Stable Scheduling Policies for Maximizing Throughput in Generalized Constrained Queueing Systems

Prasanna Chaporkar and Saswati Sarkar

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 scheduled. Whenever a queue is scheduled for service, the system transmits a packet and receives a certain reward. 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 scheduled queues. We demonstrate that the dependence of the rewards on the schedules alter fundamental relations between performance metrics like throughput and stability. Specifically, maximizing 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 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, stability, throughput, optimization, wireless, multicast, randomized algorithms**

I. INTRODUCTION

Constrained queueing networks have been extensively used to model several systems of practical interest including wireless networks [1], [2], [3], [4], [5], [6], [7], [8], input queued switches [9], [10] and database systems [4]. A constrained queueing network is a collection of queues such that only certain sets of queues can be concurrently scheduled; these "schedulable sets" depend on the underlying system. Whenever a queue is scheduled for service, the system transmits a packet and therefore receives a certain reward. In such systems, queues need to be scheduled such that (i) the total reward earned by the system per unit time ("throughput") is maximized and (ii) 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 in the last two decades [1], [2], [5], [6], [11], [12], [13], [14], [15], [16], [17], [18].

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 scheduled 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 trade-off 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, the set of receivers that can successfully decode each transmission depends on other transmissions in their neighborhoods. Thus, the reward associated with each transmission depends on the schedule. For example in Figure 1 when S_2 is transmitting to receiver R_6, receivers 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 reward 5 when S_1 alone is scheduled, and it receives reward 3 when S_1 and S_2 are scheduled together. Thus, the reward for S_1 depends on the schedule.

Now, consider one-to-one communication in wireless networks. Success of each transmission depends upon 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 schedule. 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.

Next, in many database systems, a single update opera-

P. Chaporkar is with INRIA, Paris, France. His email address is Prasanna.Chaporkar@ens.fr. S. Sarkar is with the Department of Electrical and Systems Engineering at University of Pennsylvania. Her email address is swati@seas.upenn.edu. This work was supported by the National Science Foundation under grants ANI-0106984, NCR-0238340 and CNS-0435306.

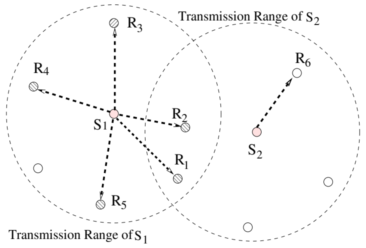

Fig. 1. Figure shows an example to demonstrate the application of generalized constrained queueing networks in one-to-many communication in wireless networks. There are two senders S_1, S_2 and 6 receivers R_1 to R_6. The dashed circles indicate the communication ranges of the senders. A single transmission from S_1 can reach all its receivers, $R_1, \ldots, R_5$.

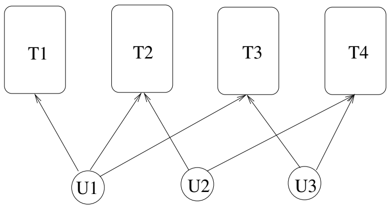

Fig. 2. Figure shows a database system with four tables $T1, \ldots, T4$ that are accessed by three applications $U1, U2$ and $U3$. The arrows indicate the tables each application updates. When there are concurrent requests for updates in the same table, the request from an application with the lowest 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.

tion 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 schedule. We demonstrate this using a specific application in Figure 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 schedules (Section III). Specifically, maximizing 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 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). We conclude in Section VIII.

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 $\Lambda_k(t) \le A_{\max}$ 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 a single slot. After service each packet departs the system. We denote by $Q_k(t)$ the queue length in k^{th} 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. The indicator $\ell_k(t) = 1$ if the k^{th} queue is served in slot t, and is 0 otherwise. The vector $\vec{\ell}(t) = [\ell_1(t) \ \cdots \ \ell_n(t)]$ denotes the schedule 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 schedule. Let $\mathcal{L} = \{\vec{\ell}_1, \ldots, \vec{\ell}_m\}$ denote the set of all valid schedules. Clearly $m \le 2^n$. For example, in Figure 1, let S_1 (S_2, resp.) indicate queue 1 (2, resp.). Thus, $k = 2$, and $\mathcal{L}$ can be $\left\{\vec{\ell}_1 = [0\ 0], \vec{\ell}_2 = [1\ 0], \vec{\ell}_3 = [0\ 1], \vec{\ell}_4 = [1\ 1]\right\}$. If the system has an additional constraint that all the receivers should receive every packet, then both S_1 and S_2 can not be scheduled concurrently. Thus, in this case, $\mathcal{L} = \left\{\vec{\ell}_1 = [0\ 0], \vec{\ell}_2 = [1\ 0], \vec{\ell}_3 = [0\ 1]\right\}$.

We assume the following about $\mathcal{L}$. If $\vec{\ell} \in \mathcal{L}$, then every $\vec{\ell}_1 \le \vec{\ell}$ also belongs to $\mathcal{L}$, where the inequality is element-wise. In other words, if a certain set of queues can be scheduled simultaneously, then any subset of these queues can also be scheduled simultaneously. Note that this assumption holds in wireless networks.

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 schedule $\vec{\ell}(t)$ in slot t, for each k. We initially assume that the reward for each queue is a deterministic function of the schedule, and later generalize to allow the reward to randomly depend on the schedule (Section VI). We assume that $r_k(\vec{\ell}(t)) \le G_k$ and G_k is finite for each k. For example, in Figure 1, $r_1(\vec{\ell}_2) = 5, r_2(\vec{\ell}_2) = 0, r_1(\vec{\ell}_4) = 3, r_2(\vec{\ell}_4) = 1$.

Next, we present some important definitions.

Definition 1 (Scheduling Policy): A scheduling policy Δ decides the schedule $\vec{\ell}^{\Delta}(t)$ in each slot $t \ge 1$ such that $\vec{\ell}^{\Delta}(t) \in \mathcal{L}$.

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

Definition 2 (System Stability): The queueing system is said to be stable if the mean queue length for every k is finite.

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 Λ is the union of the stability regions of all scheduling policies. A scheduling policy whose stability region equals Λ 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* [1] showed that $\mathcal{C} \subseteq \Lambda \subseteq \overline{\mathcal{C}}$.

Definition 3 (Stabilizable Arrival Rate Vector): We denote the arrival rate vector $\vec{\lambda}$ as stabilizable if $\vec{\lambda} \in \mathcal{C}$.

Definition 4 (Throughput): For an arrival rate vector $\vec{\lambda}$, throughput under a scheduling policy Δ is the reward it receives per unit time. Mathematically,

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

Definition 5 (Throughput Optimality): A stable scheduling policy Δ is said to be throughput optimal if it maximizes the throughput among all the stable scheduling policies. We denote the throughput attained by such a policy for arrival rate vector vector $\vec{\lambda} \in \Lambda$ by $\Omega_{\max}(\vec{\lambda})$.

Definition 6 (ϵ-Throughput Optimality): A scheduling policy Δ is said to be ϵ-throughput optimal for some $\epsilon > 0$ if (a) it is stable, and (b) $\Omega^{\Delta}(\vec{\lambda}) \geq \Omega_{\max}(\vec{\lambda}) - \epsilon$.

In the following section, we examine the relation between throughput and stability for generalized constrained queueing systems. We show that in these systems 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 throughput subject to stability*. *Specifically, our goal is to design 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 scheduling 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 scheduling policy aims to maximize the throughput, then it would like to achieve the maximum possible reward per packet, and hence may like to wait and transmit only when the reward is high. 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 is wireless networks (Figure 1), we next demonstrate that a policy that maximizes the stability region does not maximize the throughput.

Example 1: Consider a system with two queues, namely S_1 and S_2. Let $\mathcal{L} = \left\{\vec{\ell}_1 = [0\ 0], \vec{\ell}_2 = [1\ 0], \vec{\ell}_3 = [0\ 1], \vec{\ell}_4 = [1\ 1]\right\}$. Let $r(\vec{\ell}_1) = 0$, $r(\vec{\ell}_2) = 5$, $r(\vec{\ell}_3) = 1$ and $r(\vec{\ell}_4) = 4$, where $r(\vec{\ell}_i) = r_1(\vec{\ell}_i) + r_2(\vec{\ell}_i)$. Let $\vec{\lambda} = (1/2 - \epsilon, 1/2 - \epsilon)$.

