A Queue-length-based Randomized Scheduler for Wireless Networks

Atilla Eryilmaz
ECE Department
Ohio State University
Columbus, OH 43210
Email: eryilmaz@ece.osu.edu

Asuman Ozdaglar
EECS Department
Massachusetts Institute of Technology
Cambridge, MA, 02139
Email: asuman@mit.edu

Peter Marbach
CS Department
University of Toronto
Toronto, CA M5S 3G4
Email: marbach@cs.toronto.edu

***Abstract*— Dynamic queue-length-based schedulers have been shown to maximize the throughput performance of data networks. However, for interference-limited networks, such as wireless and sensor networks, their implementation is computationally intractable. To resolve this, randomized policies with both centralized and distributed implementations have been studied in the literature. In this paper, we propose and analyze a new randomized policy, called the RANDWEIGHT Scheduling Policy, that reduces the complexity of the scheduling component while providing throughput guarantees for the system. For a bipartite network topology, we study the stability region of our policy and show that under sufficiently symmetric loads the policy is guaranteed to stabilize the network. The policy presented here is centralized, however it is amenable to distributed implementation as discussed in the paper. Moreover, rate control can be jointly implemented with this policy using recent methods proposed in the context of cross-layer network control.**

I. INTRODUCTION

There has been much recent interest in developing *throughput-optimal*[1] control policies for wireless networks. Most focus has been on control policies that use appropriately maintained queue occupancy levels, which were shown to guarantee throughput-optimality in general wireless networks (e.g. [21], [22], [16], [7], [19], [6]). However, the implementation of these policies is computationally complex, thus limiting their use in practice in large-scale networks.

This motivated a recent literature that studies low-complexity and distributed implementations for throughput-optimal scheduling policies. Tassiulas [20] proposed a randomized scheduling policy and showed that this policy is throughput optimal and has linear computational complexity for an input-queued switch. The evolutionary nature of this scheduler –where in each iteration a random schedule is picked randomly and compared with the the one at hand to choose the better of the two– allows it to achieve throughput-optimality [20] under reasonable conditions. Although the original policy of [20] required centralized coordination, several distributed implementations are developed in recent works (e.g. [14], [4], [17]).

The work of the first two authors were supported by the Control-Based Mobile Ad-Hoc Networking (CBMANET) Program under DARPA subcontract no. 060786.

[1] A control policy is *throughput-optimal* if it can stabilize any input arrival rate within the capacity (or stability) region of the network

In this paper, we propose and study a new randomized policy that has provably good throughput characteristics and lends itself to distributed implementation. One of the novelties of this policy is its implicit use of network topology and interference model information in its operation. Although the scheduler is defined for a general network model, we focus on an $N \times N$ switch for its analysis in this paper. We study the throughput region of our policy and show that our policy can stabilize those arrival rates in the capacity region that are sufficiently symmetric. Moreover, we propose a distributed implementation that is motivated by our policy and leave its analysis to a future work. In our companion paper [12], we extend this work to develop completely distributed asynchronous random access strategies for general networks.

The paper is organized as follows. In Section II, we introduce the system model and provide an overview of throughput-optimal control policies in wireless networks. In Section III, we provide a formal description of our randomized policy. In Section IV, we analyze the throughput characteristics of the randomized policy and present a distributed implementation for a switch. Section V contains our concluding remarks.

II. SYSTEM MODEL AND PROBLEM STATEMENT

We consider a fixed wireless network, which can be represented by a graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ denotes the set of nodes and $\mathcal{L}$ denotes the set of undirected links. We assume that nodes are perfectly synchronized[2] to a common clock and operate over time-slots. In each slot, a given link can be scheduled to be active or inactive. When a link is activated, a single *packet* can be transmitted on that link. Due the wireless nature of the communication medium, concurrent transmissions on any two links may interfere with each other. To capture this, we consider a *collision-based interference model* that limits the proximity of simultaneous transmissions. For example, the k^{th}*-order interference model* is one where when any two active links are separated by at least k other links. Given an interference model, we define any interference-free set of link activation vector as a *feasible schedule*, denoted by $\mathbf{s} = (s_l)_{l \in \mathcal{L}} \in \{0, 1\}^{|L|}$. We use $\mathcal{S}$ to denote the set of

[2] The assumption of perfect synchronization can be relaxed by extending the duration of time-slots to accommodate buffer zones between consecutive slots. The case of complete asynchronous operation is part of future work.

feasible schedules. Note that, for the 1^{st}-order interference model, $\mathcal{S}$ corresponds to the set of *matchings*[3]. Without loss of generality, we restrict our attention to *maximal* schedules, i.e., schedules for which there exist no other links that can be activated without interfering with an active link.

