Stable Scheduling Policies for Maximizing Throughput in Generalized Constrained Queueing Systems

Prasanna Chaporkar, *Member, IEEE*, and Saswati Sarkar, *Member, IEEE*

***Abstract*—We consider a class of queueing networks referred to as "generalized constrained queueing networks" which form the basis of several different communication networks and information systems. These networks consist of a collection of queues such that only certain sets of queues can be concurrently served. Whenever a queue is served, the system receives a certain reward. Different rewards are obtained for serving different queues, and furthermore, the reward obtained for serving a queue depends on the set of concurrently served queues. We demonstrate that the dependence of the rewards on the schedules alter fundamental relations between performance metrics like throughput and stability. Specifically, maximizing the throughput is no longer equivalent to maximizing the stability region; we therefore need to maximize one subject to certain constraints on the other. Since stability is critical for bounding packet delays and buffer overflow, we focus on maximizing the throughput subject to stabilizing the system. We design provably optimal scheduling strategies that attain this goal by scheduling the queues for service based on the queue lengths and the rewards provided by different selections. The proposed scheduling strategies are however computationally complex. We subsequently develop techniques to reduce the complexity and yet attain the same throughput and stability region. We demonstrate that our framework is general enough to accommodate random rewards and random scheduling constraints.**

***Index Terms*—Constrained queueing networks, multicast, optimization, randomized algorithms, stability, throughput, wireless.**

I. Introduction

CONSTRAINED queueing networks have been extensively used to model several systems of practical interest including wireless networks [35], [34], [25], [27], input queued switches [23], and database systems [34]. A constrained queueing network is a collection of queues such that only certain sets of queues can be concurrently served; these "schedulable sets" depend on the underlying system. Whenever a queue is served, the system receives a certain reward. In such systems, queues need to be selected for service such that 1) the total reward earned by the system per unit time ("throughput") is maximized, and 2) each queue is served often enough such that the mean queue length in each queue is bounded ("system stability"). The two goals turn out to be equivalent if the service of each queue (i.e., the transmission of each packet) fetches the same reward. The performances of such networks are now reasonably well understood owing to several seminal contributions [1], [3], [4], [6], [24]–[26], [35].

We now investigate constrained queueing networks where different rewards are obtained for transmitting packets from different queues, and furthermore, the reward obtained for serving a queue depends on the set of concurrently served queues. Such *generalized constrained queueing networks* form the basis of several communication and information systems of practical interest, but have not received adequate attention in the research community. We first provide examples of such systems, and subsequently demonstrate that new resource allocation goals and techniques are required for capturing the tradeoff between different performance metrics in these systems.

First, consider one-to-many communications in wireless networks. Here, a sender may wish to transmit its packets to multiple receivers in its communication range. Due to the broadcast property of the wireless transmission, a single transmission may reach all these receivers. Here, each sender constitutes a queue, and the reward attained by a transmission is the number of receivers who successfully receive it. Since different multicast groups have different number of receivers, the reward attained by serving different queues will be different. Furthermore, whether a receiver can successfully decode a transmission depends on other transmissions in its neighborhood. Thus, the reward associated with each transmission depends on the set of queues served concurrently. For example in Fig. 1 when S_2 is transmitting to R_6, R_1, and R_2 cannot receive a transmission from S_1 as both the transmissions will collide at these receivers. Hence, S_1 receives a reward of 5 when S_1 alone is served, and it receives a reward of 3 when S_1 and S_2 are served together. Thus, the reward for S_1 depends on the set of queues served.

Now, consider one-to-one communication in wireless networks. Success of each transmission depends upon the interference due to concurrent transmissions in the network and the channel state. Let the reward for each transmission be 1 if the transmission is successful. Thus, different transmissions attain different rewards depending on the set of queues served. Furthermore, here, the same selection of sessions may generate different rewards at different times as the interferences randomly change due to fading—rewards may therefore be random.

Manuscript received February 16, 2006; revised November 12, 2007. First published September 12, 2008; current version published September 24, 2008. This work was supported by the National Science Foundation Under Grants ANI-0106984, NCR-0238340, and CNS-0435306. Recommended by Associate Editor A. Lim.

P. Chaporkar is with Indian Institute of Technology, Mumbai 400076, India (e-mail: chaporkar@ee.iitb.ac.in).

S. Sarkar is with the Department of Electrical and Systems Engineering at University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: swati@seas.upenn.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2008.929372

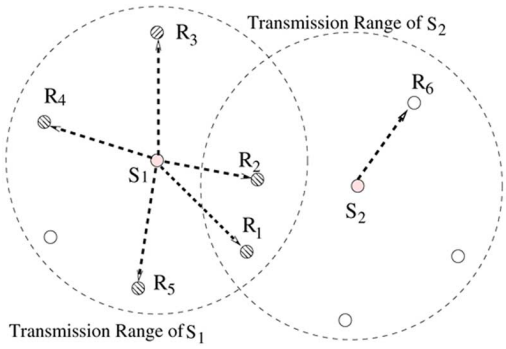

Fig. 1. Figure shows an example to demonstrate the application of generalized constrained queueing networks in one-to-many communication in wireless networks. There are two senders S_1, S_2, and six receivers R_1 to R_6. The dashed circles indicate the communication ranges of the senders. A single transmission from S_1 can reach all its receivers, $R_1, \ldots, R_5$. Here, R_6 is S_2's receiver. Each sender corresponds to a queue. Here, $\mathcal{L} = \{\vec{\ell}_1 = [0\ 0], \vec{\ell}_2 = [1\ 0]\vec{\ell}_3 = [0\ 1], \vec{\ell}_4 = [1\ 1]\}$. Here, $r_k(\vec{\ell}_1) = 0$, $k \in \{1,2\}$, $r_1(\vec{\ell}_2) = 5$, $r_2(\vec{\ell}_2) = 0$, $r_1(\vec{\ell}_3) = 0$, $r_2(\vec{\ell}_3) = 1$, $r_1(\vec{\ell}_4) = 3$, $r_2(\vec{\ell}_4) = 1$.

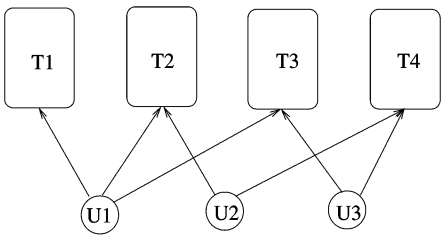

Fig. 2. Database system with four tables $T1, \ldots, T4$ that are accessed by three applications $U1, U2$ and $U3$. The arrows indicate the tables each application updates. When there are concurrent requests for updates in the same table, the request from an application with the lowest id is honored. Note that if all three applications try to simultaneously update the database, then $U1$, $U2$, and $U3$ achieve rewards 3, 1, and 0, respectively. If only $U2$ and $U3$ try to simultaneously update the database, then they achieve rewards 2 and 1, respectively.

Next, in many database systems, a single update operation from an application involves updates in many tables. Here, each application constitutes a queue, and the reward attained by an update operation is the number of tables that are successfully updated. Since different applications require to update different number of tables, rewards received by serving different queues will be different. Moreover, if many applications try to update the same table, then only one of them can do so, as the access to these tables is controlled to avoid inconsistencies due to concurrent updates. Thus, the reward for a queue depends on the set of queues served. We demonstrate this using a specific application in Fig. 2.

Our contribution is to provide a mathematical framework for modeling and optimizing key performance attributes in generalized constrained queueing networks. First, we define appropriate performance metrics (Section II). Next, we demonstrate that the fundamental relations between performance metrics such as throughput and stability change due to the dependence of the rewards on the set of queues served (Section III). Specifically, maximizing the throughput is no longer equivalent to maximizing the stability region; we therefore need to maximize one subject to certain constraints on the other. Since stability is critical for bounding packet delays and buffer overflow, we focus on maximizing the throughput subject to stabilizing the system. We design provably optimal scheduling strategies that attain this goal by scheduling the queues for service based on the queue lengths and the rewards provided by different selections (Section IV). These scheduling strategies are however computationally complex. We next develop a framework to reduce the computational complexity and yet attain the optimum performance (Section V). Finally, we consider some possible generalizations (Section VI) and describe the related work (Section VII).

II. System Model

We consider a queueing network with n queues. We assume that time is slotted. In each queue $k \in \{1, \ldots, n\}$ packets arrive as per arrival process $\{\Lambda_k(t)\}_{t=1}^{\infty}$, where $\Lambda_k(t)$ is the number of arrivals in queue k during slot t. Arrivals for the same session in different slots are independent and identically distributed. The arrival processes for different sessions are independent but not identically distributed. We assume that $\Lambda_k(t) \leq A_{\max}$ in each slot t and for any k. Let $\lambda_k \stackrel{\text{def}}{=} \mathbb{E}[\Lambda_k(t)]$ and $\vec{\lambda} = (\lambda_1, \ldots, \lambda_n)$ denote the arrival rate vector. Each packet can be served in at most one slot, and it departs the system at the end of the slot in which it is served. This assumption has been motivated by the fact that in wireless networks multiple transmissions of the same packet consume additional energy and increase the interference for other transmissions. We denote by $Q_k(t)$ the queue length of the kth queue at the beginning of slot t. Also, $\vec{Q}(t) = [Q_1(t)\ \cdots\ Q_n(t)]$.

A queue can only be served if it has a packet to transmit, and in each slot in which it is served it transmits one packet. The indicator $\ell_k(t) = 1$ if the kth queue is served in slot t, and is 0 otherwise. The vector $\vec{\ell}(t) = [\ell_1(t)\ \cdots\ \ell_n(t)]$ denotes the service vector in slot t. The system constraints may prohibit simultaneous service of certain queues. Thus, all 2^n n-dimensional binary vectors may not constitute a valid service vector. Let $\mathcal{L} = \{\vec{\ell}_1, \ldots, \vec{\ell}_m\}$ denote the set of all valid service vectors, and ℓ_{ik} denote the kth element of $\vec{\ell}_i \in \mathcal{L}$. Clearly $m \leq 2^n$. For example, Fig. 1 elucidates a constrained queueing network with $n = 2$ and $m = 4$. Now, if the system has an additional constraint that all the receivers should receive every packet, then both S_1 and S_2 cannot be served concurrently. Thus, in this case, $\mathcal{L} = \{\vec{\ell}_1 = [0\ 0], \vec{\ell}_2 = [1\ 0], \vec{\ell}_3 = [0\ 1]\}$ and $m = 3$.

We assume the following about $\mathcal{L}$. If $\vec{\ell} \in \mathcal{L}$, then every $\vec{\ell}_1 \leq \vec{\ell}$ also belongs to $\mathcal{L}$, where the inequality is element-wise. In other words, if a certain set of queues can be served simultaneously, then any subset of these queues can also be served simultaneously. Note that this assumption holds in wireless networks. For each $\vec{\ell}_i \in \mathcal{L}$ and queue length vector $\vec{Q}$, we define an n-dimensional vector $\vec{\ell}_i(\vec{Q})$ as follows. The kth component of $\vec{\ell}_i(\vec{Q})$ equals ℓ_{ik} if $Q_k > 0$, and is 0 otherwise. Clearly, for each $\vec{\ell}_i \in \mathcal{L}$ and $\vec{Q}$, $\vec{\ell}_i(\vec{Q}) \in \mathcal{L}$.

The system receives a reward for serving each queue, and the reward obtained for serving the kth queue in slot t, $r_k(\vec{\ell}(t))$ is a function of the service vector $\vec{\ell}(t)$ in slot t, for each k. We assume that $r_k(\vec{\ell}(t)) \leq G_k < \infty$ for each k. We initially assume that the reward for each queue is a deterministic function of the service vector, and later generalize to allow the reward to

randomly depend on the service vector (Section VI). Refer to Fig. 1 for some example rewards.

We assume the following properties of the reward function. First, if $\ell_k = 0$ then $r_k(\vec{\ell}) = 0$. Thus, if a queue is not served then it does not receive any reward. Next, for any $\vec{\ell}_1, \vec{\ell}_2 \in \mathcal{L}$ if $\vec{\ell}_1 \leq \vec{\ell}_2$ and $\ell_{1k} > 0$, $r_k(\vec{\ell}_1) \geq r_k(\vec{\ell}_2)$. Thus, $r_k(\vec{\ell}(\vec{Q})) \geq r_k(\vec{\ell})$ for any $\vec{Q}$, $\vec{\ell} \in \mathcal{L}$ and k such that $Q_k > 0$. We justify this assumption in context of one of the application scenarios, wireless networks. In wireless networks, when fewer queues transmit, the interference is less in the system and therefore, usually, the queues that transmit receive higher reward. If this is not the case, e.g., when the probability of success increases with increase in interference due to the use of sophisticated decoding strategies, then if an empty queue is selected, it can transmit a signal[1] so as to ensure that other queues do not receive less reward because it is empty. This may increase the overall energy consumption, but our focus here is to maximize the throughput. Joint minimization of the energy consumption and maximization of the throughput consists of interesting topics for future research. The assumption can also be similarly justified for database systems.

Next, we present some important definitions.

Definition 1 (Scheduling Policy): A scheduling policy Δ decides the service vector $\vec{\ell}^{\Delta}(t)$ in each slot $t \geq 1$ such that $\vec{\ell}^{\Delta}(t) \in \mathcal{L}$ and $\ell_i^{\Delta}(t) = 0$ if $Q_i(t) = 0$.

This class includes *offline* policies that decide their service vectors based on the knowledge of packet arrivals in each past, present and even future slots.

Since $r_k(\vec{\ell}^{\Delta}(u)) = 0$ if $\ell_k^{\Delta}(u) = 0$, and $\ell_k^{\Delta}(u) = 0$ if $Q_k(u) = 0$, $r_k(\vec{\ell}^{\Delta}(u)) = 0$ if $Q_k(u) = 0$. Thus, irrespective of the scheduling policy, no queue receives any reward in a slot in which it is empty.

Definition 2 (Throughput): For an arrival rate vector $\vec{\lambda}$, the throughput under a scheduling policy Δ, $\Omega^{\Delta}(\vec{\lambda})$, is the reward it receives per unit time. Mathematically

$$\Omega^{\Delta}(\vec{\lambda}) = \liminf_{t\to\infty} \frac{1}{t} \sum_{u=1}^{t} \sum_{k=1}^{n} r_k(\vec{\ell}^{\Delta}(u)).$$

Since $r_k(\vec{\ell}^{\Delta}(u)) = 0$ if $\ell_k^{\Delta}(u) = 0$, and $\ell_k^{\Delta}(u) \in \{0, 1\}$,

$$\Omega^{\Delta}(\vec{\lambda}) = \liminf_{t\to\infty} \frac{1}{t} \sum_{u=1}^{t} \sum_{k=1}^{n} r_k(\vec{\ell}^{\Delta}(u)) \ell_k^{\Delta}(u). \tag{1}$$

Note that if the reward $r_k(\vec{\ell})$ is the number of receivers of session k that receive a packet when the service vector is $\vec{\ell}$, the throughput under Δ is the sum, over all receivers, of the number of packets each receiver receives per unit time. This is consistent with the usual definition of throughput in a communication network.

Definition 3 (Loss): The loss under a scheduling policy Δ at any slot t is the difference between the sum of the maximum possible rewards of the queues it serves at t and the reward it obtains at t. The loss under a scheduling policy Δ, $L^{\Delta}(\vec{\lambda})$, is its total loss per unit time. Mathematically

$$L^{\Delta}(\vec{\lambda}) = \liminf_{t\to\infty} \frac{1}{t} \sum_{u=1}^{t} \sum_{k=1}^{n} (G_k - r_k(\vec{\ell}^{\Delta}(u))) \ell_k^{\Delta}(u).$$

In a communication network, usually, the loss experienced by a receiver denotes the number of packets transmitted by its source that it does not receive per unit time, and the network loss denotes the sum of the losses of all receivers. Again, if the reward $r_k(\vec{\ell})$ is the number of receivers of session k that receive a packet when the service vector is $\vec{\ell}$, then the formal definition of loss in Definition 3 has the same connotation as above.

Definition 4 (System Stability): The queueing system is said to be stable if the time average of queue lengths is finite for each queue, i.e., $\limsup_{t\to\infty}(\sum_{u=1}^{t} Q_i(u))/(t) < \infty$ with probability (w.p.) 1 for each i. A scheduling policy that stabilizes the system is called a stable scheduling policy. The stability region of a scheduling policy is the set of arrival rate vectors for which the system is stable under the policy. The stability region of the system Θ is the union of the stability regions of all scheduling policies. A scheduling policy whose stability region equals Θ is said to maximize the stability region.

Let $\overline{\mathcal{C}}$ denote the convex hull of the vectors in $\mathcal{L}$ and $\mathcal{C}$ denote the interior of $\overline{\mathcal{C}}$. In their seminal work, Tassiulas *et al.* [35, Theorem 3.2] showed that $\mathcal{C} \subseteq \Theta \subseteq \overline{\mathcal{C}}$.

Definition 5 (Stabilizable Arrival Rate Vector): We denote the arrival rate vector $\vec{\lambda}$ as stabilizable if $\vec{\lambda} \in \mathcal{C}$.

Definition 6 (Throughput Optimality): A stable scheduling policy Δ is said to be throughput optimal if w.p. 1 it attains the maximum throughput among all the stable scheduling policies. We denote the throughput attained by such a policy for arrival rate vector vector $\vec{\lambda} \in \Theta$ by $\Omega_{\max}(\vec{\lambda})$.

Definition 7 (ϵ-Throughput Optimality): A scheduling policy Δ is said to be ϵ-throughput optimal for a $\epsilon > 0$ if 1) it is stable, and 2) $\Omega^{\Delta}(\vec{\lambda}) \geq \Omega_{\max}(\vec{\lambda}) - \epsilon$ w.p. 1.

