Distributed and Provably Efficient Algorithms for Joint Channel-Assignment, Scheduling, and Routing in Multichannel Ad Hoc Wireless Networks

Xiaojun Lin, *Member, IEEE*, and Shahzada B. Rasool, *Student Member, IEEE*

***Abstract*—The capacity of ad hoc wireless networks can be substantially increased by equipping each network node with multiple radio interfaces that can operate on multiple nonoverlapping channels. However, new scheduling, channel-assignment, and routing algorithms are required to fully utilize the increased bandwidth in multichannel multiradio ad hoc networks. In this paper, we develop fully distributed algorithms that jointly solve the channel-assignment, scheduling, and routing problem. Our algorithms are online algorithms, i.e., they do not require prior information on the offered load to the network, and can adapt automatically to the changes in the network topology and offered load. We show that our algorithms are provably efficient. That is, even compared with the optimal centralized and offline algorithm, our proposed distributed algorithms can achieve a provable fraction of the maximum system capacity. Furthermore, the achievable fraction that we can guarantee is larger than that of some other comparable algorithms in the literature.**

***Index Terms*—Channel assignment, distributed algorithms, efficiency ratio, multichannel multiradio ad hoc wireless networks, routing, scheduling.**

I. Introduction

MULTICHANNEL multiradio ad hoc wireless networks have recently received a substantial amount of interest, especially under the context of wireless mesh networks [2]–[14]. It has been shown that one can significantly increase the capacity of ad hoc wireless networks by equipping each network node with multiple radio interfaces that operate on multiple nonoverlapping channels. This is motivated by some current wireless LAN standards (in particular, IEEE 802.11) where the entire frequency band is divided into multiple channels, and each radio can only access one channel at a time. Hence, if each network node has multiple radio interfaces, it can then utilize a larger amount of radio bandwidth, and hence can achieve higher system capacity. Even if each node still has only one radio interface, by operating neighboring nodes at different channels, the amount of interference is reduced, which also leads to higher system capacity.

Such multichannel multiradio networks pose an interesting set of resource allocation problems, including: 1) *channel-assignment:* what are the set of channels that each node/link should operate on? 2) *scheduling:* when should each link be activated at each channel? and 3) *routing:* how does one select paths that minimize interference and increase throughput? These three problems are interrelated with each other, and thus form a challenging cross-layer control problem across the MAC layer and the network layer.

In this work, we are interested in control protocols for multichannel multiradio ad hoc networks that achieve high system capacity. Although such control protocols for channel-assignment, scheduling and routing can be obtained via the *throughput-optimal* algorithms in [15] and [16] that are known to achieve the maximum system capacity, these algorithms are centralized and often with exponential computational-complexity. Hence, they are not easy to implement in real systems. In this paper, we develop *distributed* and *provably efficient* solutions to the above cross-layer control problem. By *provably efficient*, we mean that for any offered load that a given multichannel network can ever support (possibly by using the centralized and complex throughput-optimal algorithms of [15] and [16]), our algorithms can guarantee to support at least a constant fraction of this offered load on the same network. In other words, our algorithms can achieve a provable fraction of the maximum system capacity. Our proposed algorithms are *online* algorithms, i.e., they do not require prior knowledge of the offered load, and can automatically track the changes in the network topology and offered load. The *distributed* and *online* nature of our solution, combined with its *provable efficiency*, differentiates our work from existing control algorithms in the literature for multichannel multiradio ad hoc networks that either do not guarantee provable performance bounds [2], [3], [6], [10]–[14], require centralized and offline solutions [8], [9], [11], [15], or only provide order-optimal scaling laws [5].

Our proposed algorithms are perhaps most comparable to the polynomial complexity (but centralized) algorithm in [8], which is also shown to guarantee a certain fraction of the maximum system capacity. Compared with the centralized solution of [8], the control algorithms proposed in this paper are not only distributed and much simpler, but also guarantee a higher fraction of the maximum system capacity (see the comparison immediately after Proposition 4 in Section IV-B for details). One of the key differences between our approach and that of [8] is

Manuscript received March 09, 2008; revised December 26, 2008; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor P. Thiran. First published July 14, 2009; current version published December 16, 2009. This work was supported in part by the NSF under Grant CNS-0626703 and by a grant from the Purdue Research Foundation. An earlier version of this paper has appeared in the *Proceedings of IEEE INFOCOM*, 2007.

The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: linx@ecn.purdue.edu; srasool@ecn.purdue.edu).