Regarding the traffic model, we assume a set, $\mathcal{F}$, of end-to-end flows with fixed routes competing for the network resources. Each flow $f \in \mathcal{F}$ is described by a source-destination node pair $(b(f), e(f))$; a mean rate $\lambda^{(f)}$; and a route $\mathcal{R}^{(f)}$ containing a set of links that connects $b(f)$ to $e(f)$. We assume, for simplicity, that the arrival process $\{a^{(f)}[t]\}$ for flow f is independently and identically distributed over time slots [4] with mean $\lambda^{(f)}$ and a finite second moment $A^{(f)}$. Note that, under the fixed route assumption, we can find the mean link rates from the mean flow rates as $\lambda_l = \sum_{f:l\in\mathcal{R}^{(f)}} \lambda^{(f)}$. The incoming traffic can be *inelastic* or *elastic*. Inelastic traffic corresponds to traffic with a fixed mean rate, while elastic traffic corresponds to traffic with adjustable mean rate. Therefore, in the case of elastic traffic $\lambda^{(f)}$ is an adjustable parameter, and a congestion controller is necessary to achieve fairness across competing flows, where the level fairness is measured through utility functions (see [18] for more details).

Numerous works have addressed the problems of scheduling-routing and congestion-control in wireless networks (e.g. [9], [5], [15], [19], [1], [2]). In particular, the seminal work of Tassiulas and Ephremides ([21]) tackled the problem of scheduling and routing in a general framework. They observed that properly maintained queue-length levels can be utilized to perform scheduling and routing decisions to achieve *throughput-optimality*[5]. In particular, they introduced the *back-pressure (BP) policy*. In a separate line of work, the seminal work of Kelly et al. ([8]) resolved the problem of flow control in wireline networks through an optimization formulation, which are then extended in [10], [18]. It has recently been shown that these two frameworks can be incorporated to obtain a fair and efficient scheduling-routing and congestion-control policy (e.g. [3], [9], [6], [15], [19]).

The focus of this work is the scheduling component of the queue-length-based policies. To that end, we describe a queueing architecture that is described in [3]. We assume that each node maintains a queue for each flow that traverses it. We use $q_n^{(f)}[t]$ to denote the length of the queue at node n that contains flow f packets, at the beginning of slot t. We further let $s_l^{(f)}[t]$ denote the number of flow f packets that are scheduled for transmission over link l in slot t. Then, the evolution of $q_n^{(f)}[t]$ satisfies

$$q_n^{(f)}[t+1] \quad \leq \quad \left(q_n^{(f)}[t] - \sum_{m:(n,m)\in\mathcal{R}^{(f)}} s_{(n,m)}^{(f)}[t] \right)^+ + \sum_{k:(k,n)\in\mathcal{R}^{(f)}} s_{(k,n)}^{(f)}[t] + a_n^{(f)}[t]\mathcal{I}_{n=b(f)},$$

[3]A matching of a graph is a set of links where no two links share a node.
[4]This assumption is not restrictive, and can be eliminated easily (see [...]).
[5]A policy is called throughput-optimal if it can stably support any flow rate vector that is stably supportable by any other policy.

where $(y)^+ = max(0, y)$ and $\mathcal{I}_\mathcal{A}$ is the indicator for event $\mathcal{A}$. We use $\mathbf{q}[t]$ to denote the vector of queue-length levels at slot t. These queue-length levels are then used to obtain link weights. While there could be numerous choices for the queue-length to link weight transformation, it has been observed that the following transformation, referred to as *differential backlog*, yields attractive throughput characteristics(e.g. [21], [16], [7]):

$$w_{(n,m)}(\mathbf{q}) := \max_{f\in\mathcal{F}} \left| q_n^{(f)} - q_m^{(f)} \right|, \qquad \forall (n,m) \in \mathcal{L}. \qquad (1)$$

Notice that this transformation is locally computable since it only requires neighbor's queue-length information. Next, we describe the MAXWEIGHT scheduler that is implemented by the BP policy.