In the next section, we show that in generalized constrained queueing networks maximizing the stability region is not equivalent to maximizing the throughput. Since stability is imperative for guaranteeing bounded delay and for limiting packet drop due to buffer overflow, *we aim to maximize the throughput subject to stabilizing the system. Specifically, our goal is to design ϵ-throughput optimal policies.* We now investigate the relation between the throughput and the loss

$$\text{Now, } L^{\Delta}(\vec{\lambda}) = \sum_{k=1}^{n} G_k \liminf_{t\to\infty} \frac{1}{t} \sum_{u=1}^{t} \ell_k^{\Delta}(u) - \Omega^{\Delta}(\vec{\lambda}).$$

Note that if a system is stable under policy Δ, $\lim_{t\to\infty}(1)/(t) \sum_{u=1}^{t} \ell_k^{\Delta}(u) = \lambda_k$ w.p. 1. Thus, $L^{\Delta}(\vec{\lambda}) = \sum_{k=1}^{n} \lambda_k G_k - \Omega^{\Delta}(\vec{\lambda})$ w.p. 1. Thus, if $\vec{\lambda}$ is in the stability region of policies Δ_1, Δ_2, $\Omega^{\Delta_1}(\vec{\lambda}) + L^{\Delta_1}(\vec{\lambda}) = \Omega^{\Delta_2}(\vec{\lambda}) + L^{\Delta_2}(\vec{\lambda})$ w.p. 1. Thus, for any stabilizable arrival rate vector $\vec{\lambda}$, a throughput optimal policy must also minimize the loss, and an ϵ-throughput optimal policy attains a loss which is at most ϵ more than the loss of

[1]Transmission of a signal from an empty queue is not considered service for the empty queue.

any stable policy. Thus, we focus on obtaining ϵ-throughput optimal policies.

III. Relation Between Throughput and Stability

First, we examine what decisions policies are likely to make if they want to maximize only the stability region, or if they want to maximize only the throughput. A policy that aims to maximize the stability region serves as many packets as possible in a slot while giving priority to longer queues. If the policy aims to maximize the throughput, then it may wait and transmit only when the reward is high so that each packet fetches the maximum possible reward. Thus, the control decisions for maximizing the stability region and for maximizing the throughput are not equivalent.

Using an example that is motivated by one-to-many communication in wireless networks (Fig. 1), we next demonstrate that a policy that maximizes the stability region does not maximize the throughput.

Example 1: Consider the system shown in Fig. 1. Let $\vec{\lambda} = (1/2-\epsilon, 1/2-\epsilon)$, where ϵ is a small positive real number. Now, consider a policy $\hat{\Delta}$ that serves each queue whenever it is nonempty. Thus, if only S_1 (S_2, resp.) is nonempty, then $\hat{\Delta}$ will select service vector $\vec{\ell}_2$ ($\vec{\ell}_3$, resp.) and achieve a reward of 5 (1, resp.). If both queues are non-empty in a slot, then $\hat{\Delta}$ will select $\vec{\ell}_4$ and achieve a reward of 4. Clearly, $\hat{\Delta}$ maximizes the stability region. Now, the service process for S_1 is independent of that for S_2. Using Little's law, the fraction of slots in which S_1 (S_2, resp.) is non-empty and S_2 (S_1. resp.) is empty is $1/4-\epsilon^2$ ($1/4-\epsilon^2$, resp.), and the fraction of slots in which both queues are non-empty is $(1/2-\epsilon)^2$. Thus, $\Omega^{\hat{\Delta}}(\vec{\lambda}) = (5+1)(1/4-\epsilon^2)+4(1/2-\epsilon)^2 \approx 10/4$. Now, consider a policy Δ' that serves only S_1 when S_1 is non-empty, and serves only S_2 if S_1 is empty and S_2 is non-empty. Note that Δ' is stable as $\lambda_1+\lambda_2 = 1$. Thus, whenever S_1 (S_2, resp.) is served, the service vector is $\vec{\ell}_2$ ($\vec{\ell}_3$, resp.) and the reward is 5 (1, resp.). Since the queues are stable, S_1 and S_2 are served in $1/2-\epsilon$ fraction of slots each. Thus, $\Omega^{\Delta'} = (5+1)(1/2-\epsilon) \approx 3$. Thus, $\Omega^{\hat{\Delta}}(\vec{\lambda}) < \Omega^{\Delta'}(\vec{\lambda})$.

Note that $\hat{\Delta}$ in Example 1 always transmits the maximum number of packets in each slot and also chooses the set of queues whose sum of queue lengths is the maximum. Tassiulas *et al.* [35] showed that a policy that satisfies the latter property maximizes the stability region in arbitrary constrained queueing networks, but, Example 1 shows that $\hat{\Delta}$ does not maximize the throughput. This is because $\hat{\Delta}$ does not consider the reward structure in deciding the service vector. So, the policies designed to maximize the stability region of the constrained queueing system (e.g., see [1], [5], [19], [34], [35]) need not maximize the throughput. Thus, we need alternate mechanism to design throughput optimal policies.

Now, we consider two policies, Δ_1, Δ_2, that seek to maximize the reward in a greedy fashion. Δ_1 serves each queue only when the queue can obtain its maximum possible reward, and Δ_2 selects in each slot the service vector that attains the maximum possible reward among all valid service vectors in the slot. Simply put, Δ_1 maximizes the reward per packet, and Δ_2 greedily maximizes the reward in each slot. We show that Δ_1 does not stabilize the system even when the arrival rate vector is stabilizable, and Δ_2 does not attain the maximum throughput among all stable policies.

Example 2: Consider the system shown in Fig. 1. Let $\vec{\lambda} = (3/4, 1/2)$. Clearly, $\vec{\lambda} \in \mathcal{C}$ and policy $\hat{\Delta}$ in Example 1 stabilizes the system. Note that Δ_1 will never concurrently serve both queues. Hence, the sum of the service rates provided to the two queues is at most 1. Thus, since $\lambda_1+\lambda_2 > 1$, Δ_1 does not stabilize the system.

Note that Δ_1 maximizes the reward per packet while serving queues at rates smaller than their arrival rates and thereby compromises stability.

Example 3: Consider the system shown in Fig. 1 with the difference that $r_2(\vec{\ell}_3) = r_2(\vec{\ell}_4) = 3$. Let $\vec{\lambda} = (1/4, 1/4)$. Note that for the above rewards, Δ_2 selects the same service vectors as $\hat{\Delta}$. Thus, Δ_2 stabilizes the system. Now, the service process for S_1 is independent of that for S_2. Using Little's law, the fraction of slots in which S_1 (S_2, resp.) is non-empty and S_2 (S_1. resp.) is empty is $(1/4)(3/4)$ ($(1/4)(3/4)$), resp.), and the fraction of slots in which both queues are non-empty is $(1/4)^2$. Thus, $\Omega^{\hat{\Delta}}(\vec{\lambda}) = (5+3)(3/16)+6(1/16) = 15/8$. Now, consider Δ' described in Example 1. Again, Δ' is stable as $\lambda_1+\lambda_2 < 1$. Thus, whenever S_1 (S_2, resp.) is served, the service vector is $\vec{\ell}_2$ ($\vec{\ell}_3$, resp.) and the reward is 5 (3, resp.). Since the queues are stable, S_1 and S_2 are served in $1/4$ fraction of slots each. Thus, $\Omega^{\Delta'} = (5+3)(1/4) = 2$. Thus, Δ_2 does not attain the maximum throughput among all stable policies.

The limitation of Δ_2 is that it myopically bases its decision in a slot solely on the aggregate reward in the slot. Thus, even when it is possible to wait and serve queues in mutually disjoint slots and achieve a higher reward per packet, Δ_2 serves the queues in the same slot.

The examples demonstrate that 1) a policy that maximizes the stability region need not maximize the throughput, 2) myopically maximizing the reward in each slot or the reward per packet may not maximize the throughput or stabilize the system, and 3) the optimal policy should wait just long enough so as to achieve the highest possible reward per packet while serving each queue at a rate higher than its arrival rate.

IV. Optimal Policies

In this section, we propose two policies and prove that they are ϵ-throughput optimal for every stabilizable arrival rate vector $\vec{\lambda}$ and $\epsilon > 0$.

A. Linear Program-Based Optimal Policy (Δ^*)

The scheduling policy Δ^* selects $\vec{\ell}_i \in \mathcal{L}$ w.p. w_i in every slot. If $\vec{\ell}_i$ is chosen in slot t, then $\vec{\ell}^{\Delta^*}(t) = \vec{\ell}_i(\vec{Q}(t))$, i.e., the kth queue transmits a packet if $\ell_{ik} = 1$ and $Q_k(t) > 0$. Recall that $\vec{\ell}^{\Delta^*}(t)$ is the indicator vector for the set of queues served by Δ^* in slot t.

Let Δ^* select $\vec{\ell}_i$ in a slot t. Then $\vec{\ell}^{\Delta^*}(t) \leq \vec{\ell}_i$. The inequality is strict only when some queues in $\vec{\ell}_i$ are empty in t, and then, as discussed in Section II, $r_k(\vec{\ell}^{\Delta^*}(t)) \geq r_k(\vec{\ell}_i)$ for each k for which $Q_k(t) > 0$. The probability distribution $\vec{w} = [w_1 \cdots w_m]$ is computed using the following linear program $\mathrm{LP}(\vec{\lambda}, \delta)$. Here, δ is a parameter.

LP$(\vec{\lambda}, \delta)$:- **Maximize**: $U(\vec{\lambda}, \delta) = \sum_i \sum_k w_i \ell_{ik} r_k(\vec{\ell}_i)$

Subject to:

1) $\sum_{i=1}^{m} w_i = 1$ and $w_i \geq 0$ for every i.
2) $\sum_{i=1}^{m} w_i \ell_{ik} = \lambda_k + \delta$ for every k.

Constraint 1) ensures that $\vec{w}$ is a valid probability distribution. When $\delta > 0$, constraint 2) ensures that each queue is selected for service at a rate higher than the arrival rate in the queue. Thus, constraint 2) ensures stability.

Note that $\vec{w}$ and hence Δ^* depend on $\vec{\lambda}$ and the chosen δ. We indicate this dependence by using the notations $\vec{w}(\vec{\lambda}, \delta)$ and $\Delta^*(\vec{\lambda}, \delta)$.

Now, although $\mathbf{LP}(\vec{\lambda}, \delta)$ is well-defined, it need not have any feasible solution, for arbitrary $\vec{\lambda} \in \mathcal{R}^n$ and $\delta \in \mathcal{R}$. Theorem 1 shows that for all stabilizable $\vec{\lambda}$ and sufficiently small positive δ, $\mathbf{LP}(\vec{\lambda}, \delta)$ is feasible and $\Delta^*(\vec{\lambda}, \delta)$ is ϵ-throughput optimal. Note that allowing arbitrary $\vec{\lambda} \in \mathcal{R}^n$ and $\delta \in \mathcal{R}$ in $\mathbf{LP}(\vec{\lambda}, \delta)$ simplifies the proof for Theorem 1.

Theorem 1: Let $\vec{\lambda}$ be any stabilizable arrival rate vector. Then, for every $\epsilon > 0$ there exists a $\hat{\delta}$ such that for every $\delta \in (0, \hat{\delta})$, $\mathbf{LP}(\vec{\lambda}, \delta)$ is feasible and $\Delta^*(\vec{\lambda}, \delta)$ is ϵ-throughput optimal. Furthermore

$$\Omega^{\Delta^*(\vec{\lambda},\delta)} \geq U(\vec{\lambda}, \delta) - \delta \sum_{k=1}^{n} G_k \geq \Omega_{\max}(\vec{\lambda}) - \epsilon \quad \text{w.p.1.} \tag{2}$$

We prove Theorem 1 in the Appendix.

Finally, the stability region can be maximized using arbitrary feasible solutions of $\mathbf{LP}(\vec{\lambda}, \delta)$ [5]. Specifically, if $\Delta^*(\vec{\lambda}, \delta)$ selects the service vectors as per any probability distribution that constitutes a feasible solution of $\mathbf{LP}(\vec{\lambda}, \delta)$ for any positive δ, it stabilizes the system provided $\vec{\lambda}$ is stabilizable [5]. However, for attaining the maximum throughput among all stable policies, an optimal solution of $\mathbf{LP}(\vec{\lambda}, \delta)$ must be used. Specifically, for any stabilizable $\vec{\lambda}$ and $\epsilon > 0$, $\Delta^*(\vec{\lambda}, \delta)$ is ϵ-optimal for any $\delta \in (0, \min\{\delta_{\max}(\vec{\lambda}), \epsilon / \sum_{k=1}^{n} G_k\}]$, where $\delta_{\max}(\vec{\lambda})$ is the maximum value of δ for which $\mathbf{LP}(\vec{\lambda}, \delta)$ has a feasible solution (follows from Theorem 1 and Lemma 5 in the Appendix).

B. Queue Length-Based Optimal Policy (Δ_o)

The policy $\Delta^*(\vec{\lambda}, \delta)$ requires the knowledge of $\vec{\lambda}$ in order to obtain the optimal $\vec{w}(\vec{\lambda}, \delta)$. The system may not however know $\vec{\lambda}$. We now design a policy Δ_O that attains the maximum throughput among all stable policies and stabilizes the system for any stabilizable $\vec{\lambda}$ without knowing $\vec{\lambda}$.

Recall that an optimal policy should wait as long as possible to achieve the highest possible reward per packet without violating system stability (Section III). Now, $\Delta^*(\vec{\lambda}, \delta)$ uses the knowledge of $\vec{\lambda}$ to ensure the above, whereas Δ_O ensures the above by using only the value of $\vec{Q}(t)$.

We now describe Δ_O. In slot t, Δ_O selects the service vector $\vec{\ell}^{\Delta_O}(t)$ such that

$$\vec{\ell}^{\Delta_O}(t) = \arg\max_{\substack{\vec{\ell} \in \mathcal{L}, \\ \vec{\ell} = \vec{\ell}(\vec{Q}(t))}} \left\{ \sum_{k=1}^{n} (Q_k(t) - V \times (G_k - r_k(\vec{\ell})))\ell_k \right\} \tag{3}$$

where V is a constant. Note that the constraint $\vec{\ell} = \vec{\ell}(\vec{Q}(t))$ implies that $\ell_k = 0$ if $Q_k(t) = 0$ and $\ell_k \in \{0, 1\}$ otherwise.

Theorem 2: Let $\vec{\lambda}$ be any stabilizable arrival rate vector. Then, for every $V \geq 0$, Δ_O stabilizes the system. Moreover, for every $\epsilon > 0$, there exists $\hat{V}$ such that for every $V \geq \hat{V}$, Δ_O is ϵ-throughput optimal.

The above result implies that any stable offline policy that takes transmission decisions based on the knowledge of past, present, and future arrivals cannot attain throughput significantly more than $\Omega^{\Delta_O}(\vec{\lambda})$ for every stabilizable $\vec{\lambda}$. This holds even though Δ_O takes transmission decisions based only on the current queue lengths.

Now, we describe the intuition behind this result. Let

$$W(\vec{Q}, \vec{\ell}) \stackrel{\text{def}}{=} \sum_{k=1}^{n} (Q_k \ell_k - V \times (G_k - r_k(\vec{\ell}))\ell_k).$$
$$W_1(\vec{Q}, \vec{\ell}) \stackrel{\text{def}}{=} \sum_{k=1}^{n} Q_k \ell_k. \tag{4}$$

Note that intuitively $G_k - r_k(\vec{\ell})$ is the loss of reward of the kth queue when service vector $\vec{\ell}$ is used. Thus, in each slot t, Δ_O selects the service vector $\vec{\ell}$ that maximizes the dot product, $W(\vec{Q}(t), \vec{\ell})$, of $\vec{\ell}$ and the difference between the queue length vector $\vec{Q}(t)$ and a scaled loss vector associated with $\vec{\ell}$. Note that a policy $(\hat{\Delta})$ that selects the service vector $\vec{\ell}$ that maximizes the dot product, $W_1(\vec{Q}(t), \vec{\ell})$, of $\vec{\ell}$ and the queue length vector $\vec{Q}(t)$ stabilizes the system for every stabilizable $\vec{\lambda}$ [1], [18], [35]. This is because under $\hat{\Delta}$ the queue length process has a negative drift when $\sum_{k=1}^{n} Q_k(t)$ is sufficiently large for every stabilizable $\vec{\lambda}$. When $\sum_{k=1}^{n} Q_k(t) \gg V \sum_{k=1}^{n} G_k$, $W(\vec{Q}(t), \vec{\ell}) \approx W_1(\vec{Q}(t), \vec{\ell})$ for every $\vec{\ell} \in \mathcal{L}$, and therefore, Δ_O and $\hat{\Delta}$ select similar service vectors. Thus, intuitively, for every stabilizable $\vec{\lambda}$, the queue length process under Δ_O should also have a negative drift when $\sum_{k=1}^{n} Q_k(t)$ is sufficiently large. Hence, Δ_O also stabilizes the system for any stabilizable $\vec{\lambda}$.

We have however shown that all stable policies do not attain equal throughput (Example 1). So, it is not obvious that Δ_O maximizes the throughput among all policies that stabilize the system; we now provide the intuition behind why this is the case. Note that when the queue lengths are small, high throughput can be attained without violating stability by serving the queues only when they receive high rewards. On the other hand, stability can be ensured by selecting the queues with higher queue lengths and by serving a large number of packets when the queue lengths are large. We now demonstrate that Δ_O follows both the above principles. For simplicity, assume that $V, G_k, r_k(\vec{\ell})$ are integers for all $k, \vec{\ell} \in \mathcal{L}$. Now, when $Q_k(t) < V$, $Q_k(t) - V(G_k - r_k(\vec{\ell})) \geq 0$ only if $r_k(\vec{\ell}) = G_k$. Then, since Δ_O maximizes $W(\vec{Q}(t), \vec{\ell})$, it will serve the kth queue only if the maximum possible reward is achievable. Now, if $Q_k \in \{V, \ldots, 2V - 1\}$, then $Q_k(t) - V(G_k - r_k(\vec{\ell})) \geq 0$ only if $r_k(\vec{\ell}) \geq G_k - 1$. Thus, Δ_O will serve the kth queue only if the achievable reward is greater than or equal to $G_k - 1$. Similarly, if $Q_k(t) \in \{(G_k - u)V, \ldots, (G_k - u + 1)V - 1\}$, then Δ_O will serve the kth queue only if the achievable reward is greater than or equal to u. Summarily, Δ_O attains the maximum possible reward for every packet while maintaining stability by dynamically selecting the service vectors based on the queue lengths. Thus, Δ_O attains the maximum throughput among all stable policies.

Now, we prove Theorem 2 using a combination of optimization and Lyapunov theories. Neely *et al.* [27] proposed this proof technique in a different context.