Now, consider a policy $\widehat{\Delta}$ that schedules each queue whenever it is non-empty. Clearly, $\widehat{\Delta}$ maximizes the stability region, and $\Omega^{\widehat{\Delta}}(\vec{\lambda}) \approx 10/4$. Now, consider a policy Δ' that serves S_1 when S_1 is non-empty, and serves S_2 if S_1 is empty and S_2 is non-empty. Note that Δ' is stable as $\lambda_1 + \lambda_2 = 1$, and $\Omega^{\Delta'} \approx 3$. Thus, $\Omega^{\widehat{\Delta}}(\vec{\lambda}) < \Omega^{\Delta}(\vec{\lambda})$.

Note that $\widehat{\Delta}$ in Example 1 always transmits the maximum number of packets per slot and also chooses the set of queues whose sum of queue lengths is the maximum. Tassiulas *et al* [1] showed that a policy that satisfies the latter property maximizes the stability region in arbitrary constrained queueing networks. But, Example 1 shows that $\widehat{\Delta}$ does not maximize the throughput. This is because $\widehat{\Delta}$ does not consider the reward structure in deciding the schedule.

Now, we consider two policies that seek to maximize the reward in a greedy fashion. The first policy Δ_1 maximizes the reward per packet, and the second policy Δ_2 maximizes the reward achieved in a slot. We show that both these policies do not stabilize the system even when the arrival rate vector is stabilizable.

Example 2: Consider the generalized constrained queueing system described in Example 1. Let $\vec{\lambda} = (3/4, 1/2)$. Clearly, $\vec{\lambda} \in \mathcal{C}$ and policy $\widehat{\Delta}$ in Example 1 stabilizes the system. Note that Δ_1 will never concurrently schedule 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 is not stable.

Note that Δ_1 does not take into account the arrival rates in its scheduling decisions. It maximizes the reward per packet at the cost of serving queues at rates smaller than their arrival rates and thereby compromises stability.

Example 3: Consider the system described in Example 1, with the difference that $r(\vec{\ell}_3) = 3$ and $r(\vec{\ell}_4) = 6$. Let $\vec{\lambda} = (1/4, 1/4)$. Note that for the reward structure considered here, scheduling decisions under Δ_2 are identical to that under $\widehat{\Delta}$. Thus, Δ_2 stabilizes the system, and $\Omega^{\Delta_2} = 15/8$. But, $\Omega^{\Delta'} = 2$ (Δ' has been described in Example 1). Thus, Δ_2 does not maximize the throughput.

The limitation of Δ_2 is that it myopically bases its scheduling decision in a slot solely on the aggregate reward in the slot. Thus, even when it is possible to wait and schedule queues in mutually disjoint slots and achieve a higher reward per packet, Δ_2 schedules 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 per packet or the reward in each slot may not maximize the throughput (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 then 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 $\vec{\ell}_i \in \mathcal{L}$ with probability (w.p.) w_i in every slot. When $\vec{\ell}_i$ is chosen, the k^{th} queue transmits a packet if $\vec{\ell}_{ik} = 1$ 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 $r_k(\vec{\ell}^{\Delta^*}(t)) \neq r_k(\vec{\ell}_i)$. We assume that $r_k(\vec{\ell}^{\Delta^*}(t)) \geq r_k(\vec{\ell}_i)$, i.e., if the k^{th} queue transmits a packet in t, it receives a reward of at least $r_k(\vec{\ell}_i)$ in t. We justify the 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 greater reward. If this is not the case, e.g., when the probability of success increases with increase in interference due to use of sophisticated decoding strategies, then if an empty queue is selected, it can transmit a signal 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 throughput consists of interesting topics for future research. The assumption can also be similarly justified for database systems.

The probability distribution $\vec{w} = [w_1 \ \cdots \ w_m]$ is computed using the following linear program 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) $\sum_i w_i \ell_{ik} = \lambda_k + \delta$ for every k
3) $\vec{\ell}_i \in \mathcal{L}$ for every i.

Constraint 1) ensures that $\vec{w}$ is a valid probability distribution. When $\delta > 0$, constraint 2) ensures that each queue is scheduled at a rate higher than the arrival rate in the queue. Thus, constraint 2) ensures stability. Constraint 3) ensures that only valid schedules are considered.

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 $\Delta^*(\vec{\lambda}, \delta)$.

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

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

We prove Theorem 1 in the appendix.

B. Queue length based optimal policy (Δ_O)

The policy Δ^* 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 provides the optimal throughput 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, Δ^* uses the knowledge of $\vec{\lambda}$ to ensure the above, whereas Δ_O ensures the above by using only the value of $\vec{Q}$.

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

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

where V is a constant. Note that the constraint $\ell_k \leq Q_k(t)$ 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 $V \geq 1$, Δ_O stabilizes the system. Moreover, for every $\epsilon > 0$, there exists $\widehat{V}$ such that for every $V \geq \widehat{V}$ Δ_O is ϵ-throughput optimal w.p. 1.

The above result implies that any stable off-line policy that takes transmission decisions based on the knowledge of past, present and future arrivals can not 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

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

$$W_1(\vec{Q}, \vec{\ell}) \stackrel{\text{def}}{=} \sum_{k=1}^{n} Q_k \ell_k.$$

Tassiulas *et al* have shown that in a constrained queueing system a policy ($\widehat{\Delta}$) that schedules $\vec{\ell}$ that maximizes $W_1(\vec{Q}, \vec{\ell})$ in every slot t stabilizes the system for every stabilizable $\vec{\lambda}$ [1]. This is because under $\widehat{\Delta}$ the queue length process has a negative drift when $\sum_{k=1}^{n} Q_k$ is sufficiently large for every stabilizable $\vec{\lambda}$. When $\sum_{k=1}^{n} Q_k >> V \sum_{k=1}^{n} G_k$, $W(\vec{Q}, \vec{\ell}) \approx W_1(\vec{Q}, \vec{\ell})$ for every $\vec{\ell} \in \mathcal{L}$, and therefore, Δ_O and $\widehat{\Delta}$ select similar schedules. Thus, intuitively, for every stabilizable $\vec{\lambda}$, the queue length process under Δ_O should also have a negative drift when $\sum_{k=1}^{n} Q_k$ is sufficiently large. Hence, Δ_O also stabilizes the system for any stabilizable $\vec{\lambda}$.

We have however shown that all stable policies does 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 the goal is to attain the maximum possible reward for every packet while maintaining stability. Now, when the queue lengths are small, the system remains stable even when the queues are served after long durations, and hence Δ_O schedules the queues only when they receive high rewards. On the other hand, when the queue lengths are large, Δ_O ensures stability by selecting the schedules so as to serve a large number of packets and also prioritizes the queues with higher queue lengths. Specifically, when $Q_k \leq V$, $Q_k - V\left(G_k - r_k(\vec{\ell})\right) \geq 0$ only if $r_k(\vec{\ell}) = G_k$. Then, since Δ_O maximizes $W(\vec{Q}, \vec{\ell})$, it will schedule the k^{th} queue only if the maximum possible reward is achievable. Now, if $Q_k \in \{V + 1, \ldots, 2V\}$, then $Q_k - V\left(G_k - r_k(\vec{\ell})\right) \geq 0$ only if $r_k(\vec{\ell}) \geq G_k - 1$. Thus, Δ_O will schedule the k^{th} queue only if the achievable reward is greater than or equal to $G_k - 1$. Similarly, if $Q_k \in \{(G_k - u)V + 1, \ldots, (G_k - u + 1)V\}$, then Δ_O will schedule the k^{th} queue only if the achievable reward is greater than or equal to u. Summarily, Δ_O dynamically adjusts the reward it achieves based on the queue length, and achieves large reward when the queue lengths are small and smaller rewards when the queue lengths are large. Thus, intuitively, Δ_O achieves the largest possible reward per packet constrained to stability, and hence it maximizes throughput constrained to stability.