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

Digital Object Identifier 10.1109/TNET.2009.2021841

our assumption that, if the number of radio interfaces of a network node is less than the number of channels, the network node can switch radios from one channel to another dynamically [2], [3], [6], [7], [9], [14]. In contrast, the work in [8] requires radio-channel assignment to be fixed. It does incur additional overhead in order to enable dynamic channel switching, which include both the hardware switching delay [2], and the protocol overhead for changing channel-assignment. However, these types of overhead could be reduced by improved hardware technology and refined protocols. More importantly, we believe that our results provide a strong motivation to pursue such improved channel-switching hardwares and protocols because, by allowing dynamic channel switching, one can obtain control protocols (like the one developed in this paper) that are both simpler and with higher provable performance.

Our work is related to the recent progress in developing distributed and provably efficient scheduling algorithms for *single-channel* multihop wireless networks [17]–[24]. However, as we will show in Section II-B, straightforward extensions of some single-channel distributed scheduling algorithms to multichannel networks may lead to very poor performance. The reason is that, in multichannel networks, there may exist *channel-diversity*. That is, due to both frequency-selective multipath fading and different amount of background interference, each link can have different rate at each channel. In Section III-B, we will provide examples to show that straightforward extensions of a well-studied single-channel distributed scheduling algorithm, i.e., the Maximal Scheduling algorithm, can perform very poorly in multichannel systems with channel diversity. In contrast, the algorithms that we develop in this paper can guarantee provable efficiency even with channel diversity.

Our work is also related to the *opportunistic scheduling* algorithms in cellular networks that use orthogonal frequency division multiplexing (OFDM). Note that in OFDM, the transmitter sends information over a large number of subcarriers (e.g., 52 subcarriers in IEEE 802.11a). Hence, OFDM systems can also be viewed as a special type of multichannel multiradio wireless systems. *Opportunistic scheduling* algorithms can significantly improve the capacity of such systems by exploiting subcarrier frequency-diversity (and also time-diversity) [25]–[27]. However, such opportunistic scheduling algorithms for cellular networks cannot be extended directly to ad hoc wireless networks because of the distributed nature of *ad hoc* networks. On the other hand, the algorithms that we develop in this paper can be viewed as *distributed opportunistic scheduling* algorithms for OFDM-based ad hoc wireless networks.

The rest of the paper is organized as follows. We first outline the network model in Section II. In Section III, we illustrate the important effect of channel diversity, and show that straightforward extensions of single-channel distributed scheduling algorithms may perform very poorly in multichannel networks with channel diversity. We then present a new distributed and provably efficient algorithm in Section IV. Simulation results are presented in Section V. Then, we conclude.

II. System Model

Consider a wireless network with N nodes and L links. Each link corresponds to a pair of transmitter node and receiver node. Let $b(l)$ and $e(l)$ denote the transmitter node and the receiver node, respectively, of link l. Let $E(i)$ denote the set of all links originating or terminating at node i. There are C frequency channels in the system. In order to take into account possible channel diversity, we use r_l^c to denote the rate at which link l can transfer data on channel c, provided that there are no interfering links transmitting on channel c at the same time. The interference relationship is defined as follows. For each link l, there is a set I_l of links that interfere with l. That is, if link l and another link in I_l are transmitting on the same channel at the same time, neither of the links can transfer any useful data. We assume that the interference relationship is symmetrical, i.e., $k \in I_l$ if and only if $l \in I_k$ for any two links k and l. For simplicity, we adopt the convention that $l \in I_l$. Note that the interference relationship is identical over all channels. Furthermore, the channels are nonoverlapping. Hence, it is perfectly fine that link l and another link $k \in I_l$ transmit at different channels at the same time. The above interference model is very general and can be used to model a large class of practical interference relationships, including IEEE 802.11 DCF [19], [21], Bluetooth, and FH-CDMA [28].

A key parameter that will impact the performance of the control algorithms proposed in this paper is the *interference degree* [21], [22]. We first define a *noninterfering subset* of I_l as a subset of links in I_l such that any two links in this subset do not interfere with each other. The interference degree $\mathcal{K}(l)$ of link l is the maximum cardinality of any noninterfering subset of I_l. In other words, the interference degree $\mathcal{K}(l)$ of link l characterizes the potential loss of system capacity if link l is scheduled, i.e., it is the maximum number of links that could have been turned on simultaneously (without interference) if link l was not turned on. The interference degree $\mathcal{K}$ of the whole network is the maximum possible interference degree over all links. Note that the interference degree of the system can often be determined directly from the physical interference model, and thus can be independent of the exact network topology. For example, under the so-called node-exclusive interference model, each link only interferes with other links that share a common node. This model has been used to model Bluetooth and FH-CDMA networks, and its interference degree is 2. For the so-called bidirectional equal-power model that approximates IEEE 802.11 DCF, the interference degree is 8 [21].