Proof: Consider a stabilizable $\vec{\lambda}$. For any policy Δ

$$Q_k(t+1) = Q_k(t) + \Lambda_k(t) - \ell_k^{\Delta}(t). \tag{5}$$

Now for Δ_O, $\{\vec{Q}(t)\}_{t\geq 1}$ is an irreducible, aperiodic, and countable Markov chain. Now, consider the Lyapunov function

$$f(\vec{Q}(t)) = \sum_{k=1}^{n} (Q_k(t))^2. \tag{6}$$

Let, $M \stackrel{\text{def}}{=} n(A_{\max}^2 + 1)$. From (5) and (6), it follows that

$$f(\vec{Q}(t+1)) - f(\vec{Q}(t)) \leq M + \sum_{k=1}^{n} \left[2Q_k(t)\Lambda_k(t) - 2Q_k(t)\ell_k^{\Delta_O}(t)\right].$$

Thus,

$$\begin{aligned}
&\mathbb{E}[f(\vec{Q}(t+1)) - f(\vec{Q}(t)) \,|\, \vec{Q}(t)] \\
&\quad \leq M + \sum_{k=1}^{n} 2Q_k(t)\lambda_k \\
&\qquad - \mathbb{E}\left[\sum_{k=1}^{n} 2Q_k(t)\ell_k^{\Delta_O}(t) \,|\, \vec{Q}(t)\right] \qquad (7) \\
&\quad = M + \sum_{k=1}^{n} 2Q_k(t)\lambda_k \\
&\qquad - 2\mathbb{E}[W(\vec{Q}(t), \vec{\ell}^{\Delta_O}(t)) \,|\, \vec{Q}(t)] \\
&\qquad - 2\mathbb{E}\left[\sum_{k=1}^{n} \big(Q_k(t)\ell_k^{\Delta}(t) \right. \\
&\qquad \left. -V[G_k - r_k(\vec{\ell}^{\Delta}(t))]\ell_k^{\Delta}(t)\big) \,|\, \vec{Q}(t)\right] \\
&\qquad - 2V\mathbb{E}\left[\sum_{k=1}^{n} [G_k - r_k(\vec{\ell}^{\Delta_O}(t))]\ell_k^{\Delta_O}(t) \,\middle|\, \vec{Q}(t)\right]. \qquad (8)
\end{aligned}$$

Now, since $\vec{\lambda}$ is stabilizable, we can obtain small enough positive δ such that $\Delta^*(\vec{\lambda}, \delta)$ is $\epsilon/2$-throughput optimal (Theorem 1). We consider $\Delta^*(\vec{\lambda}, \delta)$ for such a δ. Here, $\vec{\ell}^{\Delta^*(\vec{\lambda},\delta)}(t)$ is the service vector $\Delta^*(\vec{\lambda}, \delta)$ would have used at t if it had a queue length vector of $\vec{Q}(t)$ at t.

From definition of Δ_O (3), for every Δ and t

$$\mathbb{E}[W(\vec{Q}(t), \vec{\ell}^{\Delta_O}(t)) \,|\, \vec{Q}(t)] \geq \mathbb{E}[W(\vec{Q}(t), \vec{\ell}^{\Delta}(t)) \,|\, \vec{Q}(t)].$$

Thus, from (8), for every δ

$$\begin{aligned}
&\mathbb{E}[f(\vec{Q}(t+1)) - f(\vec{Q}(t)) \,|\, \vec{Q}(t)] \\
&\quad \leq M + \sum_{k=1}^{n} 2Q_k(t)\lambda_k \\
&\qquad - 2\mathbb{E}\left[W(\vec{Q}(t), \vec{\ell}^{\Delta^*(\vec{\lambda},\delta)}(t)) \,|\, \vec{Q}(t)\right] \\
&\qquad - 2V\mathbb{E}\left[\sum_{k=1}^{n} [G_k - r_k(\vec{\ell}^{\Delta_O}(t))]\ell_k^{\Delta_O}(t) \,|\, \vec{Q}(t)\right]. \qquad (9)
\end{aligned}$$

Now $\Delta^*(\vec{\lambda}, \delta)$ chooses each service vector $\vec{\ell}_i$ w.p. $w_i(\vec{\lambda}, \delta)$ independent of the queue lengths, and subsequently serves only those queues that are included in the selected service vector and are also non-empty. Thus, if $Q_k(t) > 0$

$$\mathbb{E}\left[\ell_k^{\Delta^*(\vec{\lambda},\delta)}(t) \,|\, \vec{Q}(t)\right] = \sum_{i=1}^{m} \ell_{ik} w_i(\vec{\lambda}, \delta) = \lambda_k + \delta. \tag{10}$$

In addition, recall that if Δ^* selects $\vec{\ell}_i$, and if the kth queue is nonempty it receives a reward of at least $r_k(\vec{\ell}_i)$. Thus, if $Q_k(t) > 0$

$$\begin{aligned}
&\mathbb{E}\left[\left(G_k - r_k\left(\vec{\ell}^{\Delta^*(\vec{\lambda},\delta)}(t)\right)\right)\ell_k^{\Delta^*(\vec{\lambda},\delta)}(t) \,\middle|\, \vec{Q}(t)\right] \\
&\quad \leq \sum_{i=1}^{m} w_i(\vec{\lambda}, \delta)(G_k - r_k(\vec{\ell}_i))\ell_{ik} \\
&\quad = G_k(\lambda_k + \delta) - \sum_{i=1}^{m} w_i(\vec{\lambda}, \delta) r_k(\vec{\ell}_i)\ell_{ik}.
\end{aligned}$$

If $Q_k(t) = 0$

$$\mathbb{E}\left[\left(G_k - r_k\left(\vec{\ell}^{\Delta^*(\vec{\lambda},\delta)}(t)\right)\right)\ell_k^{\Delta^*(\vec{\lambda},\delta)}(t) \,|\, \vec{Q}(t)\right] = 0.$$

Now, $\sum_{i=1}^{m} w_i(\vec{\lambda}, \delta) r_k(\vec{\ell}_i)\ell_{ik} \leq G_k(\lambda_k + \delta)$ since $r_k(\vec{\ell}_i) \leq G_k$ and $\sum_{i=1}^{m} w_i(\vec{\lambda}, \delta)\ell_{ik} = \lambda_k + \delta$. Thus,

$$\begin{aligned}
&\mathbb{E}\left[\left(G_k - r_k\left(\vec{\ell}^{\Delta^*(\vec{\lambda},\delta)}(t)\right)\right)\ell_k^{\Delta^*(\vec{\lambda},\delta)}(t) \,\middle|\, \vec{Q}(t)\right] \\
&\qquad \leq G_k(\lambda_k + \delta) - \sum_{i=1}^{m} w_i(\vec{\lambda}, \delta) r_k(\vec{\ell}_i)\ell_{ik}. \qquad (11)
\end{aligned}$$

From (4), (10), and (11), it follows that

$$\begin{aligned}
&\mathbb{E}\left[W\left(\vec{Q}(t), \vec{\ell}^{\Delta^*(\vec{\lambda},\delta)}(t)\right) \,\middle|\, \vec{Q}(t)\right] \\
&\qquad \geq \sum_{k=1}^{n} (Q_k(t) - VG_k)(\lambda_k + \delta) + VU(\vec{\lambda}, \delta).
\end{aligned}$$

Hence, from (9)

$$\begin{aligned}
&\mathbb{E}[f(\vec{Q}(t+1)) - f(\vec{Q}(t)) \,|\, \vec{Q}(t)] \\
&\quad \leq M - \sum_{k=1}^{n} 2\delta Q_k(t) \\
&\qquad + 2V\sum_{k=1}^{n} G_k(\lambda_k + \delta) - 2VU(\vec{\lambda}, \delta) \\
&\qquad - 2V\mathbb{E}\left[\sum_{k=1}^{n} [G_k - r_k(\vec{\ell}^{\Delta_O}(t))]\ell_k^{\Delta_O}(t) \,|\, \vec{Q}(t)\right]. \qquad (12)
\end{aligned}$$

1) Stability of Δ_o: From (12), since $0 \leq r_k(\vec{\ell}) \leq G_k, \ell_k \geq 0$ for all $k, \vec{\ell} \in \mathcal{L}$, it follows that for every stabilizable $\vec{\lambda}$ and every non-negative V

$$\mathbb{E}[f(\vec{Q}(t+1)) - f(\vec{Q}(t)) \,|\, \vec{Q}(t)] \leq M - \sum_{k=1}^{n} 2\delta Q_k(t) + 2\sum_{k=1}^{n} VG_k(\lambda_k + \delta).$$

Let $\mathcal{A} = \{\vec{Q} : \sum_{k=0}^{n} Q_k \leq (V/\delta)\sum_{k=0}^{n} G_k(\lambda_k + \delta) + (M+1)/(2\delta)\}$. Then,

$$\mathbb{E}[f(\vec{Q}(t+1)) - f(\vec{Q}(t)) \,|\, \vec{Q}(t)] < \begin{cases} \infty & \text{for all } \vec{Q}(t) \\ -1 & \text{if } \vec{Q}(t) \notin \mathcal{A}. \end{cases}$$

Thus, since $|\mathcal{A}|$ is finite, by Foster's Theorem ([20, Theorem 2.2.3]), $\{\vec{Q}(t)\}_{t\geq 1}$ is positive recurrent, and for each queue k the expected queue length under its stationary distribution is finite. Thus, the system is stable under Δ_O.

2) ϵ-Throughput Optimality of Δ_o: Taking expectation on both sides of (12) with respect to the stationary distribution of $\{\vec{Q}(t)\}_{t\geq 1}$, we obtain

$$\begin{aligned}\mathbb{E}[f(\vec{Q}(t+1)) - f(\vec{Q}(t))] &\leq M - \sum_{k=1}^{n} 2\delta \mathbb{E}[Q_k(t)] \\ &\quad + 2V\sum_{k=1}^{n} G_k(\lambda_k + \delta) - 2VU(\vec{\lambda}, \delta) \\ &\quad - 2V\mathbb{E}\left[\sum_{k=1}^{n}[G_k - r_k(\vec{\ell}^{\Delta_O}(t))]\ell_k^{\Delta_O}(t)\right]. \end{aligned} \quad (13)$$

Now, $\sum_{u=1}^{t} \ell_k^{\Delta_O}(t)$ is the number of departures from queue k in $(0,t)$ under Δ_O. Since the queue length process $\{\vec{Q}(t)\}_{t\geq 1}$ under Δ_O is a positive recurrent Markov chain, for every t

$$\begin{aligned}\mathbb{E}\left[\ell_k^{\Delta_O}(t)\right] &= \lim_{u\to\infty} \frac{1}{u}\sum_{v=1}^{u} \ell_k^{\Delta_O}(v) = \lambda_k \text{ w.p. } 1 \end{aligned} \quad (14)$$

and

$$\begin{aligned}\sum_{k=1}^{n}\mathbb{E}\left[r_k(\vec{\ell}^{\Delta_O}(t))\ell_k^{\Delta_O}(t)\right] &= \lim_{u\to\infty} \frac{1}{u}\sum_{v=1}^{u}\sum_{k=1}^{n} r_k(\vec{\ell}^{\Delta_O}(v))\ell_k^{\Delta_O}(v) \text{ w.p. } 1 \\ &= \Omega^{\Delta_O}(\vec{\lambda}) \quad (\text{from } (1)). \end{aligned} \quad (15)$$

Moreover, since the expectations are with respect to the stationary distribution of $\vec{Q}(t)$, it follows that

$$\mathbb{E}[f(\vec{Q}(t+1))] = \mathbb{E}[f(\vec{Q}(t))]. \quad (16)$$

From (13), (14), (15), and (16), it follows that

$$\begin{aligned}\Omega^{\Delta_O}(\vec{\lambda}) &\geq U(\vec{\lambda}, \delta) - \delta\sum_{k=1}^{n} G_k - \frac{M}{2V} \\ &\geq \Omega_{\max}(\vec{\lambda}) - \frac{\epsilon}{2} - \frac{M}{2V} \text{ (from Theorem 1)} \\ &\geq \Omega_{\max}(\vec{\lambda}) - \epsilon \text{ if } V \geq M/\epsilon. \end{aligned} \quad (17)$$

The result follows. ∎

Finally, we comment on the role of the parameter V in determining the throughput of Δ_O. From (17), it can be seen that if $V < M/\epsilon = n(A_{\max}^2 + 1)/\epsilon$, then no throughput guarantee can be provided for Δ_O. Note that $A_{\max}$ determines the burstiness of the arrival process. Thus, the minimum required value of V is higher for more bursty arrival processes.

C. Computation Time for Δ^* and Δ_o

In the worst case, cardinality of $\mathcal{L}$ can be 2^n as it may contain all n-dimensional binary vectors. Then, $\Delta^*(\vec{\lambda}, \delta)$ can be computed by solving a linear program with $O(2^n)$ variables and $O(n)$ constraints. Thus, the time and the memory required to compute $\Delta^*(\vec{\lambda}, \delta)$ is $O(2^n)$ in the worst case. Under Δ_O, we need to find a $\vec{\ell} \in \mathcal{L}$ that maximizes $W(\vec{Q}(t), \vec{\ell})$ for every t. Since $|\mathcal{L}|$ is $O(2^n)$, the time required to compute the optimal service vector in each slot is also $O(2^n)$ unless some additional structure on the queueing system is assumed. We next propose two optimal policies which require polynomial computation time in every slot.

V. Computationally Simple Optimal Policies

We provide a general framework for designing computationally simple policies for maximizing the throughput subject to attaining stability by considering the notion of *inaccurate* scheduling (Section V-A). We subsequently utilize this framework to design two computationally simple policies for maximizing the throughput subject to stabilizing the system (Sections V-B and V-C). Finally, we discuss how these policies can be implemented using distributed computation (Section V-D).

A. Inaccurate Scheduling for Maximizing the Throughput Subject to Stabilizing the System

We first describe a class of scheduling policies referred to as "inaccurate scheduling." Note that the notion of inaccurate scheduling has earlier been proposed for designing computationally simple policies for maximizing the stability region [23], [32], [34]. Our contribution here is to generalize this notion to attain the goal of maximizing the throughput subject to stabilizing the system while using simple computations.

We consider policies Δ for which the state $\{\vec{Y}(t) = (\vec{Q}(t), \vec{\ell}^{\Delta}(t))\}_{t\geq 1}$ constitutes an irreducible, aperiodic and countable Markov chain. This assumption holds when $\vec{\ell}^{\Delta}(t)$ is computed iteratively based on $\vec{Q}(t)$ and $\vec{\ell}^{\Delta}(t-1)$. Note that then $\{\vec{Q}(t)\}_{t\geq 1}$ may not be a Markov process.

Definition 8 (γ-Inaccurate Policy): A policy Δ_γ is called γ-inaccurate if in each slot t it selects a service vector $\vec{\ell}^{\Delta}(t)$ such that

$$W(\vec{Q}(t),\vec{\ell}^{\Delta}(t)) \geq \max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q}(t))}} W(\vec{Q}(t),\vec{\ell}) - X(\vec{Y}(t)) \quad (18)$$

where $X(\vec{Y}(t))$ is a random variable that depends on $\vec{Y}(t)$ (i.e., the distribution or $X(\vec{Y}(t))$ is determined by the current system state $\vec{Y}(t)$), and if $\{\vec{Y}(t)\}_{t\geq 1}$ has a stationary distribution then the expectation $\mathbb{E}[X(\vec{Y})]$ under the stationary distribution is less than or equal to γ. Any service vector that satisfies (18) is called a γ-inaccurate service vector.

Note that if γ is large, then the number of γ-inaccurate service vectors will be large and hence the time needed to find one such service vector may be small. *We show that for appropriate choices of V all stable γ-inaccurate policies are ϵ-throughput optimal.*

Theorem 1: Let $\vec{\lambda}$ be any stabilizable arrival rate vector and Δ_γ be an arbitrary γ-inaccurate policy. Then, for every $\epsilon > 0$ and $\gamma < \infty$, there exists $\hat{V}$ such that for every $V \geq \hat{V}$,

1) if $\{\vec{Y}(t)\}_{t\geq 1}$ is a positive recurrent Markov chain, then Δ_γ is ϵ-throughput optimal, and
2) if $\mathbb{E}[X(\vec{Y}) \,|\, \vec{Y}(t) = \vec{Y}] \leq \gamma$ for every $\vec{Y}$, then $\{\vec{Y}(t)\}_{t\geq 1}$ is a positive recurrent Markov chain, and Δ_γ stabilizes the system.

Now, we provide the intuition. For simplicity of explanation, we assume that $X(\vec{Y}) \leq \gamma$ for every $\vec{Y}$, and hence the condition in (2) of Theorem 3 holds. We first explain why γ-inaccurate policies maximize the stability region [34]. For large queue lengths

$$\max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q})}} W(\vec{Q},\vec{\ell}) \gg \gamma$$

and hence from (18)

$$W(\vec{Q},\vec{\ell}^{\Delta_\gamma}) \approx \max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q})}} W(\vec{Q},\vec{\ell}).$$

Thus, Δ_O and Δ_γ select similar service vectors when the queue lengths are large. We have shown that for every stabilizable $\vec{\lambda}$, Δ_O has a negative drift when the queue lengths are large. Thus, Δ_γ also has a negative drift for large queue lengths. Hence, Δ_γ stabilizes the system whenever $\vec{\lambda}$ is stabilizable. Incidentally, other approximate policies may also maximize the stability region. For example, any policy Δ that satisfies (18) with $W(\vec{Q}(t),\vec{\ell}^{\Delta}(t))$ and $W(\vec{Q}(t),\vec{\ell})$ replaced by $W_1(\vec{Q}(t),\vec{\ell}^{\Delta}(t))$ and $W_1(\vec{Q}(t),\vec{\ell})$, respectively, maximize the stability region [23], [32], [34].

The key difference between only stabilizing the system and attaining the maximum possible throughput subject to stabilizing the system is that whereas for the former it is sufficient to appropriately select the service vector when the queue lengths are large, but for the latter appropriate selection of service vectors is required for all values of queue lengths. Hence, it is not clear that Δ_γ maximizes the throughput as well; we now explain why this is in fact somewhat counter-intuitive. Note that for small queue lengths $\max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q})}} W(\vec{Q},\vec{\ell})$ may be smaller or comparable with γ. Then, (18) does not guarantee that the service vectors selected by Δ_γ and Δ_O are similar. Hence, it is not clear that Δ_γ achieves the same throughput as Δ_O, which attains the maximum throughput.