*Definition 1 (*MAXWEIGHT *Scheduler):* In slot t, the MAXWEIGHT Scheduler serves the schedule $\mathbf{s}^\star[t] \in \mathcal{S}$ that satisfies

$$\mathbf{s}^\star[t] \quad \in \quad \arg\max_{\mathbf{s}\in\mathcal{S}} \langle \mathbf{w}[t], \mathbf{s} \rangle, \qquad (2)$$

where $\langle \mathbf{w}[t], \mathbf{s} \rangle := \sum_{l\in\mathcal{L}} w_l[t] s_l$. ◇

The operation (2) of picking the maximum weight schedule in every time slot is generally a high-complexity operation. Next, we provide a partial overview of the literature on low-complexity, distributed schedulers in this context.

Overview of the Distributed Implementations

There have been a large number of algorithms proposed in the literature to obtain low-complexity scheduling/routing policies, while providing throughput guarantees. In this subsection, we aim to give an partial overview of those policies while stressing how they contribute to our intuition in proposing our policy.

A class of policies, called PICK & COMPARE policies, exist in the literature that are based on the algorithm first introduced in [20]. In a given slot t, this policy simply picks a random schedule, say $\tilde{\mathbf{s}}$, and compares the weights of the schedules $\mathbf{s}[t-1]$ and $\tilde{\mathbf{s}}$, i.e. $\langle \mathbf{w}[t], \mathbf{s}[t-1] \rangle \gtreqless \langle \mathbf{w}[t], \tilde{\mathbf{s}} \rangle$, and sets $\mathbf{s}[t]$ to the schedule that yields the greater weight. This evolutionary policy is shown to be throughput-optimal in the same work. While centralized in its original form, subsequent works built on this approach to provide distributed policies (e.g. [14], [4], [17]) with varying complexity characteristics.

In a different line of work, several policies with attractive distributive properties are proposed (e.g. [9], [23], [11]), while sacrificing from throughput optimality. These policies can reduce the queue-length information sharing to one-hop neighborhood, thus rendering them easily implementable. However, they are guarantees to support only a fraction of the capacity region. The precise fraction varies based on the interference model, and the topology of the network, but typically no more than half the region can be guaranteed.

Our goal in this work is to provide a scheduling algorithm with provably good throughput characteristics that are amenable to distributed implementation. To that end, we first propose a randomized algorithm and study its throughput characteristics for a bipartite graph. Then, we discuss how it can be implemented in a distributed fashion.

III. Description of the Randomized Policy

In this section, we propose a randomized scheduling policy, called the RandWeight Scheduler, that will be analyzed in the following sections. The policy has three key characteristics: it exploits queue-length information, which provides attractive throughput properties; it implicitly takes advantage of the topology information and the interference model, which are assumed to be fixed and known; and its randomized nature allows for distributed implementation.

Before we give the formal description of the scheduler, we provide a few definitions that will simplify our notation: for a given queue-length vector, $\mathbf{q}$,

- let the *total weight of schedule* $\mathbf{s} \in \mathcal{S}$ be defined as

$$W_{\mathbf{s}}(\mathbf{q}) \quad := \quad \sum_{l \in \mathbf{s}} w_l(\mathbf{q}). \tag{3}$$

- let the *total weight of a link* $l \in \mathcal{L}$ be defined as

$$W_l(\mathbf{q}) \quad := \quad \sum_{\mathbf{s}: l \in \mathbf{s}} W_{\mathbf{s}}(\mathbf{q}). \tag{4}$$

- let the *total weight of the network* be defined as

$$W(\mathbf{q}) \quad := \quad \sum_{\mathbf{s} \in \mathcal{S}} W_{\mathbf{s}}(\mathbf{q}). \tag{5}$$