Now, we prove Theorem 2 using a combination of optimization and lyapunov theories. Neely *et al.* [8], [19] 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, $\{\vec{Q}(t) = [Q_1(t) \ \cdots \ Q_n(t)]\}_{t \geq 1}$ is an irreducible, aperiodic and countable Markov chain. Now, consider the Lyapunov function

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

From (5) and (6), it follows that

$$\begin{aligned}
&f(\vec{Q}(t+1)) \\
&= \sum_{k=1}^{n} \Big[(Q_k(t))^2 + (\Lambda_k(t))^2 + (\ell_k^{\Delta_O}(t))^2 + 2Q_k(t)\Lambda_k(t) \\
&\quad -2Q_k(t)\ell_k^{\Delta_O}(t) - 2\Lambda_k(t)\ell_k^{\Delta_O}(t)\Big] \\
&\leq \sum_{k=1}^{n} \Big[(Q_k(t))^2 + (\Lambda_k(t))^2 + (\ell_k^{\Delta_O}(t))^2 \\
&\quad +2Q_k(t)\Lambda_k(t) - 2Q_k(t)\ell_k^{\Delta_O}(t)\Big] \\
&\leq n(A_{\max}^2 + 1) \\
&\quad + \sum_{k=1}^{n} \Big[(Q_k(t))^2 + 2Q_k(t)\Lambda_k(t) - 2Q_k(t)\ell_k^{\Delta_O}(t)\Big].
\end{aligned}$$

Let, $M \stackrel{\text{def}}{=} n(A_{\max}^2 + 1)$. Thus,

$$\begin{aligned}
&f(\vec{Q}(t+1)) - f(\vec{Q}(t)) \\
&\quad \leq M + \sum_{k=1}^{n} \Big[2Q_k(t)\Lambda_k(t) - 2Q_k(t)\ell_k^{\Delta_O}(t)\Big].
\end{aligned}$$

Now,

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

Now, since $\vec{\lambda}$ is stabilizable, we can obtain small enough δ such that $\Delta^*(\vec{\lambda}, \delta)$ is $\epsilon/2$-throughput optimal for any $\epsilon > 0$ (Theorem 1). We consider $\Delta^*(\vec{\lambda}, \delta)$ for such a δ.

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

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

Thus, from (7), for every $\vec{\lambda}$ and δ

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

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

$$\mathbb{E}\Big[\ell_k^{\Delta^*(\vec{\lambda},\delta)}(t) \mid \vec{Q}(t)\Big] = \sum_{i=1}^{m} \ell_{ik} w_i(\vec{\lambda}, \delta) = \lambda_k + \delta. \tag{9}$$

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 $Q_k(t) > 0$,

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

If $Q_k(t) = 0$,

$$\mathbb{E}\Big[\Big(G - r_k(\vec{\ell}^{\Delta^*(\vec{\lambda},\delta)}(t))\Big)\ell_k^{\Delta^*(\vec{\lambda},\delta)}(t) \mid \vec{Q}(t)\Big] = 0.$$

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,

$$\begin{aligned}
&\mathbb{E}\Big[\Big(G_k - r_k(\vec{\ell}^{\Delta^*(\vec{\lambda},\delta)}(t))\Big)\ell_k^{\Delta^*(\vec{\lambda},\delta)}(t) \mid \vec{Q}(t)\Big] \\
&\leq G_k(\lambda_k + \delta) - \sum_{i=1}^{m} w_i(\vec{\lambda}, \delta) r_k(\vec{\ell_i})\ell_{ik}.
\end{aligned} \tag{10}$$

From (4), (9), and (10), it follows that

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

Hence, from (8)

$$\begin{aligned}
&\mathbb{E}\left[f(\vec{Q}(t+1)) - f(\vec{Q}(t)) \mid \vec{Q}(t)\right] \\
&\leq \quad M - \sum_{k=1}^{n} 2\delta Q_k(t) + 2V\sum_{k=1}^{n} G_k(\lambda_k+\delta) - 2VU(\vec{\lambda},\delta) \\
&\qquad -2V\mathbb{E}\left[\sum_{k=1}^{n}[G_k - r_k(\vec{\ell}^{\Delta_O}(t))]\ell_k^{\Delta_O}(t) \mid \vec{Q}(t)\right]. \qquad (11)
\end{aligned}$$

1) Stability of Δ_O: From (11), 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

$$\begin{aligned}
&\mathbb{E}\left[f(\vec{Q}(t+1)) - f(\vec{Q}(t)) \mid \vec{Q}(t)\right] \\
&\leq \quad M - \sum_{k=1}^{n} 2\delta Q_k(t) + 2\sum_{k=1}^{n} VG_k(\lambda_k+\delta).
\end{aligned}$$

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

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

Thus, the queue length process $\{\vec{Q}(t)\}_{t\geq 0}$ is positive recurrent by Foster's Theorem (Theorem 2.2.3 in [20]). Hence, the expected queue length is bounded for every k. Thus, the system is stable under Δ_O.

2) ϵ-Throughput Optimality of Δ_O: Since the process $\{\vec{Q}(t)\}_{t\geq 0}$ under Δ_O is a positive recurrent markov chain, it has a stationary distribution. Taking expectation on both sides of (11) with respect to this stationary distribution, we obtain

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

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 $\{\vec{Q}(t)\}_{t\geq 0}$ under Δ_O is a positive recurrent markov chain, for every t

$$\mathbb{E}[\ell_k^{\Delta_O}(t)] = \lim_{u\to\infty} \frac{1}{u}\sum_{v=1}^{u} \ell_k^{\Delta_O}(v) = \lambda_k \text{ w.p. 1}, \qquad (13)$$

$$\begin{aligned}
&\text{and } \sum_{k=1}^{n} \mathbb{E}[r_k(\vec{\ell}^{\Delta_O}(t))\ell_k^{\Delta_O}(t)] \\
&\quad = \quad \lim_{u\to\infty}\frac{1}{u}\sum_{v=1}^{u}\sum_{k=1}^{n} r_k(\vec{\ell}^{\Delta_O}(v))\ell_k^{\Delta_O}(v) \text{ w.p. 1} \\
&\quad = \quad \Omega^{\Delta_O}(\vec{\lambda}) \quad \text{(from (1))}. \qquad (14)
\end{aligned}$$

Moreover, by stationarity it follows that

$$\mathbb{E}[f(\vec{Q}(t+1))] = \mathbb{E}[f(\vec{Q}(t))]. \qquad (15)$$

From (12), (13), (14) and (15), it follows that

$$\begin{aligned}
\Omega^{\Delta_O}(\vec{\lambda}) \quad &\geq \quad U(\vec{\lambda},\delta) - \delta\sum_{k=1}^{n} G_k - \frac{M}{2V} \\
&\geq \quad \Omega_{\max} - \frac{\epsilon}{2} - \frac{M}{2V} \text{ (from Theorem 1)} \quad (16) \\
&\geq \quad \Omega_{\max} - \epsilon \text{ if } V \geq M/\epsilon.
\end{aligned}$$

The result follows. ■

Finally, we comment on the role of the parameter V in determining the throughput of Δ_O. From (16), it can be seen that if $V < M/\epsilon = n(A_{\max}+1)/\epsilon$, 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. Computational Complexity of 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, computational complexity of $\Delta^*(\vec{\lambda},\delta)$ is exponential in n. 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 worst case complexity of obtaining the optimal schedule is also $O(2^n)$ unless some additional structure on the queueing system is assumed. In the following section, we propose two optimal scheduling policies whose computational complexity in every slot is polynomial in n.

V. Computationally Simple Optimal Policies

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

A. Inaccurate scheduling for maximizing throughput subject to attaining stability

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 [4], [10], [21]. Our contribution here is to generalize this notion to attain the goal of maximizing the throughput subject to attaining stability through simple computations. Let the current queue length vector be $\vec{Q}$. Consider a scheduling policy Δ that selects a schedule $\vec{\ell}^{\Delta}$ that satisfies the following inequality:

$$W(\vec{Q},\vec{\ell}^{\Delta_O}) - W(\vec{Q},\vec{\ell}^{\Delta}) \leq X(\vec{Q}), \qquad (17)$$

where $X(\vec{Q})$ is a random variable that depends on $\vec{Q}$.