We now explain why Theorem 3 holds. We argue that for proper choice of parameters the queue lengths and the service vectors under Δ_O and Δ_γ become similar. Clearly, in the first slot, both systems have the same queue length vector, $\vec{Q}$. Now, note that for large $V, W(\vec{Q},\vec{\ell}_1)$ and $W(\vec{Q},\vec{\ell}_2)$ significantly differ if $\vec{\ell}_1$ and $\vec{\ell}_2$ are significantly different. Thus, due to (18), and since $W(\vec{Q},\vec{\ell}^{\Delta_\gamma}) \approx \max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q})}} W(\vec{Q},\vec{\ell})$, $\vec{\ell}^{\Delta_\gamma} \approx \vec{\ell}^{\Delta_O}$. Thus, the queue lengths in the next slot are also similar in both systems. Recursive use of the same argument shows that the queue lengths and the service vectors selected in each slot are similar in both systems. Thus, both policies attain similar throughput. Thus, Δ_γ is throughput optimal for large V.

Next, we prove Theorem 3.

Proof: We assume that $\vec{\lambda}$ is stabilizable. We define the following Lyapunov function:

$$f(\vec{Y}(t)) = \sum_{k=1}^{n} (Q_k(t))^2.$$

Using analysis similar to that for obtaining (8)

$$\begin{aligned}\mathbb{E}[f(\vec{Y}(t+1)) - f(\vec{Y}(t)) \,|\, \vec{Y}(t)] &\leq M + \sum_{k=1}^{n} 2Q_k(t)\lambda_k - 2\mathbb{E}[W(\vec{Q}(t),\vec{\ell}^{\Delta_\gamma}(t)) \,|\, \vec{Y}(t)] \\ &\quad - 2V\mathbb{E}\left[\sum_{k=1}^{n}[G_k - r_k(\vec{\ell}^{\Delta_\gamma}(t))]\ell_k^{\Delta_\gamma}(t) \,|\, \vec{Y}(t)\right]. \quad (19)\end{aligned}$$

From (18) and (19), it follows that

$$\begin{aligned}\mathbb{E}[f(\vec{Y}(t+1)) - f(\vec{Y}(t)) \,|\, \vec{Y}(t)] &\leq M + \sum_{k=1}^{n} 2Q_k(t)\lambda_k + 2\mathbb{E}[X(\vec{Y}(t)) \,|\, \vec{Y}(t)] \\ &\quad - 2\mathbb{E}\left[\max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q}(t))}} W(\vec{Q}(t),\vec{\ell}) \,|\, \vec{Y}(t)\right] \\ &\quad - 2V\mathbb{E}\left[\sum_{k=1}^{n}[G_k - r_k(\vec{\ell}^{\Delta_\gamma}(t))]\ell_k^{\Delta_\gamma}(t) \,|\, \vec{Y}(t)\right].\end{aligned}$$

Using arguments similar to those in the proof of (12) from (8), we can prove that

$$\begin{aligned}\mathbb{E}[f(\vec{Y}(t+1)) - f(\vec{Y}(t)) \,|\, \vec{Y}(t)] &\leq M + 2\mathbb{E}[X(\vec{Y}(t)) \,|\, \vec{Y}(t)] - \sum_{k=1}^{n} 2\delta Q_k(t) \\ &\quad + 2V\sum_{k=1}^{n} G_k(\lambda_k + \delta) - 2VU(\vec{\lambda},\delta) \\ &\quad - 2V\mathbb{E}\left[\sum_{k=1}^{n}[G_k - r_k(\vec{\ell}^{\Delta_\gamma}(t))]\ell_k^{\Delta_\gamma}(t) \,|\, \vec{Y}(t)\right] \quad (20)\end{aligned}$$

where δ is such that $\Delta^*(\vec{\lambda},\delta)$ is $\epsilon/2$-throughput optimal.

1) Proof for (1): Let the process $\{\vec{Y}(t)\}_{t\geq 1}$ be a positive recurrent Markov chain. Then this process has a stationary distribution. Taking expectation on both sides of (20) with respect to this stationary distribution, we obtain

$$\begin{aligned}\mathbb{E}[f(\vec{Y}(t+1))-f(\vec{Y}(t))] &\leq M+2\mathbb{E}\left[X(\vec{Y}(t))\right] \\ &\quad -\sum_{k=1}^{n}2\delta\mathbb{E}[Q_k(t)]+2V\sum_{k=1}^{n}G_k(\lambda_k+\delta) \\ &\quad -2VU(\vec{\lambda},\delta)-2V\mathbb{E}\left[\sum_{k=1}^{n}[G_k-r_k(\vec{\ell}^{\Delta_\gamma}(t))]\ell_k^{\Delta_\gamma}(t)\right].\end{aligned} \tag{21}$$

Since $\{\vec{Y}(t)\}_{t\geq 1}$ is a positive recurrent Markov chain

$$\begin{aligned}&\mathbb{E}\left[\ell_k^{\Delta_\gamma}(t)\right] \\ &\quad=\lim_{u\to\infty}\frac{1}{u}\sum_{v=1}^{u}\ell_k^{\Delta_\gamma}(v)=\lambda_k \text{ w.p. } 1\end{aligned} \tag{22}$$

and

$$\begin{aligned}&\sum_{k=1}^{n}\mathbb{E}\left[r_k(\vec{\ell}^{\Delta_\gamma}(t))\ell_k^{\Delta_\gamma}(t)\right] \\ &\quad=\lim_{u\to\infty}\frac{1}{u}\sum_{v=1}^{u}\sum_{k=1}^{n}r_k(\vec{\ell}^{\Delta_\gamma}(v))\ell_k^{\Delta_\gamma}(v) \text{ w.p. } 1 \\ &\quad=\Omega^{\Delta_\gamma}(\vec{\lambda}) \quad (\text{from } (1)).\end{aligned} \tag{23}$$

From stationarity,

$$\mathbb{E}[f(\vec{Y}(t+1))]=\mathbb{E}[f(\vec{Y}(t))]. \tag{24}$$

From (21), (22), (23), and (24), and since from Definition (8), $\mathbb{E}[X(\vec{Y}(t))]\leq\gamma$, it follows that

$$\begin{aligned}\Omega^{\Delta_\gamma}(\vec{\lambda}) &\geq U(\vec{\lambda},\delta)-\delta\sum_{k=1}^{n}G_k-\frac{M+2\gamma}{2V} \\ &\geq \Omega_{\max}(\vec{\lambda})-\frac{\epsilon}{2}-\frac{M+2\gamma}{2V} \text{ (from Theorem 1)} \\ &\geq \Omega_{\max}(\vec{\lambda})-\epsilon \text{if} V\geq\frac{M+2\gamma}{\epsilon}.\end{aligned}$$

The result follows.

2) Proof for (2): Now, let $\mathbb{E}[X(\vec{Y}(t))\,|\,\vec{Y}(t)=\vec{Y}]\leq\gamma$ for all $\vec{Y}$. Thus, from (20), for every γ and $V\geq 0$

$$\begin{aligned}&\mathbb{E}[f(\vec{Y}(t+1))-f(\vec{Y}(t))\,|\,\vec{Y}(t)] \\ &\qquad\leq M+2\gamma-\sum_{k=1}^{n}2\delta Q_k(t)+2V\sum_{k=1}^{n}G_k(\lambda_k+\delta).\end{aligned}$$

Let $\mathcal{B}=\{\vec{Y}=(\vec{Q},\vec{\ell}):\vec{\ell}\in\mathcal{L},\sum_{k=0}^{n}Q_k\leq(V/\delta)\sum_{k=0}^{n}G_k(\lambda_k+\delta)+(M+2\gamma+1)/(2\delta)\}$. Thus,

$$\mathbb{E}[f(\vec{Y}(t+1))-f(\vec{Y}(t))\,|\,\vec{Y}(t)]<\begin{cases}\infty & \text{for all } \vec{Y}(t)\\ -1 & \text{if } \vec{Y}(t)\notin\mathcal{B}.\end{cases}$$

Thus, since $|\mathcal{B}|$ is finite, by Foster's Theorem ([20, Theorem 2.2.3]), $\{\vec{Y}(t)\}_{t\geq 1}$ is positive recurrent, and the expectations of the queue lengths under its stationary distribution are finite. Hence, Δ_γ stabilizes the system. ■

The main challenge in computing γ-inaccurate service vectors is that $W(\vec{Q},\vec{\ell}^{\Delta_O})$ may not be known and in most cases its computation is complex. Thus, even the verification of whether a given $\vec{\ell}$ is γ-inaccurate may be computationally complex. We circumvent this challenge by designing a computationally simple approach that obtains γ-inaccurate service vectors without requiring the knowledge of $W(\vec{Q},\vec{\ell}^{\Delta_O})$.

B. Periodic Computation of Optimal Schedule

We divide the time axis in intervals of length T, i.e., in intervals of the form $[KT,(K+1)T-1]$.

$$\text{Let } \vec{\ell}^{\,\mathrm{OPT}}(t)\stackrel{\text{def}}{=}\arg\max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q}(t))}}W(\vec{Q}(t),\vec{l}).$$

We consider a policy Δ_T that computes $\vec{\ell}^{\,\mathrm{OPT}}(t)$ at the beginning of each interval, i.e., in the slots KT for $K\geq 0$, and throughout the interval serves each selected queue while it is non-empty.

The time needed to compute Δ_T is $O(2^n/T)$ in the amortized sense, i.e., $\sup_{t\geq 1}\{\sum_{u=1}^{t}c(u)/t\}$ is $O(2^n/T)$ on every sample path, where $c(u)$ is the computational complexity is slot u [17]. Thus, if we choose T to be sufficiently large ($\approx 2^n$), then Δ_T requires $O(1)$ computation time in the amortized sense.

In the following lemma, we show that $X(\vec{Y})\leq\gamma$ for all $\vec{Y}$ where $\gamma=nT(A_{\max}+2)$.

Lemma 1: Let $\vec{Q}(t)$ be the queue length vector under Δ_T in t. Then

$$W(\vec{Q}(t),\vec{\ell}^{\Delta_T}(t))\geq\max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q}(t))}}W(\vec{Q}(t),\vec{\ell})-nT(A_{\max}+2).$$

Proof: Without loss of generality let $t\in[KT,(K+1)T-1]$ for some K. Now, from (5)

$$\begin{aligned}\sum_{k=1}^{n}Q_k(t) &\leq \sum_{k=1}^{n}Q_k(KT)+\sum_{u=KT}^{t}\sum_{k=1}^{n}\Lambda_k(u) \\ &\leq \sum_{k=1}^{n}Q_k(KT)+nTA_{\max}.\end{aligned} \tag{25}$$

Similarly, from (5)

$$\begin{aligned}\sum_{k=1}^{n}Q_k(t) &\geq \sum_{k=1}^{n}Q_k(KT)-\sum_{u=KT}^{t}\sum_{k=1}^{n}\ell_k^{\Delta_T}(u) \\ &\geq \sum_{k=1}^{n}Q_k(KT)-nT.\end{aligned} \tag{26}$$

Now, from (4), (25), and (26), we obtain

$$\begin{aligned}
&\max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q}(t))}} W(\vec{Q}(t),\vec{\ell}) - W(\vec{Q}(t),\vec{\ell}^{\Delta_T}(t)) \\
&\quad\le \sum_{k=1}^{n}[Q_k(KT) - V(G_k - r_k(\vec{\ell}^{\mathrm{OPT}}(t)))]\ell_k^{\mathrm{OPT}}(t) \\
&\qquad - \sum_{k=1}^{n}[Q_k(KT) - V(G_k - r_k(\vec{\ell}^{\Delta_T}(t)))]\ell_k^{\Delta_T}(t) \\
&\qquad + nT(A_{\max}+1) \\
&\quad= W(\vec{Q}(KT),\vec{\ell}^{\mathrm{OPT}}(t)) - W(\vec{Q}(KT),\vec{\ell}^{\Delta_T}(t)) \\
&\qquad + nT(A_{\max}+1) \qquad (27)
\end{aligned}$$

Now, from (3)

$$W(\vec{Q}(KT),\vec{\ell}^{\mathrm{OPT}}(t)) \le W(\vec{Q}(KT),\vec{\ell}^{\mathrm{OPT}}(KT)). \qquad (28)$$

Also, since the service vector selected by Δ_T changes in the interval only if some queues empty during the period and then the change is to not serve them, $\vec{\ell}^{\Delta_T}(t) \le \vec{\ell}^{\mathrm{OPT}}(KT)$ and if $\ell_k^{\Delta_T}(t) < \ell_k^{\mathrm{OPT}}(KT)$ then $Q_k(KT) \le T$. Thus, $r_k(\vec{\ell}^{\Delta_T}(t)) \ge r_k(\vec{\ell}^{\mathrm{OPT}}(KT))$ for all k for which $Q_k(t) > 0$. Hence,

$$W(\vec{Q}(KT),\vec{\ell}^{\Delta_T}(t)) \ge W(\vec{Q}(KT),\vec{\ell}^{\mathrm{OPT}}(KT)) - nT. \qquad (29)$$

The result follows from (27), (28), and (29). ∎

However, $\vec{Y}(t)$ is not a Markov chain. Thus, in spite of Lemma 1, Δ_T is not γ-accurate. Now, $\vec{Y}(tT)$ is an irreducible, aperiodic, Markov chain, and the framework for γ-inaccurate scheduling can be generalized to such cases. We omit this generalization for brevity. But, using Lemma 1 and a proof similar to that for Theorem 3, we can prove that when $\vec{\lambda}$ is stabilizable, Δ_T is ϵ-throughput optimal for every $\epsilon > 0$. We formally state this in the following theorem, and prove it in the Appendix.

Theorem 4: Let $\vec{\lambda}$ be any stabilizable arrival rate vector. Then, for every $\epsilon > 0$, there exists $\hat{V}$ such that for every $V \ge \hat{V}$ the policy Δ_T is ϵ-throughput optimal.

The main challenge in using Δ_T is that it needs to periodically compute the optimal service vector. Since the time required in each such computation is exponential in n, for large n, such computations may become infeasible. We next propose an optimal randomized policy which requires $O(n)$ computation time in every slot.

C. Optimal Randomized Policy (Δ_R)

We now propose a randomized policy Δ_R which has been inspired by a randomized policy proposed by Tassiulas [34]. The policy in [34] attains the maximum possible stability region in a constrained queueing network using linear time computations in each slot. Our contribution here is to show that linear-time computable randomized policies can also maximize the throughput subject to stabilizing the system.

We now describe Δ_R. In every slot $t \ge 0, \Delta_R$ generates a service vector $\vec{\ell}(t)$ randomly among all service vectors $\vec{\ell} \in \mathcal{L}$ such that $\vec{\ell}(\vec{Q}(t)) = \vec{\ell}$ as per a distribution $P_{\vec{Q}}(\cdot)$. In every slot $t \ge 1$, once a random vector is generated as above, Δ_R obtains $\vec{\ell}^{\Delta_R}(t)$ iteratively, as shown by the equation at the bottom of the page. Thus, in any slot, Δ_R uses a new service vector only when it increases the value of $W(\cdot)$; otherwise it continues with the service vector used in the previous slot. It is interesting to observe that the randomized policy proposed by Tassiulas [34], which maximizes the stability region using linear computation time in each slot, uses a new service vector only when it increases the value of $W_1(\cdot)$.

Note that the distribution $P_{\vec{Q}}(\cdot)$ may depend on the current queue length vector. We only consider distributions $P_{\vec{Q}}(\cdot)$ such that for every $\vec{Q}$, $P_{\vec{Q}}(\vec{\ell}^{\mathrm{OPT}}) \ge \mu$ for some $\mu > 0$.

Lemma 2: Let $\vec{\lambda}$ be a stabilizable arrival rate vector. Then $\{\vec{Y}(t) = (\vec{Q}(t),\vec{\ell}^{\Delta_\gamma}(t))\}_{t\ge1}$ is a positive recurrent Markov chain, and Δ_R stabilizes the system.

We prove Lemma 2 in the Appendix. Now, we show that Δ_R is $(n/\mu)(A_{\max}+2)$-inaccurate.

Lemma 3: Let $\vec{Q}(t)$ be the queue length vector under Δ_R in t. Then, for any initial distribution of $\vec{Y}(t)$

$$\mathbf{E}\left[\max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q}(t))}} W(\vec{Q}(t),\vec{\ell}) - W(\vec{Q}(t),\vec{\ell}^{\Delta_R}(t))\right] \le \frac{n}{\mu}(A_{\max}+2).$$

Proof: Since $P_{\vec{Q}}(\vec{\ell}^{\mathrm{OPT}}) \ge \mu$ for every $\vec{Q}$, $\vec{\ell}^{\Delta_R}(t) = \vec{\ell}^{\mathrm{OPT}}(t)$ infinitely often w.p. 1. Let $\{\kappa_K\}_{K\ge1}$ be the slots in which $\vec{\ell}^{\Delta_R}(t) = \vec{\ell}^{\mathrm{OPT}}(t)$. Again, since $P_{\vec{Q}}(\vec{\ell}^{\mathrm{OPT}}) \ge \mu$ for every $\vec{Q}$, $\mathbb{E}[\kappa_{K+1} - \kappa_K] \le 1/\mu$.

Consider the K for which $t \in [\kappa_K, \kappa_{K+1}-1]$. Like in Lemma 1, we obtain

$$\begin{aligned}
&\max_{\substack{\vec{l}\in\mathcal{L}\\ \vec{l}=\vec{l}(\vec{Q}(t))}} W\left(\vec{Q}(t),\vec{\ell}\right) - W(\vec{Q}(t),\vec{\ell}^{\Delta_R}(t)) \\
&\qquad\qquad \le n(\kappa_{K+1} - \kappa_K)(A_{\max}+2).
\end{aligned}$$

Thus, the result follows since $\mathbb{E}[\kappa_{K+1} - \kappa_K] \le 1/\mu$. ∎

Now, from part 1 of Theorem 3 and Lemmas 2 and 3, it follows that Δ_R is ϵ-throughput-optimal for any μ. We formally state this in the following theorem.

Theorem 5: Let $\vec{\lambda}$ be a stabilizable arrival rate vector. Then, for every $\epsilon > 0$, there exists $\hat{V}$ such that for every $V \ge \hat{V}$ the policy Δ_R is ϵ-throughput optimal.