Let M_i be the number of radio interfaces available at node i. We assume that at any given time a radio can only tune to one channel. Therefore, for link l to successfully communicate on channel c, both the transmitting node $b(l)$ and the receiving node $e(l)$ must tune one radio to channel c. Like [2], [3], [6], [7], [9], and [14], we assume that radios can switch channels dynamically.

There are S users in the system. Each user is associated with a source node and a destination node. The traffic from each user may be routed over multiple alternate paths. Let $J(s)$ denote the number of alternate paths for user s. Let $[H_{sj}^l]$ denote the routing matrix, where $H_{sj}^l = 1$ if path j of user s traverses link l, $H_{sj}^l = 0$, otherwise. We assume that time is divided into slots of unit length. Let $A_s(t)$ denote the number of packets that user s injects into the system at time-slot t. For simplicity, we assume that $A_s(t)$ is i.i.d. across time, and is bounded from above

(while our results can also be generalized to more general arrival patterns, e.g., when packets arrive according to some Markovian arrival processes). Let $\lambda_s = \mathbf{E}[A_s(1)]$.

We assume that the channel assignment can be changed every time-slot, and the time that it takes to switch radios between channels is negligible compared to the length of each time slot. For ease of exposition, in the rest of the paper whenever there is no source of confusion, we will use the term "schedule" to refer to both the channel assignment and the link schedule at a time slot. At time slot t, let $\mathcal{M}(t) = [\mathcal{M}^c(t)]$ denote the outcome of the scheduling algorithm, where $\mathcal{M}^c(t)$ is the set of noninterfering links that are chosen to transmit at channel c at time t. Let $D_l(t)$ denote the number of packets that link l can serve at time slot t. Then $D_l(t) = \sum_{c:l\in\mathcal{M}^c(t)} r_l^c$. Let $P_{sj}(t)$ denote the fraction of traffic from user s that is routed to path j at time slot t. Furthermore, let $q_l(t)$ denote the number of packets queued at link l at the beginning of time slot t. The evolution of $q_l(t)$ may be written as

$$q_l(t+1) = [q_l(t) + \sum_{s=1}^{S}\sum_{j=1}^{J(s)} H_{sj}^l P_{sj}(t)A_s(t) - D_l(t)]^+ \quad (1)$$

where $[\cdot]^+$ denote the projection to $[0, +\infty)$. We say that the system is *stable* if the queue lengths at all links remain finite [16], i.e.,

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\mathbf{1}_{\left\{\sum_{l=1}^{L} q_l(t)>\eta\right\}} \to 0, \text{ almost surely as } \eta\to\infty.$$

Remark: Note that in (1) we have adopted the simplifying assumption that packets from each user s are applied to all links l along the path of user s simultaneously. This assumption simplifies the analysis of the paper, and allows us to focus on the channel assignment and scheduling components of the problem. In reality, packets have to traverse the links one at a time. There are a number of known methodologies that can extend our model to take into account this link-by-link packet dynamics [17], [19], [20], [24]. One such approach is to assume that users can communicate the amount of transmitted traffic $P_{sj}(t)A_s(t)$ to all links along path j through an additional control channel, and let each link l update a "virtual" queue according to (1). Then, with an appropriate packet scheduling policy at each link, one can show that the real queue (with link-by-link packet dynamics) is stable as long as the "virtual" queue defined by (1) is stable [17], [29]. Hence, in this paper we will neglect the link-by-link packet dynamics, and use (1) to describe the queue dynamics.

Let $\vec{\lambda} = [\lambda_1, \ldots \lambda_S]$ denote the offered load to the network. The *capacity region* under a particular channel-assignment, scheduling and routing algorithm is the set of $\vec{\lambda}$ such that the system remains stable. Under possible routing constraints, we define the *optimal capacity region* Ω as the supremum of the capacity regions of all algorithms. An algorithm is *throughput-optimal* if it can achieve the optimal capacity region Ω. An algorithm is said to achieve an *efficiency ratio* of γ if it can stablize the system under any load $\vec{\lambda}$ that lies strictly in $\gamma\Omega$. By definition, a throughput-optimal algorithm has an efficiency ratio of 1. The key variables in the system model are summarized in Table I.