*Definition 2 (*RandWeight *Scheduler):* In slot t, the RandWeight Scheduler picks a schedule $\tilde{\mathbf{s}}[t]$ such that :

$$\mathbb{P}\left(\tilde{\mathbf{s}}[t] = \mathbf{s}\right) \quad = \quad \frac{W_{\mathbf{s}}(\mathbf{q}[t])}{W(\mathbf{q}[t])} \qquad \text{for all } \mathbf{s} \in \mathcal{S}. \tag{6}$$

◇

Note that the RandWeight Scheduler picks a schedule with a probability that is proportional to its weight. Thus, those schedules with a high weight are more likely to be scheduled. Yet, it is still possible for low-weight schedules to be chosen for activation. When compared to (2), it can be seen that (6) relaxes the decision criterion. Also note that, in its current form, RandWeight Scheduler still requires global queue-length information. After we analyze the throughput characteristics of this policy, we will investigate ways in which it can be implemented in a distributed manner.

IV. Analysis and Discussions

In this section, we investigate the throughput characteristics of the RandWeight Scheduler described in Definition 2. Our goal is to identify the extend to which the non-optimal and randomized nature of the proposed policy reduces the stably supportable rates from the whole stability region. Although the policy is applicable to the general network model of Section II, we analyze the case of an $N \times N$ switch represented as a bipartite graph since the stability region of such graphs leads to tractable formulations. To that end, we introduce the model for an $N \times N$ switch and a few relevant definitions in the following section.

A. System Model for an $N \times N$ Switch

The graph associated with an $N \times N$ switch is a bipartite graph with N input and N output ports with N^2 flows, one for each input-output pair. We use $\mathcal{I} = \{1, ..., N\}$ and $\mathcal{O} = \{N+1, ..., 2N\}$ to denote the indices of the input and output ports, respectively. As before, we assume a time slotted system where each slot can accommodate a single packet transmission. We assume a given link level load λ, i.e., associated with every link $(i,j) \in \mathcal{L}$, there is an arrival process $a_{(i,j)}[t]$ with mean $\lambda_{(i,j)}$ and a finite second moment, A. We consider the 1^{st}-order interference model, i.e., at any given slot, at most one link incident to any given port can be active. Then, $\mathcal{S}$ corresponds to the set of maximal matchings. The capacity (stability) region of an $N \times N$ switch is defined as

Definition 3 (Capacity (Stability) Region of an $N \times N$ switch): The *capacity region of an $N \times N$ switch* is the set of arrival rates given by

$$\mathcal{C} \quad = \quad \left\{ \lambda \in \mathbb{R}_+^{N^2} : \sum_{i \in \mathcal{I}} \lambda_{(i,j)} < 1, \; \forall \; j \in \mathcal{O}, \right.$$
$$\left. \text{and} \qquad \sum_{j \in \mathcal{O}} \lambda_{(i,j)} < 1, \; \forall \; i \in \mathcal{I} \right\}.$$

◇

A queue is maintained for each link $(i,j) \in \mathcal{I} \times \mathcal{O}$. We use $q_{(i,j)}[t]$ to denote the length of queue associated with link (i,j) at the beginning of time slot t. We use $\mathbf{s}[t] := [s_{(i,j)}[t]]_{(i,j) \in \mathcal{I} \times \mathcal{O}}$ to denote the link activation vector at slot t. Then, the evolution of each queue between time slots is given by

$$\begin{aligned} q_{(i,j)}[t+1] \quad &= \quad \left(q_{(i,j)}[t] - s_{(i,j)}[t]\right)^+ + a_{(i,j)}[t] \\ &= q_{(i,j)}[t] - s_{(i,j)}[t] + u_{(i,j)}[t] + a_{(i,j)}[t], \end{aligned} \tag{7}$$

where $u_{(i,j)}[t]$ denotes the unused service that is offered to Queue-(i,j) in slot t.

Note that for this topology, link weight of link (i,j) defined in (1) is given by $w_{(i,j)}(\mathbf{q}) = q_{(i,j)}$ for all $(i,j) \in \mathcal{I} \times \mathcal{O}$. Also, note that $\mathbf{q}[t]$ forms a Markov Chain, and we say that Queue-(i,j) is *stable* if $\mathbb{E}[q_{(i,j)}[\infty]] < \infty$ where $\mathbf{q}[\infty]$ denotes the stationary distribution of the Markov Chain. The network is said to be stable if all its queues are stable.