Now, if $\mu = 2^{-n}$, Δ_R can be computed in $O(n)$ time in each slot. Each non-empty queue can be selected w.p. 1/2. If the resulting vector is not in $\mathcal{L}$, then no queue is served.

$$\vec{\ell}^{\Delta_R}(t) = \begin{cases} \vec{\ell}(t), & W(\vec{Q}(t),(\vec{\ell}^{\Delta_R}(t-1))(\vec{Q}(t))) < W(\vec{Q}(t),\ \vec{\ell}(t)) \\ (\vec{\ell}^{\Delta_R}(t-1))(\vec{Q}(t)), & \text{otherwise} \end{cases}$$

D. Distributed Implementation of Δ_R and Δ_T

Distributed scheduling can be defined in different ways. One definition is to consider a policy as distributed if each node selects its action based on its observation, state and the information it acquires by exchanging messages with its neighbors. Such policies are then evaluated on the basis of their performance and the frequency and the amount of message exchange. Another definition is to consider a policy as distributed if each node selects its action based on its observation, state and the states and actions of nodes in a certain neighborhood.

We first describe how Δ_R and Δ_T can be implemented as per the first definition. The time axis can be divided in periods of length T. Each node can broadcast its queue length at the beginning of every period. The period length T should be selected so that the broadcasts in a period reach other nodes in the same period. For executing Δ_T, each node computes the optimal service vector at the beginning of every period based on the broadcasts it receives in the previous period. For executing Δ_R, each node randomly selects a service vector at the beginning of each period, and subsequently chooses between the service vectors selected in the current and previous periods based on the broadcasts it receives in the previous period, and finally uses the chosen service vector throughout the period. All nodes use the same seeds in the random number generators and therefore obtain the same random selections. For both policies, each node's computations depend on the queue lengths of other nodes in the previous period. Theorems 4 and 5 still hold. The message exchange complexity can be made arbitrarily small in both cases by increasing T.

Determination of an optimal policy which is distributed as per the second definition for distributed scheduling remains open. Note that the design of such scheduling policies in the precursor problem, that of maximizing the stability region, is still not completely understood, although some illuminating results have been obtained recently [10], [29], [36]. We hope that the optimality results in this paper and the recent advances in context of distributed scheduling will motivate further exploration of the above open problem.

Finally, Ross *et al.* has obtained local search based policies, which are likely to be computationally simple in practice, for maximizing the stability region of certain classes of constrained queueing networks [31]. It will be interesting to determine whether the throughput can be maximized subject to stabilizing the system using similar local search policies, and how the computation time required by the γ-accurate policies we propose compare with those for the resulting local search policies.

VI. Discussions and Generalizations

We now generalize our framework so as to obtain optimal policies when some of the assumptions made in Section II do not hold. First, we have so far assumed that a packet is discarded only after it is transmitted. We discuss how our framework can be generalized to allow a queue to discard some or all packets before transmitting them, and examine the advantages and disadvantages of this option (Section VI-A). We next describe how our framework can be generalized to accommodate random rewards and random sets of valid service vectors $\mathcal{L}$ (Section VI-B). Finally, we discuss how $\mathcal{L}$ and reward functions can be chosen so as to attain certain performance goals in an important application domain for this framework that of wireless networks (Section VI-C).

A. Discarding Packets Before Transmission

In Section II, we have assumed that each packet is discarded from its queue only after it is served once. However, in practice, a packet may be discarded from its queue even before it is served. The availability of this option enhances the stability region, and its judicious use increases the throughput. For example, in Example 1 in Section III, when $\vec{\lambda} = (1-\epsilon, 1-\epsilon)$ where ϵ is a small positive number, $\Omega_{\max}(\vec{\lambda}) \approx 4 (\Omega^{\hat{\Delta}} \approx 4)$. Now, if S_2 can discard packets before serving them, Δ' is stable and attains a throughput close to 5. But, clearly, indiscriminate use of this option substantially reduces the throughput.

We now show that appropriate augmentation of $\mathcal{L}$ allows us to design policies that attain the maximum possible throughput in presence of this option. Let $\mathcal{O}$ be the original system that does not allow packets to be discarded before transmission, and let $\hat{\mathcal{O}}$ be the new system which allows the above. In $\hat{\mathcal{O}}$, a queue is said to be served when a packet is removed from its queue. The service vectors in $\hat{\mathcal{O}}$ have $2n$ 0–1 components. The first n components denote which queues are being served and the remaining components denote whether the packets from the queues that are being served are transmitted or discarded before transmission. We obtain the set $\hat{\mathcal{L}}$ of valid service vectors of $\hat{\mathcal{O}}$ from the corresponding set $\mathcal{L}$ of $\mathcal{O}$ as follows. Let $\vec{\ell} \in \mathcal{L}$ and let $\vec{\ell}$ have i 0 components where $0 \leq i \leq n$. Now, $\vec{\ell}$ corresponds to 2^i service vectors in $\hat{\mathcal{L}}$, and each of these service vectors (a) transmit packets from the queues $\vec{\ell}$ were serving in $\mathcal{O}$ and (b) discard packets from a certain (possibly empty) subset of queues which $\vec{\ell}$ were not serving in $\mathcal{O}$. Note that the set of queues $\vec{\ell}$ were not serving in $\mathcal{O}$ has 2^i subsets. Thus, the number of service vectors generated by $\vec{\ell}$ is 2^i. Let $\vec{\hat{\ell}}_1$ be one such service vector generated by $\vec{\ell}$. Since $\vec{\ell}$ and $\vec{\hat{\ell}}_1$ transmit packets from the same queues, $r_k(\vec{\hat{\ell}}_1) = r_k(\vec{\ell})$ for each $k \in \{1, \ldots, n\}$.

The stability region of $\hat{\mathcal{O}}$ is a (possibly improper) superset of that of $\mathcal{O}$. This is because as long as the arrival rate of a queue is less than 1 it can be stabilized in $\hat{\mathcal{O}}$ by simply discarding all its packets before transmission. Thus, the stability region of $\hat{\mathcal{O}}$ is a superset of $\{\vec{\lambda} : 0 \leq \lambda_i < 1 \; i = 1, \ldots, n\}$ and a subset of $\{\vec{\lambda} : 0 \leq \lambda_i \leq 1 \; i = 1, \ldots, n\}$. For any $\vec{\lambda}$ that is stabilizable in $\mathcal{O}$, the maximum throughput of a stable policy in $\mathcal{O}$ is less than or equal to that of the maximum throughput of a stable policy in $\hat{\mathcal{O}}$. This is because every policy Δ in $\mathcal{O}$ is a valid policy in $\hat{\mathcal{O}}$, since for each $\vec{\ell} \in \mathcal{L}$ there exists $\vec{\hat{\ell}} \in \hat{\mathcal{L}}$ that does not discard packets from any queue before transmission, and transmits packets from the same queues which $\vec{\ell}$ serves. Note that $\Delta^*, \Delta_O, \Delta_\gamma, \Delta_T$, and Δ_R can be defined similar to that in $\mathcal{O}$; the only difference is that $\mathcal{L}$ must be substituted by $\hat{\mathcal{L}}$. The performance guarantees for these generalized versions, i.e., Theorems 1 to 5, hold in $\hat{\mathcal{O}}$. are the same as those for $\mathcal{O}$.

However, note that higher throughput and stability region can be attained in $\hat{\mathcal{O}}$ while sacrificing fairness. Specifically, for any $\vec{\lambda}$ that is stabilizable in $\mathcal{O}$, if an ϵ-throughput optimal

policy $\hat{\Delta}$ in $\hat{\mathcal{O}}$ attains a throughput which is higher than that of an ϵ-throughput optimal policy in $\mathcal{O}$, $\hat{\Delta}$ discards packets before transmission from some queues. Thus, $\hat{\Delta}$ attains higher throughput by being unfair to some queues. Also, in communication networks, in presence of this option, some receivers may only receive a small fraction of packets transmitted by the corresponding sources, which will in turn prevent them from successfully decoding the transmitted information. Thus, this option is not likely to be widely used (refer to Section VI-C).

B. Random Rewards and Random $\mathcal{L}$

We have so far assumed that the reward received by the kth queue in t is completely determined by the service vector chosen in t. We now allow the rewards to be random variables (r.v.'s) that depend on an external random component in addition to the service vector (Section VI-B). This generalization is relevant in context of wireless networks, where the success of a transmission is a random event whose probability depends on the fading state of the channels. Thus, in one-to-many or one-to-one communication the reward is a r.v. whose distribution depends on the service vector and the channel fading state between each sender-receiver pair. We generalize $\Delta^*, \Delta_O, \Delta_\gamma, \Delta_T$ and Δ_R so as to maximize the throughput subject to stabilizing the system in presence of random rewards.

We first formally describe the generalization. We consider a random process $\{\mathcal{S}(t)\}_{t\geq 1}$ which in any slot t is in state S_i with probability b_i, $i = 1,2,\ldots,z$, independent of its state in any other slot and also independent of the arrival process in any slot. Here, $b_i > 0$ for each $i \in \{1,\ldots,z\}$. The policy knows $\mathcal{S}(t)$ at the beginning of slot t. The reward received by the kth queue when $\vec{\ell}$ is the service vector and the process $\mathcal{S}(t)$ is in state S is a random variable, $R_k(\vec{\ell},S)$, whose distribution depends on $\vec{\ell}$ and S. Let $r_k(\vec{\ell},S) \stackrel{\text{def}}{=} \mathbb{E}[R_k(\vec{\ell},S)] \leq G_k$ for every $\vec{\ell}$ and S. We assume that $R_k(\vec{\ell},S) = 0$ if $\ell_k = 0$, and $r_k(\vec{\ell}_1,S) \geq r_k(\vec{\ell}_2,S)$ if $\vec{\ell}_1 \leq \vec{\ell}_2$ and $\ell_k > 0$. Thus, the throughput $\hat{\Omega}^{\Delta}(\vec{\lambda})$ under a policy Δ and arrival rate vector $\vec{\lambda}$ is

$$\begin{aligned}\hat{\Omega}^{\Delta}(\vec{\lambda}) &= \liminf_{t\to\infty} \frac{1}{t}\sum_{u=1}^{t}\sum_{k=1}^{n} R_k(\vec{\ell}^{\Delta}(u),\mathcal{S}(u)) \\ &= \liminf_{t\to\infty} \frac{1}{t}\sum_{u=1}^{t}\sum_{k=1}^{n} R_k(\vec{\ell}^{\Delta}(u),\mathcal{S}(u))\ell_k^{\Delta}(u).\end{aligned} \tag{30}$$

Finally, when the arrival rate vector is $\vec{\lambda}$, the maximum throughput of any stable policy is $\hat{\Omega}_{\max}(\vec{\lambda})$. We next elucidate the above formalisms with a specific example.

Example 4: In Fig. 1 assume that the channel to each receiver is in good (bad, resp.) state w.p. 0.8 (0.2), and each receiver can decode the packet w.p. 0.9 (0.2, resp.) when its channel is in good (bad, resp.) state and it is not in the range of any other sender that is transmitting packets. The state of a channel in a slot is independent of that in other slots and also independent of the states of other channels in any slot. In each slot, the system knows the states of all channels, but does not know whether a receiver can decode the packet its sender transmits. Thus, the system has 64 states corresponding to different combinations of channel states. Now, if $\vec{\ell} = \vec{\ell}_2$ ($\vec{\ell} = \vec{\ell}_4$, resp.) $R_1(\vec{\ell},S)$ equals the number of receivers in the set $\{R_1,\ldots,R_5\}$ ($\{R_3,R_4,R_5\}$, resp.) that can decode the packet S_1 transmits and if ($\vec{\ell} \in \{\vec{\ell}_1,\vec{\ell}_3\}$), $R_1(\vec{\ell},S) = 0$. Next, $R_2(\vec{\ell},S) = 1$ if $\ell_2 = 1$ and R_6 can decode the packet, $R_2(\vec{\ell},S) = 0$ otherwise. Thus, $R_1(\vec{\ell},S), R_2(\vec{\ell},S)$ are random variables whose distributions depend on $\vec{\ell},S$. For example, $r_1(\vec{\ell}_2,S) = 4.5$ if S is such that the channels to $R_1,\ldots,R_5$ are in good state, $r_2(\vec{\ell},S) = 0.9$ if $\ell_2 = 1$ and S is such that the channel to R_6 is in good state.

First, note that since $\mathcal{L}$ does not depend on $\mathcal{S}(t)$, the stability region of the system remains the same. Now, we present the optimality results. We first describe how $\Delta^*(\vec{\lambda},\delta)$ can be generalized. In any slot t in which $\mathcal{S}(t) = S_z$, the generalized policy $\hat{\Delta}^*(\vec{\lambda},\delta)$ selects $\vec{\ell}_i$ w.p. w_{iz}. If $\vec{\ell}_i$ is selected in slot t, then the system selects service vector $\vec{\ell}_i(\vec{Q}(t))$ (i.e., $\vec{\ell}^{\hat{\Delta}^*(\vec{\lambda},\delta)}(t) = \vec{\ell}_i(\vec{Q}(t))$). The probability distribution $\vec{w}_z = [w_{1z}\ \ldots\ w_{mz}]$ for every $z = 1,\ldots,Z$ is computed using the following linear program. $\hat{\mathbf{LP}}(\vec{\lambda},\delta)$: **−Maximize** : $\hat{U}(\vec{\lambda},\delta) = \sum_{z=1}^{Z} b_z[\sum_{i=1}^{m}\sum_{k=1}^{n} w_{iz}\ell_{ik}r_k(\vec{\ell}_i,S_z)]$

Subject to:

1) $\sum_{i=1}^{m} w_{iz} = 1$ for every $z \in \{1,\ldots,Z\}$.
2) $w_{iz} \geq 0$ for every $i \in \{1,\ldots,m\}$ and $z \in \{1,\ldots,Z\}$.
3) $\sum_{z=1}^{Z} b_z[\sum_{i=1}^{m} w_{iz}\ell_{ik}] = \lambda_k + \delta$ for every $k \in \{1,\ldots,n\}$.

Note that $\hat{\mathrm{LP}}(\vec{\lambda},\delta)$ is similar to $\mathrm{LP}(\vec{\lambda},\delta)$; the only difference is that the distribution for selecting the service vectors depends on the state $\mathcal{S}(t)$ of the system.

Theorem 6 (Generalization of Δ^*): Let $\vec{\lambda}$ be any stabilizable arrival rate vector. Then, for every $\epsilon > 0$ there exists a $\hat{\delta}$ such that for every $\delta \in (0,\hat{\delta}), \hat{\mathrm{LP}}(\vec{\lambda},\delta)$ is feasible, and $\hat{\Delta}^*(\vec{\lambda},\delta)$ is ϵ-throughput optimal. Furthermore

$$\begin{aligned}\hat{\Omega}^{\hat{\Delta}^*(\vec{\lambda},\delta)} &\geq \hat{U}(\vec{\lambda},\delta) - \delta\sum_{k=1}^{n} G_k \\ &\geq \hat{\Omega}_{\max}(\vec{\lambda}) - \epsilon \text{ w.p. } 1.\end{aligned} \tag{31}$$

Both the statement and proof for Theorem 6 are similar to that for Theorem 1. Hence, we do not prove Theorem 6.

Theorem 7 (Generalization for Δ_O): Consider a stabilizable arrival rate vector $\vec{\lambda}$ and a scheduling policy $\hat{\Delta}_O$ that chooses service vector $\vec{\ell}^{\hat{\Delta}_O}(t)$ such that

$$\vec{\ell}^{\hat{\Delta}_O}(t) = \arg\max_{\substack{\vec{\ell}\in\mathcal{L},\\ \vec{\ell}=\vec{\ell}(\vec{Q}(t))}} \left\{\sum_{k=1}^{n}(Q_k(t) - V\times(G_k - r_k(\vec{\ell},\mathcal{S}(t))))\ell_k\right\} \tag{32}$$

in every slot $t \geq 1$. Then, for every $V \geq 0$, $\hat{\Delta}_O$ stabilizes the system. Moreover, for every $\epsilon > 0$, there exists $\hat{V}$ such that for every $V \geq \hat{V}$, $\hat{\Delta}_O$ is ϵ-throughput optimal.

Note that the only difference between $\vec{\ell}^{\hat{\Delta}_O}(t)$ and $\vec{\ell}^{\Delta_O}(t)$ is that the former considers $r_k(\vec{\ell},\mathcal{S}(t))$ in selecting the service vector while the latter considers $r_k(\vec{\ell})$ in selecting the service vector. The statement of Theorem 2 is similar to that of Theorem 7. Using the fact that $\{\vec{Q}(t)\}_{t\geq 1}$ constitutes a Markov chain, Theorem 7 can be proved using similar arguments and the same

Lyapunov function as Theorem 2. We omit the proof for Theorem 7 for brevity.

We now generalize the framework for designing computationally simple policies for maximizing the throughput subject to stabilizing the system. We first generalize the notion of inaccurate scheduling.

$$\text{Let} \tau(t,S) \stackrel{\text{def}}{=} \begin{cases} 0 \text{if } \mathcal{S}(u) \neq S \; \forall \, u < t \\ \max\{u < t \, : \, \mathcal{S}(u) = S\} & \text{otherwise.} \end{cases}$$

Note that $\tau(t,S_i)$ is the last time instant before t such that the process $\mathcal{S}$ was in state S_i. Thus, for every $u \in \{\tau(t,S_i)+1,\ldots,t-1\}$, $S(u) \neq S_i$.

We consider policies Δ for which $\{\vec{I}(t) = (\vec{Q}(t), \vec{\ell}^{\Delta}(t), \vec{\ell}^{\Delta}(\tau(t,S_1)),\ldots,\vec{\ell}^{\Delta}(\tau(t,S_z)))\}_{t\geq 1}$ are irreducible, aperiodic and countable Markov chains. Let

$$\hat{W}(\vec{Q},\vec{\ell},S) \stackrel{\text{def}}{=} \sum_{k=1}^{n} (Q_k - V \times (G_k - r_k(\vec{\ell},S)))\ell_k. \quad (33)$$

Note that the only difference between $\hat{W}(\vec{Q},\vec{\ell},S)$ and $W(\vec{Q},\vec{\ell})$ is that the former depends on the expected rewards associated with both $\vec{\ell}$ and S, whereas the latter depends on the deterministic rewards associated with $\vec{\ell}$.