TABLE I
TABLE OF ALL NOTATIONS

N, L, C, S	number of nodes, links, channels and users
$b(l), e(l)$	transmitter/receiver node of link l
$E(i)$	set of links incident at node i
r_l^c	rate of link l on channel c
I_l	set of links that interfere with link l
M_i	number of radios available at node i
$J(s)$	number of alternate paths for user s
$[H_s^l], [H_{sj}^l]$	routing matrix
$A_s(t)$	number of packets injected by user s at time t
λ_s	average packet injection rate of user s
$\mathcal{M}(t), \mathcal{M}^c(t)$	the channel assignment and schedule at time t
$D_l(t)$	number of packets served by link l at time t
$P_{sj}(t)$	fraction of traffic from user s that is routed to path j
$q_l(t)$	number of packets queued at link l
$\eta_l^c(t)$	number of packets assigned to channel queue c of link l
Ω	the optimal capacity region
γ	the efficiency ratio
$\mathcal{K}, \mathcal{K}(l)$	the interference degree
$\alpha_l, \beta_s, e_{sj}$	parameters in the control algorithms SP and MP

III. Generalizations of Single-Channel Scheduling Algorithms to Multchannel Networks

In this paper, we are interested in distributed control algorithms with provable efficiency ratios. A number of provably efficient and low-complexity scheduling algorithms have been proposed for *single-channel* multihop wireless networks [17], [19]–[24]. One would naturally hope that the generalization of these single-channel scheduling algorithms may lead to equally efficient and low-complexity scheduling algorithms for multichannel networks. Unfortunately, this is not always the case. In this section, we study the generalization of two such single-channel scheduling algorithms, i.e., Greedy Maximal Scheduling and Maximal Scheduling, to multichannel networks. We will first show that the generalization of Greedy Maximal Scheduling can still guarantee efficiency-ratios almost as tight as in single-channel networks. While this result is encouraging, the complexity of Greedy Maximal Scheduling grows linearly with the size of the network. It would be more desirable that we can develop distributed scheduling algorithms for multichannel ad hoc wireless networks with even lower complexity. However, we will show that straightforward extensions of Maximal Scheduling, an even simpler low-complexity and distributed algorithm, can result in much lower efficiency ratios in multichannel networks.

A. Greedy Maximal Scheduling

It is known that the maximum system capacity can be attained by the following throughput-optimal scheduling algorithm [15]–[17]: at each time slot t, the schedule $\mathcal{M}(t)$ should be chosen to maximize the queue-weighted rate-sum, i.e.,

$$\mathcal{M}(t) = \operatorname*{argmax}_{\mathcal{M}} \sum_{(l,c):l\in\mathcal{M}^c} q_l(t) r_l^c. \quad (2)$$

Furthermore, if an algorithm can generate schedules whose weights are guaranteed to be above a fraction γ of the maximum weight in (2), then the algorithm will achieve an efficiency ratio

of γ [17], [18]. The Greedy Maximal Scheduling algorithm is one such algorithm. Roughly speaking, a schedule $\mathcal{M}(t)$ is *maximal* if $\mathcal{M}(t)$ is a noninterfering schedule, and no more links can be added to $\mathcal{M}^c(t)$ at any channel c without violating the interference constraint and radio interface constraint. The Greedy Maximal Scheduling algorithm computes a maximal schedule by always starting from the link-channel pair with the largest queue-weighted rate $q_l(t)r_l^c$. Specifically, the Greedy Maximal Scheduling algorithm proceeds as follows in each time-slot t.

Greedy Maximal Scheduling:

(a) Form a set $\mathcal{F}$ of all link-channel pairs (l, c). Define the weight of each link-channel pair (l, c) to be $q_l(t)r_l^c$, where $q_l(t)$ is the current queue length at link l. Start from an empty schedule $\mathcal{M}(t)$.

(b) First search for the link-channel pair (l, c) with the largest weight $q_l(t)r_l^c$. Add link l to $\mathcal{M}^c(t)$.

(c) Remove from $\mathcal{F}$ all link-channel pairs that cannot be scheduled after (l, c) is scheduled, due to either the interference constraints in channel c or the radio interface constraints at end-points of link l.

(d) Find the link-channel pair (l, c) with the largest weight $q_l(t)r_l^c$ from the *remaining* pairs in $\mathcal{F}$. Add link l to $\mathcal{M}^c(t)$.

(e) Repeat Step (c) and Step (d) until no link-channel pairs are left in $\mathcal{F}$.

Note that the above Greedy Maximal Scheduling algorithm is a natural generalization of the Greedy Maximal Scheduling algorithm for single-channel networks [17] and high-speed switches [30].