B. Throughput Analysis of RandWeight

In this section, we specify the region of stabilizable mean rates under the RandWeight Scheduler for an $N \times N$ switch and prove its stabilizing properties. We start with two lemmas that will be used in the proof of the theorem.

Lemma 1: Under the RandWeight Scheduler, for a given queue-length vector $\mathbf{w}$, the average rate provided to link (i,j) is given by

$$\mathbb{E}[\tilde{s}_{(i,j)}[t] \mid \mathbf{q}[t]] \quad = \quad \frac{W_{(i,j)}(\mathbf{q}[t])}{W(\mathbf{q}[t])} \tag{8}$$

Proof: By the definition of RANDWEIGHT

$$\mathbb{E}[\tilde{s}_{(i,j)}[t] \mid \mathbf{q}[t]] = \sum_{\mathbf{s}\in\mathcal{S}:(i,j)\in\mathbf{s}} \frac{W_{\mathbf{s}}(\mathbf{q}[t])}{W(\mathbf{q}[t])} = \frac{W_{(i,j)}(\mathbf{q}[t])}{W(\mathbf{q}[t])}$$

■

Lemma 2: $\displaystyle\sum_{(i,j)\in\mathcal{I}\times\mathcal{O}} q_{(i,j)} = \frac{1}{(N-1)!}W(\mathbf{q})$.

Proof:

$$\begin{aligned} W(\mathbf{q}) &= \sum_{\mathbf{s}\in\mathcal{S}}\sum_{(i,j)\in\mathbf{s}} q_{(i,j)} = \sum_{(i,j)} q_{(i,j)} \sum_{\mathbf{s}:(i,j)\in\mathbf{s}} 1 \\ &= (N-1)!\sum_{(i,j)} q_{(i,j)}. \end{aligned}$$

where the last step follows from the fact that there exists $(N-1)!$ maximal matchings that contain a given link. ■

Next theorem identifies a region of rates supportable by the RANDWEIGHT Scheduler. It states that the policy achieves that portion of the stability region that yields sufficiently symmetric arrival rates.

Theorem 1: The RANDWEIGHT Policy stabilizes the network for any arrival rate $\lambda \in \mathcal{C}$ satisfying $\lambda_{(i,j)} \in [0, 1/N)$.

Proof: Our goal is to show that for any rate satisfying the conditions of the theorem, the queues evolve to the origin starting from any initial condition. To prove this theorem, we use Foster's criterion ([13]): if a Markov Chain $X[t]$ satisfies

$$\begin{aligned} &\mathbb{E}[f(X[t+1]) - f(X[t]) \mid X[t] = X] \\ &\qquad\qquad \le -\delta g(X)\mathcal{I}_{X\in\mathcal{A}} + B\mathcal{I}_{X\in\mathcal{A}^c}, \end{aligned}$$

for some positive δ, bounded value B, bounded set $\mathcal{A}$ and non-negative functions $f(\cdot)$ and $g(\cdot)$, then $\mathbb{E}[g(X[\infty])] < \infty$, where $X[\infty]$ is the stationary distribution of $X[t]$.

Let us define the Lyapunov function

$$V(\mathbf{q}) = \frac{1}{2}\sum_{(i,j)\in\mathcal{I}\times\mathcal{O}} q^2_{(i,j)}.$$

Then, we consider the mean drift of $V(\cdot)$ at time t for a given queue-length state $\mathbf{q}$.

$$\begin{aligned} \Delta V(\mathbf{q}) &:= \mathbb{E}[V(\mathbf{q}[t+1]) - V(\mathbf{q}[t]) \mid \mathbf{q}[t] = \mathbf{q}] \\ &= \frac{1}{2}\sum_{(i,j)}\left(\mathbb{E}[q^2_{(i,j)}[t+1] \mid \mathbf{q}[t]=\mathbf{q}] - q^2_{(i,j)}\right) \\ &= \frac{1}{2}\sum_{(i,j)}\Big(\mathbb{E}\,\big[(q_{(i,j)}[t] + a_{(i,j)}[t] - s_{(i,j)}[t])^2 \\ &\quad +2(q_{(i,j)}[t] + a_{(i,j)}[t] - s_{(i,j)}[t])u_{(i,j)}[t] \qquad (9) \\ &\quad +\ u^2_{(i,j)}[t] \mid \mathbf{q}[t]=\mathbf{q}\big] - q^2_{(i,j)}\Big) \end{aligned}$$