Definition 9 (Generalized γ-Inaccurate Policy): A policy $\hat{\Delta}_\gamma$ is called generalized-γ-inaccurate if in each slot t it selects a service vector $\vec{\ell}^{\Delta_\gamma}(t)$ such that

$$\hat{W}(\vec{Q}(t),\vec{\ell}^{\Delta_\gamma}(t),\mathcal{S}(t)) \geq \max_{\substack{\vec{\ell}\in\mathcal{L},\\ \vec{\ell}=\vec{\ell}(\vec{Q}(t))}} \hat{W}\left(\vec{Q}(t),\vec{\ell},\mathcal{S}(t)\right) - \hat{X}(\vec{I}(t)) \quad (34)$$

where $\hat{X}(\vec{I}(t))$ is a random variable that depends on $\vec{I}(t)$ and if $\vec{I}(t)$ has a stationary distribution, then the expectation $\mathbb{E}[\hat{X}(\vec{I}(t))]$ with respect to the stationary distribution, is upper bounded by γ.

The main difference between a γ-inaccurate policy and a generalized-γ-inaccurate policy is that the former seeks to approximate $\max_{\substack{\vec{\ell}\in\mathcal{L},\\ \vec{\ell}=\vec{\ell}(\vec{Q}(t))}} W(\vec{Q}(t),\vec{\ell})$ and the latter seeks to approximate $\max_{\substack{\vec{\ell}\in\mathcal{L},\\ \vec{\ell}=\vec{\ell}(\vec{Q}(t))}} \hat{W}(\vec{Q}(t),\vec{\ell},\mathcal{S}(t))$ at every time t.

We next show that for appropriate choice of V all stable generalized-γ-inaccurate policies are $\epsilon-$throughput optimal.

Theorem 8: Let $\vec{\lambda}$ be any stabilizable arrival rate vector and $\hat{\Delta}_\gamma$ an arbitrary generalized-γ-inaccurate policy. Then, for every $\epsilon > 0$ and $\gamma < \infty$, there exists $\hat{V}$ such that for every $V \geq \hat{V}$,

1) if $\{\vec{I}(t)\}_{t\geq 1}$ is a positive recurrent Markov chain, then $\hat{\Delta}_\gamma$ is ϵ-throughput optimal, and
2) if $\mathbb{E}[\hat{X}(\vec{I}) \,|\, \vec{I}(t) = \vec{I}] \leq \gamma$ for every $\vec{I}$, then $\{\vec{I}(t)\}_{t\geq 1}$ is a positive recurrent Markov chain, and $\hat{\Delta}_\gamma$ stabilizes the system.

Both the statement and proof for Theorem 8 are similar to that for Theorem 3; the only difference is that we consider $\vec{I}(t)$ as the system state in the former and $\vec{Y}(t)$ as the system state in the latter. We omit the proof for Theorem 8 for brevity.

Both Δ_T and Δ_R can be generalized using the framework of generalized-γ-inaccurate policies. For brevity, we only describe how Δ_R can be generalized. We denote the generalized version of Δ_R as $\hat{\Delta}_R$.

The policy $\hat{\Delta}_R$ obtains the service vector $\{\vec{\ell}^{\hat{\Delta}_R}(t)\}_{t\geq 1}$ as follows. In every slot $t \geq 0$, $\hat{\Delta}_R$ generates a random service vector $\vec{\ell}(t)$ among all service vectors $\vec{\ell} \in \mathcal{L}$ such that $\vec{\ell}(\vec{Q}(t)) = \vec{\ell}$ as per a distribution $P_{Q(\vec{t}),\mathcal{S}(t)}(\cdot)$ that may depend on $\vec{Q}(t)$ and $\mathcal{S}(t)$. In every slot $t \geq 1$, after generating the random service vector, $\hat{\Delta}_R$ obtains $\vec{\ell}^{\hat{\Delta}_R}(t)$ using the following iterative algorithm, as shown in (35) at the bottom of the page. We only consider distributions $P_{\vec{Q},S}(\cdot)$ such that for every $\vec{Q}, S$, $P_{\vec{Q},S}(\vec{\ell}^{\Delta_O}) \geq \mu$ for some $\mu > 0$.

First, we point out the key difference between Δ_R and $\hat{\Delta}_R$. In each slot t, Δ_R compares $W(\cdot)$ for the randomly generated service vector $\vec{\ell}(t)$ with $W(\cdot)$ under the service vector used in slot $t-1$. Now, $\hat{\Delta}_R$ compares $\hat{W}(\cdot)$ for the randomly generated service vector $\vec{\ell}(t)$ with $\hat{W}(\cdot)$ under the service vectors used in slots $t-1$ and $\tau(t,\mathcal{S}(t))$. For example, recall that there are 64 system states in Example 4. Let $\mathcal{S}(t) = S_5$, and $t_1,\ldots,t_{64}$ be the times at which states $1,\ldots,64$ were last encountered before t. Then $\hat{\Delta}_R$ compares $\hat{W}(\cdot)$ for the randomly generated service vector $\vec{\ell}(t)$ with $\hat{W}(\cdot)$ under the service vectors used in slots $t-1, t_5$. This additional comparison is necessary as the reward in the generalized system also depends on the state of the process $\mathcal{S}(t)$. Hence, a service vector $\vec{\ell}$ that maximizes $\hat{W}(\cdot)$ for some state S_j may not do so for some other state $S_{j'}$.

Theorem 9 (Generalization for Δ_R): Consider a stabilizable arrival rate vector $\vec{\lambda}$. Then, $\hat{\Delta}_R$ stabilizes the system for every $V \geq 0$. Moreover, for every $\epsilon > 0$, there exists a $\hat{V}$ such that for every $V \geq \hat{V}$, $\hat{\Delta}_R$ is ϵ-throughput optimal.

The statement of Theorem 9 is similar to that for Theorem 5. We prove Theorem 9 in the Appendix.

Note that we have so far assumed that the maximum number of packet arrivals in each slot t in any queue k is upper bounded by a finite constant $A_{\max}$. However, even when the above assumption is relaxed, as long as the arrival distribution has finite second moment, all the results, except Lemmas 1, 3 and Theorems 4, 5, 9 hold.

We finally consider the case where the set of allowed service vectors evolves randomly. Specifically, $\{\mathcal{L}(t)\}_{t\geq 1}$ evolves as per a finite state random process whose state in any slot is independent of that in any other slot and independent of the number

$$\vec{\ell}^{\hat{\Delta}_R}(t) = \arg \max_{\vec{\ell}\in\{\vec{\ell}(t),\ ((\vec{\ell}^{\hat{\Delta}_R}(t-1))(\vec{Q}(t))),\ ((\vec{\ell}^{\hat{\Delta}_R}(\tau(\mathcal{S}(t),t)))(\vec{Q}(t)))\}} \hat{W}(\vec{Q}(t),\vec{\ell},\mathcal{S}(t)). \quad (35)$$

of arrivals in any queue in any slot. The stability region is now different from that when $\mathcal{L}$ does not change. We refer to the interior of this new stability region as $\mathcal{C}'$. We assume that the policy knows $\mathcal{L}(t)$ at the beginning of slot t. All policies can be generalized to this case as well, using the framework of random rewards. Here, consider a new system in which the set of allowed service vectors is the power-set of the set of queues, and the reward for serving each queue is 0 in a slot if the service vector is not in $\mathcal{L}(t)$. The system is otherwise similar to the actual system. Theorems 7, 8, and 9 hold for all $\vec{\lambda} \in \mathcal{C}'$, and for $\hat{\Delta}_O$ and $\hat{\Delta}_R$ computed in the new system. For small ϵ, these policies rarely select service vectors in $\mathcal{L} \setminus \mathcal{L}(t)$ if $\vec{\lambda} \in \mathcal{C}'$.

C. Choice of $\mathcal{L}$ and Rewards in Wireless Networks

Our model allows each packet to be delivered to a subset of receivers, and therefore induces some packet loss. We can appropriately design the set $\mathcal{L}$ so as to ensure that the receivers can successfully decode the packets in presence of packet loss. For example, we can eliminate packet loss by restricting $\mathcal{L}$ to consist of only those service vectors that serve a queue only when its packet can be delivered to all receivers. For example, in Fig. 1, $\mathcal{L} = \{\vec{\ell}_1 = [0\ 0], \vec{\ell}_2 = [1\ 0], \vec{\ell}_3 = [1\ 0]\}$ will accomplish the above goal, but, observe that if $\mathcal{L}^{(1)} \subseteq \mathcal{L}^{(2)}$ then the stability region in a system where $\mathcal{L} = \mathcal{L}^{(1)}$ is a subset of that in a system where $\mathcal{L} = \mathcal{L}^{(2)}$. Also, for any $\vec{\lambda}$ which is in the stability region of both systems, the maximum throughput (minimum loss, resp.) of a stable policy in the former is greater than or equal to (less than or equal to, resp.) that in the latter. Thus, such restrictions on $\mathcal{L}$ should be imposed only when the system cannot tolerate any loss.

Many applications, e.g., real-time applications like audio, video, and some data applications like anycast[2] can inherently tolerate certain amount of packet loss. Applications can recover the information present in lost packets when they use coding redundancy (forward error correction [28], [30] or digital fountain [8]), path diversity (multiple transmissions of the same packet in different paths [21]), retransmissions at higher layers[3] (e.g., TCP or RTCP resend a packet at the transport layer if an end-to-end acknowledgement is not received within a time-out period). Also, for multicast transmissions a receiver may recover lost packets by requesting transmission from another receiver that has received the packet [9]. This "local recovery" is often useful if receivers are clustered and the distance between receivers in each cluster is significantly less than that between a receiver and the sender. $\mathcal{L}$ should be larger in all the above cases.

Thus, $\mathcal{L}$ must be chosen in accordance with application requirements and system design. The loss tolerant applications and also the mechanisms for recovering lost packets are effective only when either each packet is delivered to a certain minimum number of receivers, or each receiver receives a certain minimum fraction of packets transmitted by its source. The former is useful for anycast applications and local recovery mechanisms. The latter is useful for real time traffic, and in presence of loss recovery schemes like forward error correction, path diversity and retransmissions at higher layers. In the first case, $\mathcal{L}$ may be designed to consist of only those service vectors that deliver each packet of queue i to at least d_i receivers, where $(d_1, \ldots, d_n)$ can be determined based on application requirements and recovery mechanisms. Usually, $d_i > 0$ for each i, which in turn implies that packets cannot be discarded from the queues before transmission.

In the second case, $\mathcal{L}$ may be designed to consist of only those service vectors that ensure that each receiver receives a packet transmitted by its source with a certain minimum probability, which can in turn be determined in accordance with application requirements and system design (e.g., the amount of coding redundancy, multipath diversity and local recovery used). Note that the design of $\mathcal{L}$ under this requirement may be computationally hard as in the worst case each subset of the possible 2^n service vectors may need to be examined to determine whether the desired policy, e.g., one among Δ^*, Δ_O, Δ_γ, Δ_T, and Δ_R, attains the above goal. However, this computation needs to be performed once every time nodes move, and hence only once in static networks, and infrequently in networks where nodes move slowly. Furthermore, heuristic selection strategies may be used to ensure fast computation, e.g., heuristics for the coverage problems [22] may be used if we assume the knowledge of the probability that a service vector in $\mathcal{L}$ is selected by the given policy. Designing computationally simple algorithms for appropriately selecting $\mathcal{L}$ given the requirements of the application and the higher layer protocols and the service vector selection policy (e.g., one among Δ^*, Δ_O, Δ_γ, Δ_T) is a topic of future research.

Finally, the reward functions can also be appropriately selected so as to ensure that optimal policies prefer service vectors that facilitate successful decoding of information. For example, if a receiver has limited loss tolerance owing to application requirements and/or the nature of its loss recovery schemes, the reward associated with service vectors that deliver packets to this receiver can be made high. Appropriate selection of reward functions constitutes a topic for future research.

VII. Related Work

Tassiulas *et al.* have characterized the stability region of constrained queueing networks, and have obtained a scheduling policy that maximizes the stability region [35]. Several interesting generalizations of this basic result have been obtained in context of mild assumptions on arrival and service processes [1], [5], [18] and a diverse class of systems including wireless networks [19], [34], input queued switches [23], parallel processing systems [6], and manufacturing systems [2]. We consider constrained queueing networks where different queues receive different rewards for service, and more importantly, the reward obtained by the same queue may be different depending on the set of concurrently served queues. An important performance goal in such networks is to maximize the reward per unit time or the throughput subject to stabilizing the system. Our contribution has been to design a scheduling policy that attains this goal. We have earlier designed a scheduling policy that attains the same goal but only in a system with a single queue [12], [16].

[2]In anycast, a packet need only be delivered to a certain minimum number of receivers. An example application of anycast is a client-server query system. When a client needs to locate a service, it needs its query packet to reach a certain minimum number of servers.

[3]These retransmissions are treated as separate packets at lower layers.

Recently, Neely has considered a queueing system in which in each slot different queues can be simultaneously served at different rates [27]. The rate vector can be selected among some given choices, and different selections have different costs. In this scenario, Neely has proposed a scheduling policy that minimizes the cost while stabilizing the system. In our case, in each slot all queues that are served must be served at the same rate, but receive different rewards depending on the service vector. We maximize the total reward achieved per unit time subject to stabilizing the system. Thus, in some sense, we study the dual of the problem studied in [27]. Concurrent with our work, Stolyar has investigated a similar problem, and has proposed optimal policies similar to $\Delta^*(\vec{\lambda}, \delta)$ and Δ_O [33].[4] Our proof techniques are however significantly different, and also simpler, than that used by Stolyar. Furthermore, the optimal policies proposed by Stolyar, and also the basic optimal policies $\Delta^*(\vec{\lambda}, \delta)$, Δ_O we propose, turn out to be computationally complex. One of our important contributions has been to provide a general framework for designing optimal policies that are also computationally simple. The design of this general framework in turn relies on the techniques used for proving the optimality of $\Delta^*(\vec{\lambda}, \delta)$ and Δ_O.

Bonald *et al.* also showed that a policy that maximizes the instantaneous throughput does not attain the system stability region [7]. However, while they focus on a wire-line network we consider more general scheduling constraints. Also, they assume that flows arrive as per an arrival process and each flow arrives with a random number of packets, whereas we assume that the set of flows do not change but packets arrive in each flow as per an arrival process. Finally, the most important difference is that they investigated the tradeoff between fairness and stability, whereas we maximize the average throughput subject to stability.

We now describe some interesting open problems, and how some existing results can be used to solve these problems. We have assumed that the arrivals and the random rewards are temporally independent, and every packet can be served in 1 slot. An interesting direction for future research is to generalize our results for all stationary, ergodic arrival, service and reward processes. Several classes of policies have been shown to maximize the stability region in constrained queueing networks under the above mild assumptions on the arrival and service processes [1], [5], [18]. The analytical techniques proposed in these papers may be useful for the above generalizations in our context.

We have assumed that a packet can be transmitted at most once. Note that since each additional transmission increases the energy consumption, and the interference for other transmissions, several existing medium access policies, e.g., IEEE 802.11, transmit a packet only a bounded number of times, and subsequently discard the packet even if it has not reached some, or all, of its receivers. We assume this bound to be one which corresponds to a special case of the above. Note that in the broadcast mode IEEE 802.11 transmits every packet only once at the MAC layer, which is consistent with our assumption. An interesting open problem is to maximize the throughput subject to stabilizing the system when each packet can be transmitted up to k times where $k > 1$. We have recently proposed a policy that minimizes, in a network consisting of a single multicast sender, the amount of time each packets waits at the head of line position of the queue before it is transmitted, when each packet can be transmitted up to k times where k is a parameter [13]. It will be interesting to investigate whether similar results can be obtained for a network consisting of multiple queues and whether the guarantee on the waiting time at the head of line position can be used to obtain guarantees on the throughput and the stability region.

[4]Most of our results have been reported in [11], [14]. Both papers were submitted before Stolyar published his result [33].

APPENDIX
I. PROOF OF THEOREM 1

First, we prove two supporting lemmas (Section A) and subsequently prove Theorem 1 using these lemmas (Section B). In Lemma 4, we show that if $\vec{\lambda}$ is stabilizable, then there exists $\delta > 0$ such that $\mathrm{LP}(\vec{\lambda}, \delta)$ has a feasible solution. Thus, policy $\Delta^*(\vec{\lambda}, \delta)$ is well defined. In Lemma 5, we upper bound the throughput of any stable scheduling policy. For stating these lemmas, we generalize the definitions of $\mathrm{LP}(\vec{\lambda}, \delta)$ and $U(\vec{\lambda}, \delta)$. Let $\vec{\delta} = (\delta_1, \ldots, \delta_n)$.

LP$(\vec{\lambda}, \vec{\delta})$:- **Maximize**: $U(\vec{\lambda}, \vec{\delta}) = \sum_i \sum_k w_i \ell_{ik} r_k(\vec{\ell}_i)$.
Subject to:
1) $\sum_{i=1}^{m} w_i = 1$ and $w_i \geq 0$ for every i.
2) $\sum_{i=1}^{m} w_i \ell_{ik} = \lambda_k + \delta_k$ for every k.

Supporting Lemmas:

Lemma 4: Let $\vec{\lambda} \in \mathcal{C}$. Then, there exists a neighborhood $\mathcal{N}_0$ around $\vec{\lambda}$ such that $\mathrm{LP}(\vec{\lambda}, \vec{\delta})$ is feasible if $\vec{\lambda} + \vec{\delta} \in \mathcal{N}_0$.

Proof: Let $\vec{\lambda}_1 \in \mathcal{C}$. Since $\mathcal{C}$ is the interior of the convex hull of $\mathcal{L}$

$$\vec{\lambda}_1 = \sum_{i=1}^{m} c_i \vec{\ell}_i, \text{ where } c_i \geq 0, \forall\, i \text{ and } \sum_{i=1}^{m} c_i = 1.$$

Thus, $\vec{c} = [c_1 \ \cdots \ c_m]$ is a valid distribution. Moreover, $\vec{c}$ is a feasible solution for $\mathrm{LP}(\vec{\lambda}_1 - \vec{\delta}', \vec{\delta}')$ for every $\vec{\delta}'$.