Definition 7 (γ-Inaccurate Policy): A policy Δ_γ is called γ-inaccurate if it satisfies (17) and $\mathbb{E}[X(\vec{Q})] \leq \gamma$. Any schedule that satisfies (17) is called a γ-inaccurate schedule.

Note that if γ is large, then the number of γ-inaccurate schedules will be large and hence complexity of finding one such schedule may be small. *We show that all stable γ-inaccurate policies maximize the throughput subject to stability.*

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

Theorem 3: Let $\vec{\lambda}$ be any stabilizable arrival rate vector. Then, for every $\epsilon > 0$ and $\gamma < \infty$, there exists $\widehat{V}$ such that for every $V \geq \widehat{V}$,

1) if Δ_γ stabilizes the system, then Δ_γ is ϵ-throughput optimal w.p. 1, and
2) if $\mathbb{E}[X(\vec{Q}) \mid \vec{Y}(t) = \vec{Y}] \leq \gamma$ for every $\vec{Y}$, then Δ_γ stabilizes the system.

Now, we describe the intuition behind the above Theorem. For simplicity of explanation, we assume that $X(\vec{Q}) \leq \gamma$ for every $\vec{Q}$, and hence the condition in (2) of Theorem 3 holds. We first explain why γ-inaccurate policies maximize the stability region [4]. For large queue lengths, $W(\vec{Q}, \vec{\ell}^{\Delta_O}) >> \gamma$, and hence from (17), $W(\vec{Q}, \vec{\ell}^{\Delta_\gamma}) \approx W(\vec{Q}, \vec{\ell}^{\Delta_O})$. Thus, Δ_O and Δ_γ select similar schedules 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 sufficiently large queue lengths. Hence, Δ_γ stabilizes the system whenever $\vec{\lambda}$ is stabilizable.

The key difference between stability analysis and throughput analysis constrained to stability is that whereas for the former it suffices only to appropriately schedule when the queue lengths are large, but for the latter appropriate scheduling 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 $W(\vec{Q}, \vec{\ell}^{\Delta_O})$ may be smaller or comparable with γ. Then, (17) does not guarantee that $W(\vec{Q}, \vec{\ell}^{\Delta_\gamma}) \approx W(\vec{Q}, \vec{\ell}^{\Delta_O})$. Thus, the schedules selected in the two systems may be quite different in this case. 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 by proper choice of parameters the queue lengths and the schedules under Δ_O and Δ_γ are similar. The key idea here is to ensure that $\gamma << |W(\vec{Q}, \vec{\ell}^{\Delta_0})|$ for all $\vec{Q}$ such that $r_k(\vec{\ell}^{\Delta_O}) < G_k$ for some k for which $Q_k > 0$. Then, in the first slot, both systems have the same queue length vector, $\vec{Q}$. Now, due to (17), and since $\gamma << |W(\vec{Q}, \vec{\ell}^{\Delta_0})|$, $W(\vec{Q}, \vec{\ell}^{\Delta_0}) \approx W(\vec{Q}, \vec{\ell}^{\Delta_\gamma})$. Hence, $\vec{\ell}^{\Delta_\gamma} \approx \vec{\ell}^{\Delta_O}$. 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 schedules selected in each slot are similar in both systems. Thus, both policies attain similar throughput. We now describe how we can ensure that $\gamma << |W(\vec{Q}, \vec{\ell}^{\Delta_0})|$ for all $\vec{Q}$ such that $r_k(\vec{\ell}^{\Delta_O}) < G_k$ for some k for which $Q_k > 0$. When $||\vec{Q}||$ is large, then clearly the above holds. When $||\vec{Q}||$ is small, the above also holds if V is large. 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 an analysis similar to that for obtaining (7),

$$\begin{aligned}
&\mathbb{E}\left[f(\vec{Y}(t+1)) - f(\vec{Y}(t)) \mid \vec{Y}(t)\right] \\
&\leq M + \sum_{k=1}^{n} 2Q_k(t)\lambda_k - 2\mathbb{E}\left[W(\vec{Q}(t), \vec{\ell}^{\Delta_\gamma}(t)) \mid \vec{Y}(t)\right] \\
&\quad -2V\mathbb{E}\left[\sum_{k=1}^{n} [G_k - r_k(\vec{\ell}^{\Delta_\gamma}(t))]\ell_k^{\Delta_\gamma}(t) \mid \vec{Y}(t)\right]. \quad (18)
\end{aligned}$$

From (17) and (18), it follows that

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

Using arguments similar to those in the proof of (11) from (7), we can prove that

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

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

1) Proof for (1): Let Δ_γ stabilize the system. Then, since the process $\{\vec{Y}(t)\}_{t\geq 0}$ is a positive recurrent markov chain, it has a stationary distribution. Taking expectation on both sides of (19) with respect to this stationary distribution, we obtain

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

Since $\{\vec{Y}(t)\}_{t\geq 0}$ is a positive recurrent markov chain,

$$\mathbb{E}[\ell_k^{\Delta_\gamma}(t)] = \lim_{u\to\infty} \frac{1}{u} \sum_{v=1}^{u} \ell_k^{\Delta_\gamma}(v) = \lambda_k \text{ w.p. } 1, \quad (21)$$

$$\text{and } \sum_{k=1}^{n} \mathbb{E}[r_k(\vec{\ell}^{\Delta_\gamma}(t))\ell_k^{\Delta_\gamma}(t)]$$
$$= \lim_{u\to\infty} \frac{1}{u} \sum_{v=1}^{u}\sum_{k=1}^{n} r_k(\vec{\ell}^{\Delta_\gamma}(v))\ell_k^{\Delta_\gamma}(v) \text{ w.p. } 1$$
$$= \Omega^{\Delta_\gamma}(\vec{\lambda}) \quad \text{(from (1))}. \tag{22}$$

Moreover, by stationarity it follows that

$$\mathbb{E}[f(\vec{Y}(t+1))] = \mathbb{E}[f(\vec{Y}(t))]. \tag{23}$$

From (20), (21), (22) and (23), and since from Definition (7) $\mathbb{E}\left[X(\vec{Q}(t))\right] \leq \gamma$, it follows that

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

The result follows.

2) Proof for (2): Now, let $\mathbb{E}\left[X(\vec{Q}(t))|\vec{Y}(t)=\vec{Y}\right] \leq \gamma$ for all $\vec{Y}$. Thus, from (19), for every γ and $V \in [1,\infty)$,

$$\begin{aligned} &\mathbb{E}\left[f(\vec{Y}(t+1)) - f(\vec{Y}(t)) \mid \vec{Y}(t)\right] \\ &\leq M + 2\gamma - \sum_{k=1}^{n} 2\delta Q_k(t) + 2\sum_{k=1}^{n} VG_k(\lambda_k+\delta). \end{aligned}$$

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

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

Thus, the process $\{\vec{Y}(t)\}_{t\geq 0}$ is positive recurrent by Foster's Theorem (Theorem 2.2.3 in [20]). Hence, the expected queue length is bounded for every k. Thus, Δ_γ stabilizes the system. ■

The main challenge in computing γ-inaccurate schedules 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 comnplex. We circumvent this challenge by designing a computationally simple approach that obtains γ-inaccurate schedules 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 $[KT,(K+1)T-1]$ for $K \geq 0$. We consider a policy Δ_T that determines the schedule that attains the maximum in (3) at the beginning of each interval, i.e., in the slots KT for $K \geq 0$, and throughout the interval serves each queue in this schedule while it is non-empty.

The computational complexity of Δ_T is $O(2^n/T)$ in the amortized sense [22]. Thus, if we choose T to be sufficiently large ($\approx 2^n$), then Δ_T has $O(1)$ complexity in the amortized sense.

In the following lemma, we show that Δ_T is γ-inaccurate, and furthermore, $\mathbb{E}\left[X(\vec{Q}(t))|\vec{Y}(t)=\vec{Y}\right] \leq \gamma$ for all $\vec{Y}$ where $\gamma = nT(2A_{\max}+1)$.

