal{B}^X}\mu_b(\phi)\Big) \\ &\overset{(10)}{=} \frac{1+(|K^X|-1)\delta}{|K^X|}\Big(\lambda - \sum_{n:n=n_k,k\in\mathcal{B}^X}\bar{\mu}_n\Big) \\ &\overset{\text{Lemma 1}}{<} 0, \end{aligned}$$

which completes the proof. □

VI. Concluding remarks

We study the stability of open queuing networks under a class of model data-independent control policies. We derive an easy-to-use stability criterion based on route expansion of the network and explicit piecewise-linear test functions. In addition, we generalize the classical join-the-shortest-queue policy to centralized/decentralized settings and attain maximum throughput. Our results provide insights for robust and secure control of queuing networks.

References

[1] P. Kumar and S. P. Meyn, "Stability of queueing networks and scheduling policies," *IEEE Transactions on Automatic Control*, vol. 40, no. 2, pp. 251–260, 1995.

[2] S. P. Meyn, "Sequencing and routing in multiclass queueing networks part i: Feedback regulation," *SIAM Journal on Control and Optimization*, vol. 40, no. 3, pp. 741–776, 2001.

[3] S. L. Smith, M. Pavone, F. Bullo, and E. Frazzoli, "Dynamic vehicle routing with priority classes of stochastic demands," *SIAM Journal on Control and Optimization*, vol. 48, no. 5, pp. 3224–3245, 2010.

[4] D. Bertsimas, I. C. Paschalidis, and J. N. Tsitsiklis, "Optimization of multiclass queueing networks: Polyhedral and nonlinear characterizations of achievable performance," *The Annals of Applied Probability*, pp. 43–75, 1994.

[5] D. Down and S. P. Meyn, "Piecewise linear test functions for stability and instability of queueing networks," *Queueing Systems*, vol. 27, no. 3-4, pp. 205–226, 1997.

[6] R. G. Gallager, *Stochastic processes: theory for applications*. Cambridge University Press, 2013.

[7] J. G. Dai, "On positive Harris recurrence of multiclass queueing networks: A unified approach via fluid limit models," *The Annals of Applied Probability*, pp. 49–77, 1995.

[8] S. Foss and N. Chernova, "On the stability of a partially accessible multi-station queue with state-dependent routing," *Queueing Systems*, vol. 29, no. 1, pp. 55–73, 1998.

[9] Y. M. Suhov and N. D. Vvedenskaya, "Fast jackson networks with dynamic routing," *Problems of Information Transmission*, vol. 38, no. 2, pp. 136–153, 2002.

[10] G. Foschini and J. Salz, "A basic dynamic routing problem and diffusion," *IEEE Transactions on Communications*, vol. 26, no. 3, pp. 320–327, 1978.

[11] A. Ephremides, P. Varaiya, and J. Walrand, "A simple dynamic routing problem," *IEEE transactions on Automatic Control*, vol. 25, no. 4, pp. 690–693, 1980.

[12] N. D. Vvedenskaya, R. L. Dobrushin, and F. I. Karpelevich, "Queueing system with selection of the shortest of two queues: An asymptotic approach," *Problemy Peredachi Informatsii*, vol. 32, no. 1, pp. 20–34, 1996.

[13] P. Eschenfeldt and D. Gamarnik, "Join the shortest queue with many servers. the heavy-traffic asymptotics," *Mathematics of Operations Research*, vol. 43, no. 3, pp. 867–886, 2018.

[14] J. Dai, J. J. Hasenbein, and B. Kim, "Stability of join-the-shortest-queue networks," *Queueing Systems*, vol. 57, no. 4, pp. 129–145, 2007.

[15] R. D. Foley and D. R. McDonald, "Join the shortest queue: stability and exact asymptotics," *The Annals of Applied Probability*, vol. 11, no. 3, pp. 569–607, 2001.

[16] X. Ling, M.-B. Hu, R. Jiang, and Q.-S. Wu, "Global dynamic routing for scale-free networks," *Physical Review E*, vol. 81, no. 1, p. 016113, 2010.

[17] P. Sarachik and U. Ozguner, "On decentralized dynamic routing for congested traffic networks," *IEEE Transactions on Automatic Control*, vol. 27, no. 6, pp. 1233–1238, 1982.

[18] J. G. Dai and S. P. Meyn, "Stability and convergence of moments for multiclass queueing networks via fluid limit models," *IEEE Transactions on Automatic Control*, vol. 40, no. 11, pp. 1889–1904, 1995.