Now, consider a given $\vec{\lambda} \in \mathcal{C}$. Since $\mathcal{C}$ is an open set, there exists an open ball centered at $\vec{\lambda}$ (denoted by $\mathcal{N}_0$) such that $\mathcal{N}_0 \in \mathcal{C}$. Thus, $\vec{\lambda} + \vec{\delta} \in \mathcal{C}$ for every $\vec{\lambda} + \vec{\delta} \in \mathcal{N}_0$. Now, as shown above, this implies that $\mathrm{LP}(\vec{\lambda}, \vec{\delta})$ has a feasible solution. ■

Lemma 5: For every stabilizable $\vec{\lambda}$, $\Omega_{\max}(\vec{\lambda}) \leq U(\vec{\lambda}, 0)$ w.p. 1.

Proof: Consider any policy Δ that stabilizes $\vec{\lambda}$.

Let $\xi_i^{\Delta}(t)$ denote the number of slots in which Δ uses $\vec{\ell}_i$ as the service vector till time t, i.e.,

$$\xi_i^{\Delta}(t) = \sum_{u=1}^{t} \mathbf{1}_{\{\vec{\ell}^{\Delta}(u) = \vec{\ell}_i\}}.$$

$$\text{Now, } \xi_i^{\Delta}(t) \geq 0, \text{ for every } t \geq 1, \tag{36}$$

$$\frac{\sum_{i=1}^{m} \xi_i^{\Delta}(t)}{t} = 1, \text{ for every } t \geq 1, \tag{37}$$

$$\lim_{t \to \infty} \frac{\sum_{i=1}^{m} \xi_i^{\Delta}(t) \ell_{ik}}{t} = \lambda_k \text{ w.p. 1 for every } k. \tag{38}$$

The last equality follows since Δ is stable.

Consider any t, and let for each k

$$c_{k,t} = \frac{\sum_{i=1}^{m} \xi_i^{\Delta}(t)\ell_{ik}}{t} - \lambda_k. \tag{39}$$

Let $\vec{c}_t = (c_{1,t}, \ldots, c_{n,t})$. Since from (36), (37) and (39), $(\xi_i^{\Delta}/(t)t)$ is a feasible solution of $\mathrm{LP}(\vec{\lambda}, \vec{c}_t)$

$$\frac{\sum_{i=1}^{m}\sum_{k=1}^{n} \xi_i^{\Delta}(t)\ell_{ik} r_k(\vec{\ell}_i)}{t} \leq U(\vec{\lambda}, \vec{c}_t). \tag{40}$$

We will show that given any $\iota > 0$, w.p. 1 there exists 1) a $\vec{\delta}$ such that $0 < \delta_i < \iota$ for all i, $1 \leq i \leq n$, and 2) $\hat{t}$ such that for every $t \geq \hat{t}$

$$\frac{\sum_{i=1}^{m}\sum_{k=1}^{n} \xi_i^{\Delta}(t)\ell_{ik} r_k(\vec{\ell}_i)}{t} \leq \sup_{-\vec{\delta}<\vec{\delta}'<\vec{\delta}} \{U(\vec{\lambda}, \vec{\delta}')\}. \tag{41}$$

From Lemma 4, there exists a neighborhood $\mathcal{N}_0$ around $\vec{\lambda}$ such that $\mathrm{LP}(\vec{\lambda}, \vec{\delta})$ is feasible if $\vec{\lambda} + \vec{\delta} \in \mathcal{N}_0$. Thus, given any $\iota > 0$, there exists a $\vec{\delta}$ such that $0 < \delta_i < \iota$ for all i, $1 \leq i \leq n$, and $\vec{\lambda} + \vec{\delta}' \in \mathcal{N}_0$ for all $\vec{\delta}'$ such that $-\vec{\delta} < \vec{\delta}' < \vec{\delta}$. Thus, $U(\vec{\lambda}, \vec{\delta}')$ is defined for all $\vec{\delta}'$ such that $-\vec{\delta} < \vec{\delta}' < \vec{\delta}$. From (38), w.p. 1 there exists $\hat{t}$ such that for every k, $t \geq \hat{t}$, $-\delta_k < c_{k,t} < \delta_k$. Thus, for every $t \geq \hat{t}$, $U(\vec{\lambda}, \vec{c}_t) \leq \sup_{-\vec{\delta}<\vec{\delta}'<\vec{\delta}}\{U(\vec{\lambda}, \vec{\delta}')\}$. Now, (41) follows from (40).

Now, from (41) and by the continuity of $U(\vec{\lambda}, \vec{\delta})$, w.p. 1

$$\liminf_{t\to\infty} \frac{\sum_{i=1}^{m}\sum_{k=1}^{n} \xi_i^{\Delta}(t)\ell_{ik} r_k(\vec{\ell}_i)}{t} \leq U(\vec{\lambda}, 0).$$

Since Δ is an arbitrary stable policy, the lemma follows from the above inequality, (1) and Definition 6.

Proof of Theorem 1:

Proof: From Lemma 4, there exists a neighborhood $\mathcal{N}_0$ around $\vec{\lambda}$ such that $\mathrm{LP}(\vec{\lambda}, \vec{\delta})$ is feasible if $\vec{\lambda} + \vec{\delta} \in \mathcal{N}_0$. Thus, there exists a $\delta_1 > 0$ such that $\vec{\lambda} + (\delta, \ldots, \delta) \in \mathcal{N}_0$ for all δ, $0 < \delta < \delta_1$. Thus, by Lemma 4, $\mathrm{LP}(\vec{\lambda}, \delta)$ has a feasible solution for every δ, $0 < \delta < \delta_1$.

Now, from continuity of $U(\vec{\lambda}, \delta)$ it follows that

$$\lim_{\delta\to 0}\left[U(\vec{\lambda}, \delta) - \delta\sum_{k=1}^{n} G_k\right] = U(\vec{\lambda}, 0).$$

Thus, from Lemma 5 it follows that for every $\epsilon > 0$ there exists $\hat{\delta} > 0$ such that for every $0 < \delta < \hat{\delta}$, $U(\vec{\lambda}, \delta)$ is well-defined and

$$U(\vec{\lambda}, \delta) - \delta\sum_{k=1}^{n} G_k \geq \Omega_{\max}(\vec{\lambda}) - \epsilon \text{ w.p. } 1. \tag{42}$$

Select δ such that $0 < \delta < \hat{\delta}$. Now, $\Delta^*(\lambda, \delta)$ is well-defined. Since $\Delta^*(\lambda, \delta)$ selects the kth queue for service w.p. $\lambda_k + \delta$, the rate at which the kth queue is offered service is greater than its arrival rate. Hence, $\{\vec{Q}(t)\}_{t\geq 1}$ under $\Delta^*(\vec{\lambda}, \delta)$ constitutes a positive recurrent Markov chain, and the expected queue lengths under the stationary distribution of this Markov chain are finite. Thus, $\Delta^*(\vec{\lambda}, \delta)$ is stable.

Let $\gamma_i(u) = 1$ if $\Delta^*(\vec{\lambda}, \delta)$ selects $\vec{\ell}_i$ in slot u, and 0 otherwise. Thus,

$$\lim_{t\to\infty} \frac{1}{t}\sum_{u=1}^{t} \gamma_i(u) = w_i(\vec{\lambda}, \delta) \text{ w.p. } 1, \tag{43}$$

$$\lim_{t\to\infty} \frac{1}{t}\sum_{u=1}^{t}\sum_{i=1}^{m} \gamma_i(u)\ell_{ik}\mathbf{1}_{\{Q_k(u)>0\}} = \lambda_k \text{ w.p. } 1, \tag{44}$$

and

$$\lim_{t\to\infty} \frac{1}{t}\sum_{u=1}^{t}\sum_{i=1}^{m} \gamma_i(u)\ell_{ik} = \lambda_k + \delta \text{ w.p. } 1. \tag{45}$$

Relation (44) follows because $\Delta^*(\vec{\lambda}, \delta)$ is stable and $\sum_{u=1}^{t}\sum_{i=1}^{m} \gamma_i(u)\ell_{ik}\mathbf{1}_{\{Q_k(u)>0\}}$ is the number of packets departing from the kth queue in $(0, t)$. Relation (45) follows from (43) and $\mathrm{LP}(\vec{\lambda}, \delta)$. Now

$$\begin{aligned}
&\Omega^{\Delta^*(\vec{\lambda},\delta)}(\vec{\lambda}) \\
&= \lim_{t\to\infty} \frac{\sum_{u=1}^{t}\sum_{k=1}^{n}\sum_{i=1}^{m} \gamma_i(u)\ell_{ik}(\vec{Q}(u)) r_k(\vec{\ell}_i(\vec{Q}(u)))}{t} \\
&\quad \text{(the limit exists w.p. 1 since } \{\vec{Q}(t)\}_{t\geq 1} \text{ under } \Delta^*(\vec{\lambda}, \delta) \\
&\quad \text{constitutes a positive recurrent Markov chain)} \\
&\geq \lim_{t\to\infty} \frac{\sum_{u=1}^{t}\sum_{k=1}^{n}\sum_{i=1}^{m} \gamma_i(u)\ell_{ik} r_k(\vec{\ell}_i)\mathbf{1}_{\{Q_k(u)>0\}}}{t} \\
&\quad (\text{since } r_k(\vec{\ell}_i) \geq r_k(\vec{\ell}_j) \text{ if } \vec{\ell}_i \leq \vec{\ell}_j \text{ and } \ell_{ik} > 0) \\
&= \lim_{t\to\infty} \frac{\sum_{u=1}^{t}\sum_{k=1}^{n}\sum_{i=1}^{m} \gamma_i(u)\ell_{ik} r_k(\vec{\ell}_i)}{t} \\
&\quad - \lim_{t\to\infty} \frac{\sum_{u=1}^{t}\sum_{k=1}^{n}\sum_{i=1}^{m} \gamma_i(u)\ell_{ik} r_k(\vec{\ell}_i)\mathbf{1}_{\{Q_k(u)=0\}}}{t} \\
&\geq U(\vec{\lambda}, \delta) - \lim_{t\to\infty} \frac{\sum_{u=1}^{t}\sum_{k=1}^{n} G_k \sum_{i=1}^{m} \gamma_i(u)\ell_{ik}\mathbf{1}_{\{Q_k(u)=0\}}}{t} \\
&\quad (\text{from}(43), \mathrm{LP}(\vec{\lambda}, \delta) \text{ and since } r_k(\vec{\ell}_i) \leq G_k \ \forall i) \\
&\geq U(\vec{\lambda}, \delta) - \delta\sum_{k=1}^{n} G_k \text{ w.p. } 1 \text{ (from (44) and (45))}.
\end{aligned} \tag{46}$$

The result follows from (42) and (46).

II. Proof of Theorem 4

Proof: Let $\vec{\lambda}$ be stabilizable and

$$f(\vec{Y}(t)) = \sum_{k=1}^{n} (Q_k(t))^2.$$

Using an analysis similar to that for obtaining (8)

$$\begin{aligned}
&\mathbb{E}[f(\vec{Y}((T+1)t)) - f(\vec{Y}(Tt))|\vec{Y}(Tt)] \\
&\quad \leq MT^2 + nA_{\max}T + T\sum_{k=1}^{n} 2Q_k(Tt)\lambda_k \\
&\qquad - 2\sum_{K=0}^{T-1} \mathbb{E}[W(\vec{Q}(Tt+K), \vec{\ell}^{\Delta_T}(Tt+K))|\vec{Y}(Tt)] \\
&\qquad - 2V\sum_{K=0}^{T-1} \mathbb{E}\left[\sum_{k=1}^{n}[G_k - r_k(\vec{\ell}^{\Delta_T}(Tt+K))]\right. \\
&\qquad \left. \times\ \ell_k^{\Delta_T}(Tt+K)|\vec{Y}(Tt)\right].
\end{aligned} \tag{47}$$

From Lemma 1 and (47), it follows that

$$\begin{aligned}
&\mathbb{E}[f(\vec{Y}((T+1)t)) - f(\vec{Y}(Tt))|\vec{Y}(Tt)] \\
&\quad \le MT^2 + nA_{\max}T + 2nT^2(A_{\max}+2) + T\sum_{k=1}^{n} 2Q_k(t)\lambda_k \\
&\qquad - 2\sum_{K=0}^{T-1}\mathbb{E}\left[\max_{\substack{\vec{\ell}\in\mathcal{L},\\ \vec{\ell}=\vec{\ell}(\vec{Q}(Tt+K))}} W(\vec{Q}(Tt+K),\vec{\ell})|\vec{Y}(Tt)\right] \\
&\qquad - 2V\sum_{K=0}^{T-1}\mathbb{E}\left[\sum_{k=1}^{n}[G_k - r_k(\vec{\ell}^{\Delta_T}(Tt+K))]\right. \\
&\qquad \left.\times\, \ell_k^{\Delta_T}(Tt+K)|\vec{Y}(Tt)\right].
\end{aligned}$$

Using arguments similar to those in the proof of (12) from (8), we can prove that

$$\begin{aligned}
&\mathbb{E}[f(\vec{Y}((T+1)t)) - f(\vec{Y}(Tt))|\vec{Y}(Tt)] \\
&\quad \le MT^2 + nA_{\max}T + 2nT^2(A_{\max}+2) - 2T\sum_{k=1}^{n}\delta Q_k(t) \\
&\qquad + 2VT\sum_{k=1}^{n} G_k(\lambda_k+\delta) - 2VTU(\vec{\lambda},\delta) \\
&\qquad - 2V\sum_{K=0}^{T-1}\mathbb{E}\left[\sum_{k=1}^{n}[G_k - r_k(\vec{\ell}^{\Delta_T}(Tt+K))]\right. \\
&\qquad \left.\times\, \ell_k^{\Delta_T}(Tt+K)|\vec{Y}(Tt)\right] \qquad (48)
\end{aligned}$$

where δ is such that $\Delta^*(\vec{\lambda},\delta)$ is $\epsilon/2$-throughput optimal.

1) Proof for Stability of Δ_T: From (48), for every $V \ge 0$

$$\begin{aligned}
&\mathbb{E}[f(\vec{Y}((T+1)t)) - f(\vec{Y}(Tt))|\vec{Y}(Tt)] \\
&\quad \le MT^2 + nA_{\max}T + 2nT^2(A_{\max}+2) - 2T\sum_{k=1}^{n}\delta Q_k(t) \\
&\qquad + 2VT\sum_{k=1}^{n} G_k(\lambda_k+\delta).
\end{aligned}$$

Let $\mathcal{B}_T = \{\vec{Y} = (\vec{Q},\vec{\ell}) : \vec{\ell} \in \mathcal{L}, \sum_{k=0}^{n} Q_k \le (V/\delta)\sum_{k=0}^{n} G_k(\lambda_k+\delta) + (MT + nA_{\max} + 2nT(A_{\max}+2)+1)/(2\delta)\}$. Thus,

$$\mathbb{E}[f(\vec{Y}((T+1)t)) - f(\vec{Y}(Tt))|\vec{Y}(t)] < \begin{cases} \infty & \text{for all } \vec{Y}(Tt) \\ -1 & \text{if } \vec{Y}(Tt) \notin \mathcal{B}_T. \end{cases}$$

Thus, since $|\mathcal{B}_T|$ is finite, by Foster's Theorem ([20, Theorem 2.2.3]), $\{\vec{Y}(Tt)\}_{t\ge 1}$ is positive recurrent and the expectations of the queue lengths under the stationary distribution of $\{\vec{Y}(Tt)\}_{t\ge 1}$ are finite. Thus, since the queue lengths in consecutive slots can differ only by a constant, Δ_T stabilizes the system.

2) Proof That $\Omega^{\Delta_T}(\vec{\lambda}) \ge \Omega_{\max}(\vec{\lambda}) - \epsilon$: Taking expectation on both sides of (48) with respect to the stationary distribution for $\{\vec{Y}(Tt)\}_{t\ge 1}$, we obtain

$$\begin{aligned}
&\mathbb{E}[f(\vec{Y}((T+1)t)) - f(\vec{Y}(Tt))] \\
&\quad \le MT^2 + nA_{\max}T + 2nT^2(A_{\max}+2) \\
&\qquad - 2T\sum_{k=1}^{n}\delta\mathbb{E}[Q_k(Tt)] \\
&\qquad + 2VT\sum_{k=1}^{n} G_k(\lambda_k+\delta) - 2VTU(\vec{\lambda},\delta) \\
&\qquad - 2V\sum_{K=0}^{T-1}\mathbb{E}\left[\sum_{k=1}^{n}[G_k - r_k(\vec{\ell}^{\Delta_T}(Tt+K))]\ell_k^{\Delta_T}(Tt+K)\right]. \qquad (49)
\end{aligned}$$

Since $\{\vec{Y}(Tt)\}_{t\ge 1}$ is a positive recurrent markov chain, and $\{\vec{Y}(t), t\%T\}_{t\ge 1}$ is a periodic markov chain with period T

$$\begin{aligned}
&\sum_{K=0}^{T-1}\mathbb{E}[\ell_k^{\Delta_T}(Tt+K)] \\
&\quad = T\lim_{u\to\infty}\frac{1}{u}\sum_{v=1}^{u}\ell_k^{\Delta_T}(v) = T\lambda_k \text{ w.p. } 1, \qquad (50)
\end{aligned}$$

and

$$\begin{aligned}
&\sum_{K=0}^{T-1}\sum_{k=1}^{n}\mathbb{E}[r_k(\vec{\ell}^{\Delta_T}(Tt+K))\ell_k^{\Delta_T}(Tt+K)] \\
&\quad = T\lim_{u\to\infty}\frac{1}{u}\sum_{v=1}^{u}\sum_{k=1}^{n} r_k(\vec{\ell}^{\Delta_T}(v))\ell_k^{\Delta_T}(v) \text{ w.p. } 1 \\
&\quad = T\Omega^{\Delta_T}(\vec{\lambda}) \quad (\text{from}(1)). \qquad (51)
\end{aligned}$$

$$\text{From stationarity}, \mathbb{E}[f(\vec{Y}((T+1)t))] = \mathbb{E}[f(\vec{Y}(Tt))]. \qquad (52)$$

From (49), (50), (51), and (52), it follows that

$$\begin{aligned}
&\Omega^{\Delta_T}(\vec{\lambda}) \\
&\quad \ge U(\vec{\lambda},\delta) - \delta\sum_{k=1}^{n} G_k - \frac{MT + nA_{\max} + 2nT(A_{\max}+2)}{2V} \\
&\quad \ge \Omega_{\max}(\vec{\lambda}) - \frac{\epsilon}{2} - \frac{MT + nA_{\max} + 2nT(A_{\max}+2)}{2V} \\
&\qquad (\text{from Theorem 1}) \\
&\quad \ge \Omega_{\max}(\vec{\lambda}) - \epsilon \text{ if } V \ge \frac{MT + nA_{\max} + 2nT(A_{\max}+2)}{\epsilon}.
\end{aligned}$$

The result follows since Δ_T stabilizes the system as well. ■

III. Proof of Lemma 2

Here, we outline the proof, but provide the complete proof in [15].