Lemma 1: Let $\vec{Q}(t)$ be the queue length vector under Δ_T in t. Then, $W(\vec{Q}(t),\vec{\ell}^{\Delta_O}(t)) - W(\vec{Q}(t),\vec{\ell}^{\Delta_T}(t)) \leq nT(2A_{\max}+1)$.

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

$$\begin{aligned} \sum_{k=1}^{n} Q_k(t) &\leq \sum_{k=1}^{n} Q_k(KT) + \sum_{u=KT}^{t}\sum_{k=1}^{n} \Lambda_k(u) \\ &\leq \sum_{k=1}^{n} Q_k(KT) + nTA_{\max}. \end{aligned} \tag{24}$$

Similarly, from (5),

$$\begin{aligned} \sum_{k=1}^{n} Q_k(t) &\geq \sum_{k=1}^{n} Q_k(KT) - \sum_{u=KT}^{t}\sum_{k=1}^{n} \ell_k^{\Delta_T}(u) \\ &\geq \sum_{k=1}^{n} Q_k(KT) - nT. \end{aligned} \tag{25}$$

Now, from (4), (24) and (25), we obtain

$$\begin{aligned} &W(\vec{Q}(t),\vec{\ell}^{\Delta_O}(t)) - W(\vec{Q}(t),\vec{\ell}^{\Delta_T}(t)) \\ &\leq \sum_{k=1}^{n}[Q_k(KT) - V(G_k - r_k(\vec{\ell}^{\Delta_O}(t)))]\ell_k^{\Delta_O}(t) \\ &\quad -\sum_{k=1}^{n}[Q_k(KT) - V(G_k - r_k(\vec{\ell}^{\Delta_T}(t)))]\ell_k^{\Delta_T}(t) \quad (26) \\ &\quad +nT(A_{\max}+1) \\ &= W(\vec{Q}(KT),\vec{\ell}^{\Delta_O}(t)) - W(\vec{Q}(KT),\vec{\ell}^{\Delta_T}(t)) \\ &\quad +nT(A_{\max}+1) \end{aligned} \tag{27}$$

Now, from (3),

$$W(\vec{Q}(KT),\vec{\ell}^{\Delta_O}(t)) \leq W(\vec{Q}(KT),\vec{\ell}^{\Delta_O}(KT)). \tag{28}$$

Also, since the schedule selected by Δ_T change in the interval only if some queues empty during the period and then the change is to not serve them, $\vec{\ell}^{\Delta_T}(t) \leq \vec{\ell}^{\Delta_T}(KT)$ and if $\ell_k^{\Delta_T}(t) < \ell_k^{\Delta_T}(KT)$ then $Q_k(KT) \leq A_{\max}T$. Thus, $r_k\left(\vec{\ell}^{\Delta_T}(t)\right) \geq r_k\left(\vec{\ell}^{\Delta_T}(KT)\right)$ for all k. Hence,

$$W(\vec{Q}(KT),\vec{\ell}^{\Delta_T}(t)) \geq W(\vec{Q}(KT),\vec{\ell}^{\Delta_T}(KT)) - nA_{\max}T. \tag{29}$$

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

Thus, from Theorem 3, we conclude that when $\vec{\lambda}$ is stabilizable, Δ_T is ϵ-throughput optimal for every $\epsilon > 0$. We formally state this in the following theorem.

Theorem 4: Let $\vec{\lambda}$ be any stabilizable arrival rate vector. Then, for every $\epsilon > 0$, there exists $\widehat{V}$ such that for every $V \geq \widehat{V}$ the policy Δ_T is ϵ-throughput optimal w.p. 1.

The main challenge in using Δ_T is that it needs to periodically compute the optimal schedule. Since the complexity of each such computation is exponential in n, for large n, such computations may become infeasible. We next propose an optimal randomized policy whose computational complexity is $O(n)$ 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 [4]. The policy in [4] attains the maximum possible stability region in a constrained queueing network using linear complexity computations per slot. Our contribution here is to show that linear-complexity randomized policies can also maximize the throughput subject to stabilizing the system.

We now describe Δ_R. In every slot $t \geq 0$, Δ_R generates a schedule $\vec{\ell}(t)$ randomly as per a distribution $P_{\vec{Q}}(\cdot)$. In every slot $t \geq 1$, once a random schedule is generated, Δ_R obtains $\vec{\ell}^{\Delta_R}(t)$ iteratively as follows.

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

Thus, in any slot, Δ_R uses a new schedule only when it increases the value of $W(\cdot)$; otherwise it continues with the schedule used in the previous slot.

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 $\vec{Q}$, $P_{\vec{Q}}(\vec{\ell}^{\Delta_O}) \geq \mu$ for some $\mu > 0$.

Lemma 2: Let $\vec{\lambda}$ be a stabilizable arrival rate vector. Then Δ_R stabilizes the system.

We prove Lemma 2 in technical report [28]. Now, we show that Δ_R is $\frac{n}{\mu}(2A_{\max}+1)$-inaccurate.

Lemma 3: Let $\vec{Q}(t)$ be the queue length vector under Δ_R in t. Then, $W(\vec{Q}(t), \vec{\ell}^{\Delta_O}(t)) - W(\vec{Q}(t), \vec{\ell}^{\Delta_R}(t)) \leq X\left(\vec{Q}(t)\right)$, where $\mathbf{E}\left[X\left(\vec{Q}(t)\right)\right] \leq \frac{n}{\mu}(2A_{\max}+1)$.

Proof: Since $P_{\vec{Q}}(\vec{\ell}^{\Delta_O}) \geq \mu$ for every $\vec{Q}$, $\vec{\ell}^{\Delta_R}(t) = \vec{\ell}^{\Delta_O}(t)$ infinitely often w.p. 1. Let $\{\tau_K\}_{K\geq 1}$ be the slots in which $\vec{\ell}^{\Delta_R}(t) = \vec{\ell}^{\Delta_O}(t)$. Again, since $P_{\vec{Q}}(\vec{\ell}^{\Delta_O}) \geq \mu$ for every $\vec{Q}$, $\mathbb{E}[\tau_{K+1} - \tau_K] \leq 1/\mu$.

Without loss of generality, let $t \in [\tau_K, \tau_{K+1} - 1]$ for some K. Like in Lemma 1, we obtain

$$\begin{aligned} & W(\vec{Q}(t), \vec{\ell}^{\Delta_O}(t)) - W(\vec{Q}(t), \vec{\ell}^{\Delta_R}(t)) \\ \leq \;& n(\tau_{K+1} - \tau_K)(2A_{\max}+1). \end{aligned}$$

Thus, the result follows since $\mathbb{E}[\tau_{K+1} - \tau_K] \leq 1/\mu$. ∎

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 $\widehat{V}$ such that for every $V \geq \widehat{V}$ the policy Δ_R is ϵ-throughput optimal w.p. 1.

Now, Δ_R can be computed in $O(n)$ complexity if $\mu = 2^{-n}$. Each non-empty queue can be selected w.p. 1/2. If the resulting schedule is not in $\mathcal{L}$, then no packet is served. Thus, the optimal policy can be computed in linear complexity per slot.

D. Distributed Implementation of Δ_R and Δ_T

The concept of distributed scheduling has different definitions. 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 schedule 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 schedule at the beginning of each period, then chooses between the schedules selected in the current and previous periods based on the broadcasts it receives in the previous period, and finally uses the chosen schedule 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 well-understood, although some illuminating results have been obtained recently [6], [23], [24], [25], [26], [27]. 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.

VI. Generalizations

We have so far assumed that the reward received by the k^{th} queue in t is completely determined by the schedule 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 chosen schedule. Even in this generalized scenario, after suitable modifications, all the proposed policies, Δ^*, Δ_O, Δ_γ, Δ_T and Δ_R, continue to maximize throughput subject to attaining stability. Due to space constraints, we only present analytical results for Δ_O and Δ_R, and prove them in technical report [28].