Since $u_{(i,j)}[t]$ is nonzero only if $q_{(i,j)}[t] < s_{(i,j)}[t]$, we can upper-bound (9) with $2\lambda_{(i,j)}\mathbb{E}[u_{(i,j)}[t] \mid \mathbf{q}[t]=\mathbf{q}]$. Also, using the fact that $u_{(i,j)}[t] \le 1$ for all (i,j) and t, we can upper-bound the mean drift as

$$\begin{aligned} &\Delta V(\mathbf{q}) \\ &\le \frac{1}{2}\sum_{(i,j)}\Big(\mathbb{E}\,\big[(q_{(i,j)}[t] + a_{(i,j)}[t] - s_{(i,j)}[t])^2 \mid \mathbf{q}[t]=\mathbf{q}\big] \\ &\qquad\qquad -q^2_{(i,j)}\Big) + B_1 \\ &= \sum_{(i,j)}\Big(q_{(i,j)}\mathbb{E}[a_{(i,j)}[t] - s_{(i,j)}[t] \mid \mathbf{q}[t]=\mathbf{q}] \\ &\qquad\qquad +\frac{\mathbb{E}[(a_{(i,j)}[t] - s_{(i,j)}[t])^2 \mid \mathbf{q}[t]=\mathbf{q}]}{2}\Big) + B_1 \\ &\le \sum_{(i,j)}\left[q_{(i,j)}\left(\lambda_{(i,j)}[t] - \frac{W_{(i,j)}(\mathbf{q})}{W(\mathbf{q})}\right)\right] + B_1 + B_2, \end{aligned}$$

where $B_1 = N^2 + 2\sum_{(i,j)}\lambda_{(i,j)} \le N^2 + 2N$, $B_2 \le (N^2(A+1) + 2N)$ since $\mathbb{E}[a^2_{(i,j)}[t]] \le A$, and the last inequality uses Lemma 1. Let us define $\tilde{\lambda} = (\tilde{\lambda}_{(i,j)})_{(i,j)} := (\lambda_{(i,j)} + \epsilon)_{(i,j)}$, where $\epsilon \in \left(0, \min_{(i,j)\in\mathcal{I}\times\mathcal{O}}\left(\frac{1}{N} - \lambda_{(i,j)}\right)\right)$. Such an ϵ exists due to our assumption on λ. Note that $\tilde{\lambda}_{(i,j)} < 1/N$ for all $(i,j) \in \mathcal{I}\times\mathcal{O}$ under this choice of ϵ. Then, we can re-write the last upper-bound as

$$\begin{aligned} \Delta V(\mathbf{q}) &\le -\epsilon\sum_{(i,j)} q_{(i,j)} + B_1 + B_2 \\ &\quad + \sum_{(i,j)}\left[q_{(i,j)}\left(\tilde{\lambda}_{(i,j)}[t] - \frac{W_{(i,j)}(\mathbf{q})}{W(\mathbf{q})}\right)\right] \\ &\le -\epsilon\sum_{(i,j)} q_{(i,j)} + B_1 + B_2 \qquad (10) \\ &\quad + \sum_{(i,j)}\left[q_{(i,j)}\left(\frac{1}{N} - \frac{W_{(i,j)}(\mathbf{q})}{W(\mathbf{q})}\right)\right]. \qquad (11) \end{aligned}$$

Next, we focus on (11) and show that it is non-positive.