Let the system use Δ_R and the arrival rate vector $\vec{\lambda}$ be stabilizable. Let $\vec{Y}(t) = (\vec{Q}(t), \vec{\ell}^{\Delta_R}(t))$ represent the

system state under Δ_R. Consider a Lyapunov function: $f(\vec{Y}(t)) = f_1(\vec{Y}(t)) + f_2(\vec{Y}(t))$, where

$$f_1(\vec{Y}(t)) = \sum_{k=1}^{n} (Q_k(t))^2,$$
$$f_2(\vec{Y}(t)) = \left[\sum_{k=1}^{n} Q_k(t)\left(\ell_k^{\mathrm{OPT}}(t) - \ell_k^{\Delta_R}(t)\right)\right]^2.$$

Using similar technique as that in the proof of Proposition 1 of [34], we show that $\mathbb{E}[f(\vec{Y}(t+1)) - f(\vec{Y}(t)) \,|\, \vec{Y}(t) = \vec{Y}] < 0$ for all but the finite number of $\vec{Y}$'s. Thus, by Foster's theorem ([20, Theorem 2.2.3]), the process $\{\vec{Y}(t)\}_{t\geq 1}$ is a positive recurrent markov chain. Hence, the system is stable under Δ_R.

IV. Proof of Theorem 9

Let the system use $\hat{\Delta}_R$ and the arrival rate vector $\vec{\lambda}$ be stabilizable. Let $\vec{I}(t)$ be the system state under $\hat{\Delta}_R$. Using similar arguments as in the proof of Lemma 2, we can prove that $\{\vec{I}(t)\}_{t\geq 1}$ is a positive recurrent markov chain. Next, we outline the proof.

Let $\vec{\ell}^{\hat{\Delta}_O}(t)$ denote the service vector selected by $\hat{\Delta}_O$ in slot t if the queue length vector and the random process $\mathcal{S}$ at the beginning of t are $\vec{Q}(t)$ and $\mathcal{S}(t)$. As in the proof of Lemma 2, we consider the Lyapunov function $f(\vec{I}(t)) = f_1(\vec{I}(t)) + f_2(\vec{I}(t))$, where

$$f_1(\vec{I}(t)) = \sum_{k=1}^{n} (Q_k(t))^2,$$
$$f_2(\vec{I}(t)) = \left[\sum_{k=1}^{n} Q_k(t)\left(\ell_k^{\hat{\Delta}_O}(t) - \ell_k^{\hat{\Delta}_R}(t)\right)\right]^2.$$

Using $\hat{W}(\vec{Q}(t), \ldots, \mathcal{S}(t))$ instead of $W(\vec{Q}(t), \ldots)$, and the same arguments as in the proof of Lemma 2, we can show that 1) there exists a constant $\hat{B}_1 > 0$ such that $\mathbb{E}[f(\vec{I}(t+1)) - f(\vec{I}(t)) \,|\, \vec{I}(t)] < \hat{B}_1$ for all $\vec{I}(t)$, and 2) there exists a constant $\hat{B}_2 > 0$, such that $\mathbb{E}[f(\vec{I}(t+1)) - f(\vec{I}(t)) \,|\, \vec{I}(t)] < 0$ for all $\vec{I}(t)$ such that $\sum_{k=1}^{n} Q_k(t) \geq \hat{B}_2$. Thus, since $|\{\vec{I} : \sum_{k=1}^{n} Q_k < \hat{B}_2\}|$ is finite, the stability of $\hat{\Delta}_R$ follows from Foster's theorem ([20, Theorem 2.2.3]). Thus, the first part of Theorem 9 follows.

We now prove that $\hat{W}(\vec{Q}(t), \vec{\ell}^{\hat{\Delta}_O}(t), \mathcal{S}(t)) - \hat{W}(\vec{Q}(t), \vec{\ell}^{\hat{\Delta}_R}(t), \mathcal{S}(t)) \leq (n)/(\mu \min_i b_i)(A_{\max} + 1)$.[5] Then, the second part of Theorem 9 follows from the first part of Theorem 8.

Recall that $\mathbb{P}(\mathcal{S}(t) = S_i) = b_i$ for each $i \in \{1, \ldots, z\}$. Since $P_{\vec{Q},S}(\vec{\ell}^{\hat{\Delta}_O}) \geq \mu$ for every $\vec{Q}$, S, and $b_i > 0$ for each $i \in \{1, \ldots, z\}$, $(\vec{\ell}^{\hat{\Delta}_R}(t), \mathcal{S}(t)) = (\vec{\ell}^{\hat{\Delta}_O}(t), S_i)$ infinitely often w.p. 1 for each $i \in \{1, \ldots, z\}$. Let $\{\hat{\kappa}_{K,i}\}_{K\geq 1}$ be the slots in which $(\vec{\ell}^{\hat{\Delta}_R}(t), \mathcal{S}(t)) = (\vec{\ell}^{\hat{\Delta}_O}(t), S_i)$. Again, since $P_{\vec{Q},S}(\vec{\ell}^{\hat{\Delta}_O}) \geq \mu$ for every $\vec{Q}$, S, $\mathbb{E}[\hat{\kappa}_{K+1,i} - \hat{\kappa}_{K,i}] \leq 1/(b_i\mu)$ for each $i \in \{1, \ldots, z\}$.

[5]Note that henceforth all expectations are under the stationary distribution of the process $\{\vec{I}(t)\}$.

Without loss of generality, let $\mathcal{S}(t) = S_i$ and $t \in [\hat{\kappa}_{K,i}, \hat{\kappa}_{K+1,i} - 1]$ for some K, i. Now

$$Q_k(t) \leq Q_k(\hat{\kappa}_{K,i}) + A_{\max}(\hat{\kappa}_{K+1,i} - \hat{\kappa}_{K,i}) \,\forall\, k. \quad (53)$$
$$Q_k(t) \geq Q_k(t-1) - 1 \,\forall\, k. \quad (54)$$
$$\hat{W}(\vec{Q}(t), \vec{\ell}^{\hat{\Delta}_O}(t), \mathcal{S}(t)) \leq \hat{W}(\vec{Q}(\hat{\kappa}_{K,i}), \vec{\ell}^{\hat{\Delta}_O}(t), \mathcal{S}(t)) + nA_{\max}(\hat{\kappa}_{K+1,i} - \hat{\kappa}_{K,i})$$
(from (33) and (53) and since for every $\ell \in \mathcal{L}, i = 1, \ldots, z$
$$\hat{W}(\vec{Q}_1, \vec{\ell}, S_i) - \hat{W}(\vec{Q}_1, \vec{\ell}, S_i) \leq \sum_{j=1}^{n} (Q_{1j} - Q_{2j})\ell_j)$$
$$\leq \hat{W}(\vec{Q}(\hat{\kappa}_{K,i}), (\vec{\ell}^{\hat{\Delta}_O}(t))(\vec{Q}(\hat{\kappa}_{K,i})), \mathcal{S}(t)) + nA_{\max}(\hat{\kappa}_{K+1,i} - \hat{\kappa}_{K,i})$$
(since $\hat{W}(\vec{Q}, \vec{\ell}, S_i) \leq \hat{W}(\vec{Q}, \vec{\ell}(\vec{Q}), S_i) \,\forall\, \vec{\ell}, \vec{Q}, i = 1, \ldots, z$)
$$\leq \max_{\substack{\vec{\ell}\in\mathcal{L}, \\ \vec{\ell}=\vec{\ell}(\vec{Q}(\hat{\kappa}_{K,i}))}} \hat{W}(\vec{Q}(\hat{\kappa}_{K,i}), \vec{\ell}, \mathcal{S}(t)) + nA_{\max}(\hat{\kappa}_{K+1,i} - \hat{\kappa}_{K,i})$$
$$= \hat{W}(\vec{Q}(\hat{\kappa}_{K,i}), \vec{\ell}^{\hat{\Delta}_O}(\hat{\kappa}_{K,i}), \mathcal{S}(t)) + nA_{\max}(\hat{\kappa}_{K+1,i} - \hat{\kappa}_{K,i})$$
(from (32) since $\mathcal{S}(t) = S(\hat{\kappa}_{K,i})$). (55)

Next

$$\hat{W}(\vec{Q}(t), \vec{\ell}^{\hat{\Delta}_R}(t), \mathcal{S}(t))$$
$$\geq \hat{W}(\vec{Q}(t), ((\vec{\ell}^{\hat{\Delta}_R}(\tau(t, \mathcal{S}(t)))(\vec{Q}(t))), \mathcal{S}(t)) \text{(from (35))}$$
$$\geq \hat{W}(\vec{Q}(t),\ \vec{\ell}^{\hat{\Delta}_R}(\tau(t, \mathcal{S}(t))), \mathcal{S}(t))$$
$$\geq \hat{W}(\vec{Q}(\tau(t, \mathcal{S}(t))),\ \vec{\ell}^{\hat{\Delta}_R}(\tau(t, \mathcal{S}(t))),\ \mathcal{S}(t)) - n(t - \tau(t, \mathcal{S}(t))) \text{(from (54))}.$$

Thus, since $\hat{\kappa}_{K,i} = \tau(\tau(\cdots\tau(t, S_i), S_i \cdots), S_i)$

$$\hat{W}(\vec{Q}(t), \vec{\ell}^{\hat{\Delta}_R}(t), \mathcal{S}(t)) \geq \hat{W}(\vec{Q}(\hat{\kappa}_{K,i}), \vec{\ell}^{\hat{\Delta}_R}(\hat{\kappa}_{K,i}), \mathcal{S}(t)) - n(\hat{\kappa}_{K+1,i} - \hat{\kappa}_{K,i}). \quad (56)$$

Thus, from (55) and (56), and since from the definition of $\hat{\kappa}_{K,i}$ $\vec{\ell}^{\hat{\Delta}_R}(\hat{\kappa}_{K,i}) = \vec{\ell}^{\hat{\Delta}_O}(\hat{\kappa}_{K,i})$

$$\hat{W}(\vec{Q}(t), \vec{\ell}^{\hat{\Delta}_O}(t), \mathcal{S}(t)) - \hat{W}(\vec{Q}(t), \vec{\ell}^{\hat{\Delta}_R}(t), \mathcal{S}(t)) \leq n(A_{\max} + 1)(\hat{\kappa}_{K+1,i} - \hat{\kappa}_{K,i}).$$

The result follows since $\mathbb{E}[\hat{\kappa}_{K+1,i} - \hat{\kappa}_{K,i}] \leq 1/(b_i\mu)$ for each $i \in \{1, \ldots, z\}$. ■

References

[1] M. Armony and N. Bambos, "Queueing dynamics and maximal throughput scheduling in switched processing systems," *Queueing Syst.*, vol. 44, no. 3, pp. 209–252, Jul. 2003.

[2] N. Bambos and G. Michailidis, "On the stationary dyanmics of parallel queues with random server connectivity," in *Proc. 34th IEEE Conf. Decision Control*, New Orleans, LA, Dec. 1995, pp. 3638–3643.

[3] N. Bambos and G. Michailidis, "On parallel queueing with random server connectivity and routing constraints," *Probability in the Eng. Inf. Sci.*, vol. 16, no. 2, pp. 185–203, 2002.
[4] N. Bambos and G. Michailidis, "Queueing and scheduling in random environment," *Adv. Appl. Probability*, vol. 36, no. 1, pp. 293–317, Mar. 2004.
[5] N. Bambos and G. Michailidis, "Queueing networks of random link topology: Stationary dynamics of maximal throughput schedules," *Queueing Syst.*, vol. 50, no. 1, pp. 5–52, May 2005.
[6] N. Bambos and J. Walrand, "Maximal throughput for stability of a class of parallel processing systems," in *Proc. 29th IEEE Conf. Decision Control*, Honolulu, HI, Dec. 1990, pp. 161–166.
[7] T. Bonald and L. Massoulie, "Impact of fairness on internet performance," in *Proc. ACM SIGMETRICS*, Cambridge, MA, Jun. 2001.
[8] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 28, no. 4, pp. 56–67, Oct. 1998.
[9] R. Chandra, V. Ramasubramaniam, and P. Birman, "Anonymous gossip: Improving multicast reliability in mobile ad-hoc networks," in *Proc. 21st Int. Conf. Distributed Comput. Syst. (ICDCS)*, Phoenix, AZ, 2001, pp. 275–283.
[10] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput guarantees through maximal scheduling in multihop wireless networks," in *Proc. 43d Annu. Allerton Conf. Communication, Control, Comput.*, Allerton, Monticello, IL, Sep. 28–30, 2005.
[11] P. Chaporkar and S. Sarkar, "Utility optimal scheduling for general reward states and stability constraint," in *Proc. 42nd IEEE Conf. Decision Control (CDC)*, Seville, Spain, Dec. 2005, pp. 5132–3137.
[12] P. Chaporkar and S. Sarkar, "Wireless multicast: Theory and approaches," *IEEE Trans. Inf. Theory*, vol. 51, no. 6, pp. 1954–1972, Jun. 2005.
[13] P. Chaporkar and S. Sarkar, "Minimizing delay in loss-tolerant mac layer multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4701–4713, Oct. 2006.
[14] P. Chaporkar and S. Sarkar, "Stable scheduling policies for maximizing throughput in generalized constrained queueing systems," in *Proc. IEEE INFOCOM*, Barcelona, Spain, Apr. 2006, pp. 1–13.
[15] P. Chaporkar and S. Sarkar, "Stable scheduling policies for maximizing throughput in generalized constrained queueing systems," Indian Inst. of Technol. Bombay, Tech. Rep., 2008 [Online]. Available: http://www.ee.iitb.ac.in/wiki/faculty/chaporkar
[16] P. Chaporkar, R. Shetty, and S. Sarkar, "Dynamic quorum policy for maximizing throughput in limited information mac," *IEEE/ACM Trans. Netw.*, vol. 14, no. 4, pp. 835–848, Aug. 2006.
[17] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*. Cambridge, MA: MIT Press, 2001.
[18] J. Dai and B. Prabhakar, "The throughput of data switches with and without speedup," in *Proc. INFOCOM*, Tel Aviv, Israel, Mar. 2000, pp. 556–564.
[19] A. Eryilmaz and R. Srikant, "Stable scheduling policies for fading wireless channels," *IEEE/ACM Trans. Netw.*, vol. 13, no. 2, pp. 411–424, 2005.
[20] G. Fayolle, V. A. Malyshev, and M. V. Menshikov, *Topics in the Constructive Theory of Countable Markov Chains.*. Cambridge, MA: Cambridge Univ. Press, 1995.
[21] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 5, no. 4, pp. 11–25, Oct. 2001.
[22] D. Hochbaum, *Approximation Algorithms for NP-Hard Problems.*. New York: Thomson Learning, 1996.
[23] I. Keslassy and N. McKeown, "Analysis of scheduling algorithms that provide 100% throughput in input-queued switches," in *Proc. 39th Annu. Allerton Conf. Commun., Control, Comput.*, Monticello, IL, Oct. 2001, CD-ROM.
[24] P. Kumar and S. Meyn, "Stability of queueing networks and scheduling policies," *IEEE Trans. Autom. Control*, vol. 40, no. 2, pp. 251–260, Feb. 1995.
[25] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks," in *Proc. IEEE INFOCOM'05*, Miami, FL, Mar. 2005, pp. 1804–1814.
[26] S. Meyn, "Stability, performance evaluation, and optimization," in *Markov Decision Processes: Models, Methods, Directions, and Open Problems*, ser. Kluwer Series on Operations Research. Norwell, MA: Kluwer, 2000.
[27] M. Neely, "Energy optimal control for time varying wireless networks," *Proc. IEEE INFOCOM'05*, pp. 2915–2934, Mar. 2005.
[28] J. Nonnenmacher, E. Biersack, and D. Towsley, "Parity-based loss recovery for reliable multicast transmission," *IEEE/ACM Trans. Netw.*, vol. 6, no. 4, pp. 349–361, Aug. 1997.
[29] S. Ray and S. Sarkar, "Arbitrary throughput versus complexity tradeoffs in wireless networks using graph partitioning," in *Proc. 2nd Workshop Inf. Theory and Applicat.*, San Diego, CA, Jan. 2007, CD-ROM.
[30] L. Rizzo, "Effective erasure codes for reliable computer communication protocols," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 27, no. 2, pp. 24–36, Apr. 1997.
[31] K. Ross and N. Bambos, "Local search scheduling algorithms for maximal throughput in packet switches," in *Proc. INFOCOM*, Hong Kong, China, 2004, pp. 1158–1169.
[32] S. Sarkar and L. Tassiulas, "A framework for routing and congestion control for multicast information flows," *IEEE Trans. Inf. Theory*, vol. 48, no. 10, pp. 2690–2708, Oct. 2002.
[33] A. Stolyar, "Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm," *Queueing Syst.*, vol. 50, no. 6, pp. 401–457, Aug. 2005.
[34] L. Tassiulas, "Linear complexity algorithms for maximum throughput in radio networks and input queued switches," in *Proc. INFOCOM'98*, 1998, pp. 533–539.
[35] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.
[36] X. Wu and R. Srikant, "Regulated maximal matching: A distributed scheduling algorithm for multihop wireless networks with node-exclusive spectrum sharing," in *Proc. IEEE CDC-ECC'05*, Seville, Spain, Dec. 2005, pp. 5342–5347.

Prasanna Chaporkar (S'01–M'07) received the Ph.D. degree in electrical and systems engineering from the University of Pennsylvania, Philadelphia, PA.

He is currently an Assistant Professor in electrical engineering at the Indian Institute of Technology, Mumbai, India. His research interests are in resource allocation in communication networks, stochastic control, and queueing theory.

Saswati Sarkar (S'98–M'00) received the Ph.D. in electrical and computer engineering from University of Maryland, College Park, in 2000.

She is currently an Associate Professor in the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA. Her research interests are in resource allocation and performance analysis in communication networks.