The above generalization is relevant in context of wireless networks. In wireless networks, 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 schedule and the channel fading state between each sender-receiver pair. Typically, fading state in a channel is modeled by a finite positive recurrent Markov chain (MC) [29]. Now, we formally describe the generalization.

We consider a random process $\{\mathcal{S}(t)\}_{t\geq 1}$ which evolves as per a finite state, positive recurrent MC independent of the constrained queueing network. We assume that the scheduling policy knows $\mathcal{S}(t)$ at the beginning of slot t. Let the state space of the MC be $S_1, \ldots, S_z$. The reward achieved by the k^{th} queue is a r.v. whose distribution depends on the schedule and the process $\mathcal{S}(t)$. We denote the reward received by the k^{th} queue when $\vec{\ell}$ is scheduled and the process $\mathcal{S}(t)$ is in state S by $R_k(\vec{\ell}, S)$. 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.

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.

Theorem 6 (Generalization for Δ_O): Consider a stabilizable arrival rate vector $\vec{\lambda}$ and a scheduling policy $\widehat{\Delta}_O$ that chooses schedule $\vec{\ell}^{\widehat{\Delta}_O}(t)$ such that

$$\begin{aligned} &\vec{\ell}^{\widehat{\Delta}_O}(t) \\ = \ &\arg \max_{\substack{\vec{\ell} \in \mathcal{L}, \\ \vec{\ell}_k \leq Q_k(t)}} \left\{ \sum_{k=1}^{n} \left(Q_k(t) - V \left(G_k - r_k(\vec{\ell}, \mathcal{S}(t)) \right) \right) \ell_k \right\} \end{aligned}$$

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

Next, we describe how Δ_R can be generalized. We denote the generalized version of Δ_R as $\widehat{\Delta}_R$. Let

$$\widehat{W}(\vec{Q}, \vec{\ell}, S) \stackrel{\text{def}}{=} \sum_{k=1}^{n} \left(Q_k - V(G_k - r_k(\vec{\ell}, S)) \right) \ell_k$$

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

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

The policy $\widehat{\Delta}_R$ obtains the schedule$\{\vec{\ell}^{\widehat{\Delta}_R}(t)\}_{t \geq 1}$ as follows. In every slot $t \geq 0$, $\widehat{\Delta}_R$ generates a random schedule $\vec{\ell}(t)$. In every slot $t \geq 1$, after generating the random schedule, $\widehat{\Delta}_R$ obtains $\vec{\ell}^{\widehat{\Delta}_R}(t)$ using the following iterative algorithm:

$$\vec{\ell}^{\widehat{\Delta}_R}(t) = \begin{cases} \vec{\ell}(t) \text{ if } \widehat{W}\left(\vec{Q}(t), \vec{\ell}^{\widehat{\Delta}_R}\left(\tau(t, S(t))\right), S(t) \right) \\ \qquad < \widehat{W}\left(\vec{Q}(t), \vec{\ell}(t), S(t) \right), \\ \vec{\ell}^{\widehat{\Delta}_R}\left(\tau(t, S(t))\right) \quad \text{otherwise.} \end{cases}$$

First, we point out the key difference between Δ_R and $\widehat{\Delta}_R$. In each slot t, Δ_R compares $W(\cdot)$ for the randomly generated schedule $\vec{\ell}(t)$ with $W(\cdot)$ under the schedule used in slot $t-1$. Under $\widehat{\Delta}_R$, if $\mathcal{S}(t) = S$, then $\widehat{\Delta}_R$ compares $\widehat{W}(\cdot)$ for the randomly generated schedule $\vec{\ell}(t)$ with $\widehat{W}(\cdot)$ under the schedule used in slot $\tau(t, S)$. Thus, $\widehat{\Delta}_R$ effectively partitions time in z sets, each corresponding to a state of the process $\mathcal{S}(t)$, and executes Δ_R in each of these partitions. This is necessary as the reward in the generalized system also depends on the state of the process $\mathcal{S}(t)$. Hence, a schedule $\vec{\ell}$ that maximizes $\widehat{W}(\cdot)$ for some state S_j may not do so for some other state $S_{j'}$.

Theorem 7 (Generalization for Δ_R): Consider a stabilizable arrival rate vector $\vec{\lambda}$. Then, $\widehat{\Delta}_R$ stabilizes the system for every $V \geq 1$. Moreover, for every $\epsilon > 0$, there exists $\widehat{V}$ such that for every $V \geq \widehat{V}$, $\widehat{\Delta}_R$ is ϵ-throughput optimal.

Furthermore, all the results hold even when the arrival process $\vec{\Lambda} = [\Lambda_1 \ \cdots \Lambda_n]$ is stationary ergodic. In this case, arrivals in various queues can have arbitrary correlation.

We finally consider the case where the set of allowed schedules evolves randomly. Specifically, $\{\mathcal{L}(t)\}_{t \geq 1}$ evolves as per a finite state, positive recurrent MC independent of the constrained queueing network. We assume that the scheduling policy knows $\mathcal{L}(t)$ at the beginning of slot t. All policies can be generalized to this case as well. The generalizations can be considered in two different ways. In the first, the generalization of Δ_O is similar to $\widehat{\Delta}_O$ except that the schedules must be selected amongst $\mathcal{L}(t)$ instead of $\mathcal{L}$ in each slot t. The generalization of Δ_R is similar to $\widehat{\Delta}_R$ except that $\tau(t, S, L)$ must now be considered instead of $\tau(t, S)$, and $\tau(t, S, L) = 0$ if $(\mathcal{S}(u), \mathcal{L}(u)) \neq (S, L)$ and $\tau(t, S, L) = \max\{u < t \ : \ (\mathcal{S}(u), \mathcal{L}(u)) = (S, L)\}$. The stability regions are now different from that when $\mathcal{L}$ does not change. We refer to the interior of this new stability region as $\mathcal{C}'$. But, both Theorems 6 and 7 hold for all $\vec{\lambda} \in \mathcal{C}'$. Another approach is to consider this scenario in the framework of random rewards. Here, consider a new system in which the set of allowed schedules is the power-set of the set of queues, and the reward for serving each queue is 0 in a slot if the schedule is not in $\mathcal{L}(t)$. The system is otherwise similar to the actual system. Again, both Theorems 6 and 7 hold for all $\vec{\lambda} \in \mathcal{C}'$, and for $\widehat{\Delta}_O$ and $\widehat{\Delta}_R$ computed in the new system. These policies rarely select schedules in $\mathcal{L} \setminus \mathcal{L}(t)$ if $\vec{\lambda} \in \mathcal{C}'$.

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 [1]. Several generalizations of this basic result have been obtained in context of a diverse class of systems including wireless networks [3], [4], [5], [7], input queued switches [10], parallel processing systems [11] and manufacturing systems [30].

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 scheduled queues. An important performance goal in such networks is to maximize the reward per unit time or throughput subject to maximizing the stability region. 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 [31], [32], [33].

Recently, Neely has considered a queueing system in which in each slot different queues can be simultaneously served at different rates [8], [19]. 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 scheduled queues must be served at the same rate, but receive different rewards depending on the schedule. We maximize the total reward achieved per unit time subject to stability. Thus, in some sense, we study the dual of the problem studied in [8]. Concurrent with our work, Stolyar has investigated a similar problem, and has proposed optimal policies similar to $\Delta^*(\vec{\lambda}, \delta)$ and Δ_O [34]. 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.

Bonald *et al* also showed that a policy that maximizes the instantaneous throughput does not attain the system stability region [35]. But, they considered a wire-line network, whereas we consider constrained queueing systems which accommodate 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 tradeoff between fairness and stability, whereas we maximize the average throughput subject to stability.

We now describe the related work in optimizing other performance criteria in constrained queueing networks. Scheduling policies that provide certain delay characteristics [2], [36], [37], [38], [39], [40], [41] and those that minimize the packet loss when buffers are finite [17], [42] have been obtained in special cases. Another related branch of research has been to maximize network utility in systems with saturated queues (i.e., when each queue always has a packet to transmit) [43], [44], [45], [46], [47], [48], [49].