$$\begin{aligned} (11) &= \frac{1}{N}\sum_{(i,j)} q_{(i,j)} + \frac{1}{W(\mathbf{q})}\sum_{(i,j)}\left[q_{(i,j)}\sum_{\{\mathbf{s}:(i,j)\in\mathbf{s}\}} W_{\mathbf{s}}(\mathbf{q})\right] \\ &= \frac{1}{N!}W(\mathbf{q}) - \frac{1}{W(\mathbf{q})}\sum_{\mathbf{s}\in\mathcal{S}}\left[W_{\mathbf{s}}(\mathbf{q})\sum_{(i,j)\in\mathbf{s}} q_{(i,j)}\right] \qquad (12) \\ &= \frac{1}{N!}W(\mathbf{q}) - \frac{1}{W(\mathbf{q})}\sum_{\mathbf{s}\in\mathcal{S}} W^2_{\mathbf{s}}(\mathbf{q}), \qquad (13) \end{aligned}$$

where (12) follows from the use of Lemma 2 on the first sum, and a reordering of the second and third sums.

Finally, we utilize Jensen's Inequality:

$$\left(\frac{1}{N!}W(\mathbf{q})\right)^2 = \left(\frac{1}{N!}\sum_{\mathbf{s}\in\mathcal{S}} W_{\mathbf{s}}(\mathbf{q})\right)^2 \le \frac{1}{N!}\sum_{\mathbf{s}\in\mathcal{S}} W^2_{\mathbf{s}}(\mathbf{q})$$

in (13) to complete the proof of our claim that $(11) \le 0$.

After substituting this upper bound in (10)-(11), we have

$$\Delta V(\mathbf{q}) \le -\epsilon\sum_{(i,j)} q_{(i,j)} + B_1 + B_2$$

which satisfies Foster's criterion, and hence we have $\mathbb{E}[\sum_{(i,j)} q_{(i,j)}[\infty]] < \infty$, and the proof is complete. ∎

Noting that the maximum symmetric rate achievable by a switch is given by $\lambda_{(i,j)} = 1/N - \delta$ for all (i,j) with $\delta > 0$ arbitrarily small, Theorem 1 proves that the RANDWEIGHT scheduler can achieve the largest possible symmetric rate. In contrast to other schedulers in this context that support half of the stability region for all rates, this scheduler yields optimal throughput performance under symmetric rates.

C. On the Distributed Implementation

In this section, we discuss how the RANDWEIGHT scheduler may be implemented in a distributed fashion. To that end, we recall from Lemma 1, that link (i,j) needs to be activated with a probability of $W_{(i,j)}(\mathbf{q}[t])/W(\mathbf{q}[t])$. Finding this probability could be straight-forward for some cases, as is in the case of an $N \times N$ switch provided next.

Lemma 3: For an $N \times N$ switch, the probability with which a given link, say $(i,j) \in \mathcal{I} \times \mathcal{O}$, is active under the RANDWEIGHT scheduler is given by

$$\mathbb{P}[\tilde{s}_{(i,j)}[t] = 1 \mid \mathbf{q}[t]] = \frac{\left[q_{(i,j)}[t] + \sum_{\{(n,m): n \neq i, m \neq j\}} q_{(n,m)}[t] \right]}{(N-1) \sum_{(n,m)} q_{(n,m)}[t]}$$

Proof: Recall that $W(\mathbf{q}[t])$ is given in Lemma 2. Next, we must find $W_{(i,j)}(\mathbf{q}[t])$, which is the sum of the weights of all maximal matchings that contain (i,j). There exists $(N-1)!$ such matchings since there are $(N-1)$ input and $(N-1)$ output ports that can be freely matched, once ports i and j are fixed. Lemma 2 applies to the resulting $(N-1) \times (N-1)$ switch, which results in

$$W_{(i,j)}(\mathbf{q}[t]) = (N-2)! \left[q_{(i,j)} + \sum_{\{(n,m): n \neq i, m \neq j\}} q_{(n,m)}[t] \right].$$

Substituting this expression and the statement of Lemma 2 in (8) completes the proof. ∎

Lemma 3 yields an easily computable expression for link activation probabilities by using the queue-length values available at the input. Then, the individual links can be activated independently according to these probabilities in a distributed fashion. Although such independent operation of nodes may not achieve the same average rates as the RANDWEIGHT policy, it is of interest to study the performance of the resulting distributed implementation. We leave this for future work.