VIII. Conclusion

Summarily, we consider scheduling in generalized constrained queueing systems where rewards obtained for serving different queues depend on the schedule. We demonstrate that maximizing the throughput in such systems is not equivalent to maximizing the stability region. Subsequently, we design an adaptive scheduling policy that maximizes the throughput among all policies that stabilize the system, whenever the arrival rate vector is stabilizable. This policy does not require any knowledge of arrival statistics but is computationally complex. We subsequently design a framework for obtaining policies that attain the same goal using simple computations. All our results hold under mild assumptions like stationary, ergodic arrival processes, random rewards and random scheduling constraints.

References

[1] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, Dec 1992.

[2] ——, "Dynamic scheduling for minimum delay in tandem and parallel constrained queueing models," *Annals of Operation Research*, vol. 48, pp. 333–355, 1993.

[3] ——, "Dynamic server allocation to parallel queues with randomly varying connectivity," *IEEE Trans. Info. Theory*, vol. 30, no. 2, pp. 466–478, Mar 1993.

[4] L. Tassiulas, "Linear complexity algorithms for maximum throughput in radio networks and input queued switches," in *INFOCOM'98*, 1998, pp. 533–539.

[5] A. Eryilmaz and R. Shrikant, "Stable scheduling policies for fading wireless channels," *IEEE/ACM Trans. on Networking*, vol. 13, no. 2, pp. 411–424, 2005.

[6] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks," in *Proceedings of IEEE INFOCOM'05*, Miami, FL, Mar 2005.

[7] M. Neely and E. Modiano, "Capacity and delay tradeoffs for ad-hoc mobile networks," *IEEE Trans. on Information Theory*, vol. 51, no. 6, pp. 1917–1937, Jun 2005.

[8] M. Neely, "Energy optimal control for time varying wireless networks," in *IEEE INFOCOM'05*, Mar 2005.

[9] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," *IEEE Trans. on Communications*, vol. 47, no. 8, Aug 1999.

[10] I. Keslassy and N. McKeown, "Analysis of scheduling algorithms that provide 100% throughput in input-queued switches," in *The 39th Annual Allerton Conf. on Communication, Control and Computing*, Monticello, IL, Oct 2001.

[11] N. Bambos and J. Walrand, "Maximal throughput for stability of a class of parallel processing systems," in *the 29th IEEE Conference on Decision and Control*, Honolulu, Hawaii, Dec 1990, pp. 161–166.

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

[13] A. Stolyar, "On the stability of multiclass queueing networks: A relaxed sufficient condition via limiting fluid processes," in *Markov Processes and Related Fields*, 1995, pp. 491–512.

[14] S. Meyn, "Stability and optimization of queueing networks and their fluid models," in *the Summer Seminar on "The Mathematics of Stochastic Manufacturing Systems"*, Williamsburg, VA, Jun 1996.

[15] J. Morrison and P. Kumar, "New linear program performance bounds for queueing networks," *Journal of Optimization and Applications*, vol. 100, no. 3, pp. 575–597, Mar 1999, a Voulme in Honor of Prof. Y. C. Ho.

[16] S. Meyn, "Stability, performance evaluation, and optimization," in *MARKOV DECISION PROCESSES: Models, Methods, Directions, and Open Problems*. Kluwer Series on Operations Research, 2000.

[17] N. Bambos and G. Michailidis, "On parallel queueing with random server connectivity and routing constraints," *Probability in the Engineering and Information Sciences*, vol. 16, no. 2, pp. 185–203, 2002.

[18] ——, "Queueing and scheduling in random environment," *Adv. in Applied Probability*, vol. 36, no. 1, pp. 293–317, Mar 2004.

[19] M. Neely, "Dynamic power allocation and routing for satellite and wireless networks with time varying channels," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, Nov 2003.

[20] G. Fayolle, V. A. Malyshev, and M. V. Menshikov, *Topics in the Constructive Theory of Countable Markov Chains*. Cambridge University Press, 1995.

[21] S. Sarkar and L. Tassiulas, "A framework for routing and congestion control for multicast information flows," *IEEE Transactions on Information Theory*, vol. 48, no. 10, pp. 2690–2708, Oct 2002.

[22] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*. MIT Press, 2001.

[23] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput guarantees through maximal scheduling in multihop wireless networks," in *Proceedings of 43d Annual Allerton Conference on Communication, Control and Computing*, Allerton, Monticello, Illinois, September 28-30 2005.

[24] S. Sarkar, P. Chaporkar, and K. Kar, "Fairness and throughput guarantees through distributed scheduling in multihop wireless networks," in *Proceedings of 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, Boston, MA, April 2006.

[25] X. Wu and R. Srikant, "Regulated maximal matching: a distributed scheduling algorithm for multihop wireless networks with node-exclusive spectrum sharing," in *Proceedings of IEEE CDC-ECC'05*, Seville, Spain, Dec 2005.

[26] ——, "Bounds on the capacity region of multihop wireless networks under distributed greedy scheduling," in *Proceedings of IEEE INFOCOM*, Barcelona, Spain, April 2006.

[27] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu, "Joint asynchronous congestion control and distributed scheduling for multihop wireless networks," in *Proceedings of IEEE INFOCOM*, Barcelona, Spain, April 2006.

[28] P. Chaporkar and S. Sarkar, "Stable scheduling policies for maximizing throughput in generalized constrained queueing systems," University of Pennsylvania, http://www.seas.upenn.edu/∼prasanna/techrep, Tech. Rep., Jun 2005.

[29] Q. Zhang and S. Kassam, "Finite-state markov model for rayleigh fading channels," *IEEE Transactions on Communications*, vol. 47, pp. 1688–1692, Nov 1999.

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

[31] P. Chaporkar, A. Bhat, and S. Sarkar, "An adaptive transmission strategy to maximize throughput in mac layer wireless multicast," in *5th ACM Intl. Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Tokyo, Japan, May 2004, pp. 256–267.

[32] P. Chaporkar and S. Sarkar, "On-line optimal wireless multicast," in *2nd Intl. Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Nwtwoeks (WiOpt)*, Univ. of Cambridge, UK, Mar 2004, pp. 282–291.

[33] ——, "Wireless multicast: Theory and approaches," *IEEE Transactions on Information Theory*, vol. 51, no. 6, pp. 1954–1972, Jun 2005.

[34] A. Stolyar, "Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm," *Queueing Systems*, vol. 50, no. 6, pp. 401–457, August 2005.

[35] T. Bonald and L. Massoulie, "Impact of fairness on internet performance," in *Proceedings of ACM SIGMETRICS*, Cambridge, MA, June 2001.

[36] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, P. Whiting, and R. Vijayakumar, "Providing quality of service over a shared wireless link," *IEEE Communications Magazine*, vol. 39, no. 2, pp. 150–154, Feb 2001.

[37] S. Borst, "User-level performance of channel aware scheduling algorithms in wireless data networks," in *IEEE INFOCOM*, San Francisco, CA, Apr 2003.

[38] M. Goyal, A. Kumar, and V. Sharma, "Power constrined and delay optimal policies for scheduling transmission over a fading channel," in *IEEE INFOCOM*, San Francisco, CA, Apr 2003.

[39] S. Shakkottai and A. Stolyar, "Scheduling for multiple flows sharing a time-varying channel: The exponential rule," *American Mathematical Society Translations, Series 2, A volume in memory of F. Karpelevich*, 2002.

[40] A. Stolyar and K. Ramanan, "Largest weighted delay first scheduling: Large deviations andoptimality," *Annals of Applied Probability*, vol. 11, no. 1, pp. 1–48, 2001.

[41] A. Stolyar, S. Shakkottai, and R. Srikant, "Pathwise optimality of the exponential scheduling rule for wireless channels," *Advances in Applied Probability*, 2004.

[42] S. Sarkar, "Qptimum scheduling and memory management in input queued switches with finite buffer space," *IEEE Trans. on Information Theorey*, vol. 50, no. 12, pp. 3197–3220, Dec 2004.

[43] S. Borst and P. Whiting, "Dynamic rate control algorithms for hdr throughput optimization," in *IEEE INFOCOM*, Anchorage, Alaska, Apr 2001.

[44] S. Kulkarni and C. Rosenberg, "Opportunistic scheduling policies for wireless systems with short term fairness constraints," in *IEEE Globecom*, San Francisco, CA, Dec 2003.

[45] ——, "Opprotunistic scheduling in wireless systems with multiple interfaces and multiple constraints," in *6th ACM Intl. Workshop on Modeling, Analysis and Simulations of Wireless and Mobile Systems (MSWim)*, San Diego, CA, Sep 2003.

[46] X. Liu, E. Chong, and N. Shroff, "Opportunistic transmission scheduling with resource-sharing constraints in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 10, pp. 2053–2064, Dec 2001.

[47] ——, "A framework for opportunistic scheduling in wireless networks," *Computer Networks Journal (Elsevier)*, vol. 41, no. 4, pp. 451–474, 2003.

[48] Y. Liu and E. Knightly, "Opportunistic fair scheduling over multiple wireless channels," in *IEEE INFOCOM*, San Francisco, CA, Apr 2003.

[49] M. Neely, E. Modiano, and C.-P. Li, "Fairness and optimal stochastic control for heterogeneous networks," in *IEEE INFOCOM*, Miami, FL, Mar 2005.

APPENDIX

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 LP$(\vec{\lambda}, \delta)$ has a feasible solution. Thus, policy $\Delta^*(\vec{\lambda}, \delta)$ is well defined. In Lemma 5, we obtain an upper bound on the throughput of any stable scheduling policy. For stating these lemmas, we generalize the definitions of LP$(\vec{\lambda}, \delta)$ and $U(\vec{\lambda}, \delta)$. Let $\vec{\delta} = (\delta_1, \dots, \delta_n)$.

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) $\sum_i w_i \ell_{ik} = \lambda_k + \delta_k$ for every k
3) $\vec{\ell}_i \in \mathcal{L}$ for every i.

A. Supporting Lemmas

Lemma 4: Let $\vec{\lambda} \in \mathcal{C}$. Then, there exists a neighborhood $\mathcal{N}_0$ around $\vec{\lambda}$ such that 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 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 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 schedules $\vec{\ell}_i$ till time t, i.e.,

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

$$\begin{aligned} \text{Now, } \xi_i^{\Delta}(t) &\geq 0, \text{ for every } t \geq 1, && (30)\\ \frac{\sum_{i=1}^m \xi_i^{\Delta}(t)}{t} &= 1, \text{ for every } t \geq 1, && (31)\\ \lim_{t \to \infty} \frac{\sum_{i=1}^m \xi_i^{\Delta}(t) \ell_{ik}}{t} &= \lambda_k \text{ w.p. 1 for every } k. && (32) \end{aligned}$$

The last inequality holds since Δ is stable.

Consider any t, and let for each k,

$$c_{k,t} = \frac{\sum_{i=1}^m \xi_i^{\Delta}(t) \ell_{ik}}{t} - \lambda_k. \quad (33)$$

Let $\vec{c}_t = (c_{1,t}, \dots, c_{n,t})$. Since from (30), (31) and (33), $\frac{\xi_i^{\Delta}(t)}{t}$ is a feasible solution of LP$(\vec{\lambda}, \vec{c}_t)$,

$$\frac{\sum_{i=1}^m \sum_{k=1}^n \xi_i^{\Delta}(t) \ell_{ik} r_k(\vec{\ell}_i)}{t} \leq U(\vec{\lambda}, \vec{c}_t). \quad (34)$$

We will show that given any $\iota > 0$, there exists (a) a $\vec{\delta}$ such that $0 < \delta_i < \iota$ for all i, $1 \le i \le n$, and (b) $\hat{t}$ such that for every $t \ge \hat{t}$

$$\frac{\sum_{i=1}^{m}\sum_{k=1}^{n}\xi_i^{\Delta}(t)\ell_{ik}r_k(\vec{\ell}_i)}{t} \le \sup_{-\vec{\delta}<\vec{\delta}'<\vec{\delta}}\left\{U(\vec{\lambda},\vec{\delta}')\right\} \text{ w.p. } 1. \tag{35}$$

From Lemma 4, there exists a neighborhood $\mathcal{N}_0$ around $\vec{\lambda}$ such that 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, $1 \le i \le n$, 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 (32), there exists $\hat{t}$ such that for every k, $t \ge \hat{t}$, $-\delta_k < c_{k,t} < \delta_k$ w.p. 1. Thus, for every $t \ge \hat{t}$, $U(\vec{\lambda},\vec{c}_t) \le \sup_{-\vec{\delta}<\vec{\delta}'<\vec{\delta}}\left\{U(\vec{\lambda},\vec{\delta}')\right\}$. Now, (35) follows from (34).

Now, from (35) and by continuity of $U(\vec{\lambda},\vec{\delta})$,

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

Since Δ is an arbitrary stable policy, the lemma follows from the above inequality, (1) and Definition 5. ■

B. Proof of Theorem 1

Proof: From Lemma 4, there exists a neighborhood $\mathcal{N}_0$ around $\vec{\lambda}$ such that 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, 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

$$U(\vec{\lambda},\delta) - \delta\sum_{k=1}^{n}G_k \ge \Omega_{\max}(\vec{\lambda}) - \epsilon \text{ w.p. } 1. \tag{36}$$

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 service rate for the kth queue is greater than the arrival rate. Hence, the kth queue is stable. Let $\gamma_i(u) = 1$ if $\Delta^*(\vec{\lambda},\delta)$ selects $\vec{\ell}_i$ in slot u, and 0 otherwise. Thus,

$$\lim_{t\to\infty}\frac{1}{t}\sum_{u=1}^{t}\gamma_i(u) = w_i(\vec{\lambda},\delta) \text{ w.p. } 1, \tag{37}$$

$$\lim_{t\to\infty}\frac{1}{t}\sum_{u=1}^{t}\sum_{i=1}^{m}\gamma_i(u)\ell_{ik}\mathbf{1}_{\{Q_k(u)>0\}} = \lambda_k \text{ w.p. } 1, \tag{38}$$

$$\text{and } \lim_{t\to\infty}\frac{1}{t}\sum_{u=1}^{t}\sum_{i=1}^{m}\gamma_i(u)\ell_{ik} = \lambda_k+\delta \text{ w.p. } 1. \tag{39}$$

Relation (38) 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 k^{th} queue in $(0,t)$. Relation (39) follows from (37) and LP($\vec{\lambda},\delta$).

Now w.p. 1

$$\begin{aligned}
&\Omega^{\Delta(\vec{\lambda},\delta)}(\vec{\lambda})\\
&= \lim_{t\to\infty}\frac{\sum_{u=1}^{t}\sum_{k=1}^{n}\sum_{i=1}^{m}\gamma_i(u)\ell_{ik}r_k(\vec{\ell}_i)\mathbf{1}_{\{Q_k(u)>0\}}}{t}\\
&= \lim_{t\to\infty}\frac{\sum_{u=1}^{t}\sum_{k=1}^{n}\sum_{i=1}^{m}\gamma_i(u)\ell_{ik}r_k(\vec{\ell}_i)}{t}\\
&\quad - \lim_{t\to\infty}\frac{\sum_{u=1}^{t}\sum_{k=1}^{n}\sum_{i=1}^{m}\gamma_i(u)\ell_{ik}r_k(\vec{\ell}_i)\mathbf{1}_{\{Q_k(u)=0\}}}{t}\\
&\ge U(\vec{\lambda},\delta)\\
&\quad - \lim_{t\to\infty}\frac{\sum_{u=1}^{t}\sum_{k=1}^{n}G_k\sum_{i=1}^{m}\gamma_i(u)\ell_{ik}\mathbf{1}_{\{Q_k(u)=0\}}}{t}\\
&\quad \text{(from (37), LP}(\vec{\lambda},\delta)\text{ and since } r_k(\vec{\ell}_i) \le G_k \;\; \forall\, i)\\
&\ge U(\vec{\lambda},\delta) - \delta\sum_{k=1}^{n}G_k \text{ w.p. } 1 \text{ (from (38) and (39))}.
\end{aligned} \tag{40}$$

The result follows from (36) and (40). ■