V. CONCLUSIONS

In this paper, we proposed a randomized queue-length-based scheduling policy that is applicable to general wireless networks, and studied its throughput properties. For a bipartite graph and 1^{st}-order interference model, we showed that the policy can achieve rates that can be arbitrarily close to the boundary of the capacity region as long as they are sufficiently symmetric. As a distributed implementation, we proposed a random access strategy where the transmission probabilities are explicitly characterized as a function of queue-lengths. We leave throughput characteristics and distributed implementations of this policy for general network topologies for future research. Our future plans also include investigating the rate of convergence and delay characteristics of this policy.

REFERENCES

[1] P. Chaporkar, K. Kar, and S. Sarkar. Throughput guarantees through maximal scheduling in wireless networks. In *Proceedings of the Allerton Conference on Control, Communications and Computing*, 2005.

[2] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle. Jointly optimal congestion control, routing, and scheduling for wireless ad hoc networks. In *Proceedings of IEEE Infocom*, Barcelona, Spain, April 2006.

[3] A. Eryilmaz. Efficient and fair scheduling for wireless networks. PhD thesis, University of Illinois at Urbana, Champaign, August 2005.

[4] A. Eryilmaz, A. Ozdaglar, and E. Modiano. Polynomial complexity algorithms for full utilization of multi-hop wireless networks. 2007. Proceedings of IEEE Infocom.

[5] A. Eryilmaz and R. Srikant. Fair resource allocation in wireless networks using queue-length based scheduling and congestion control. In *Proceedings of IEEE Infocom*, volume 3, pages 1794–1803, 2005.

[6] A. Eryilmaz and R. Srikant. Resource allocation of multi-hop wireless networks. In *Proceedings of International Zurich Seminar on Communications*, February 2006.

[7] A. Eryilmaz, R. Srikant, and J. R. Perkins. Stable scheduling policies for fading wireless channels. *IEEE/ACM Transactions on Networking*, 13:411–425, April 2005.

[8] F. P. Kelly, A. Maulloo, and D. Tan. Rate control in communication networks: Shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49:237–252, 1998.

[9] X. Lin and N. Shroff. The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks. In *Proceedings of IEEE Infocom*, Miami, FL, March 2005.

[10] S. H. Low and D. E. Lapsley. Optimization flow control, I: Basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7:861–875, December 1999.

[11] P. Marbach. Rate control for random access networks: The finite node case. In *IEEE Conference on Decision and Control*, 2004.

[12] P. Marbach, A. Eryilmaz, and A. Ozdaglar. Achievable rate region of csma schedulers in wireless networks with primary interference constraints. In *IEEE Conference on Decision and Control*, 2007.

[13] S. Meyn and R. Tweedie. *Markov Chains and Stochastic Stability*. Springer-Verlag, 1993.

[14] E. Modiano, D. Shah, and G. Zussman. Maximizing throughput in wireless networks via gossiping. In *ACM SIGMETRICS/IFIP Performance*, 2006.

[15] M.J. Neely, E. Modiano, and C. Li. Fairness and optimal stochastic control for heterogeneous networks. In *Proceedings of IEEE Infocom*, pages 1723–1734, Miami, FL, March 2005.

[16] M.J. Neely, E. Modiano, and C.E. Rohrs. Dynamic power allocation and routing for time varying wireless networks. In *Proceedings of IEEE Infocom*, pages 745–755, April 2003.

[17] S. Sanghavi, L. Bui, and R. Srikant. Distributed link scheduling with constant overhead, 2007. Technical Report.

[18] R. Srikant. *The Mathematics of Internet Congestion Control*. Birkhäuser, Boston, MA, 2004.

[19] A. Stolyar. Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm. *Queueing Systems*, 50(4):401–457, 2005.

[20] L. Tassiulas. Linear complexity algorithms for maximum throughput in radio networks and input queued switches. In *Proceedings of IEEE Infocom*, pages 533–539, 1998.

[21] 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*, 36:1936–1948, December 1992.

[22] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 39:466–478, March 1993.

[23] X. Wu and R. Srikant. Regulated maximal matching: A distributed scheduling algorithm for multi-hop wireless networks with node-exclusive spectrum sharing. In *Proceedings of* IEEE Conference on Decision and Control., 2005.