The following proposition shows that Greedy Maximal Scheduling can guarantee an efficiency ratio of $1/(\mathcal{K}+2)$ in multichannel networks, where $\mathcal{K}$ is the interference degree defined in Section II. Note that the Greedy Maximal Scheduling algorithm has been shown to achieve an efficiency ratio of $1/\mathcal{K}$ in single-channel networks [17]. Thus, we can conclude that its performance in multichannel networks is similar. For simplicity, we only present the result for the case where each user has one fixed path through the network. Let $H_s^l = 1$ if the path of user s uses link l, $H_s^l = 0$, otherwise.

Proposition 1: Assume that each user can only use one path, and the routing matrix is given by $[H_s^l]$. The above Greedy Maximal Scheduling algorithm can achieve an efficiency ratio of $1/(\mathcal{K}+2)$. Furthermore, if $M_i = C$ for all i, i.e., there are no radio interface constraints, Greedy Maximal Scheduling can achieve an efficiency ratio of $1/\mathcal{K}$.

The proof of Proposition 1 is provided in our online technical report [31], and is along the line of similar results for switches [30] and for single-channel networks [17], [18]. The main idea is to show that the weight of the schedule produced by Greedy Maximal Scheduling is guaranteed to be above $1/(\mathcal{K}+2)$ of the maximum weight of the optimal schedule in (2). The Greedy Maximal Scheduling algorithm can be implemented by a centralized algorithm with complexity $O(L\log L)$, where L is the total number of links. Alternatively, it may be implemented by a distributed algorithm with time-complexity of $O(L)$ (see [32], which is a distributed variant of the algorithm in [33][1]). Note that in the literature, there are other algorithms that can generate schedules with weights guaranteed to be above a certain fraction of the optimal schedule. Some of these algorithms are distributed and have even lower complexity than Greedy Maximal Scheduling (see, e.g., [34]–[36]). However, these distributed algorithms either guarantee lower approximation ratios than Greedy Maximal Scheduling [34], or require additional geometric constraints on the network topology [35], [36].

B. Maximal Scheduling

For single-channel networks, another algorithm called Maximal Scheduling is known to guarantee an efficiency ratio of $1/\mathcal{K}$, is easy-to-implement in a distributed fashion, and has even lower complexity than Greedy Maximal Scheduling. Under the node-exclusive interference model, the complexity of Maximal Scheduling can be as low as $O(\log L)$ for single-channel networks [37]. Hence, it would be interesting to see whether extensions of Maximal Scheduling to multichannel ad hoc wireless networks can also attain similar efficiency ratios as Greedy Maximal Scheduling. Recall the definition of a maximal schedule in Section III-A. Mathematically, a maximal schedule can be stated as follows.

Multichannel Maximal Scheduling: A multichannel maximal schedule $\mathcal{M}(t)$ consists of links l that are backlogged, i.e., $q_l(t) \geq \sum_{c=1}^{C} r_l^c$. Furthermore, for any link-channel pair (l, c) such that link l is backlogged, at least one of the following is true:

- either link l is scheduled in channel c, i.e., $l \in \mathcal{M}^c(t)$;
- one of the interfering links k to link l (i.e., $k \in I_l$) is backlogged and scheduled in channel c (i.e., $k \in \mathcal{M}^c(t)$);
- either the transmitter or the receiver of link l has used up all the radios, i.e., $\sum_{k\in E(b(l))}\sum_{d=1}^{C}\mathbf{1}_{\{k\in\mathcal{M}^d(t)\}} = M_{b(l)}$, or $\sum_{k\in E(e(l))}\sum_{d=1}^{C}\mathbf{1}_{\{k\in\mathcal{M}^d(t)\}} = M_{e(l)}$.

For single-channel networks, such Maximal Scheduling algorithms have been shown to achieve an efficiency ratio of $1/\mathcal{K}$. However, for multichannel networks with channel diversity, such algorithms can perform much worse, because the algorithm could pick the weakest (i.e., least capacity) links at each channel into the maximal schedule. A slightly improved version of Maximal Scheduling for multichannel networks would be to enforce an additional constraint that the scheduling pattern on all channels must be the same. Assume that $M_i = C$ for all node i, i.e., each node has enough radio interfaces to access all channels simultaneously. Thus, if we aggregate all channels together, the capacity of link l is $\sum_{c=1}^{C} r_l^c$. We can then define the following Aggregated Maximal Scheduling as in single-channel networks.

[1]Although these algorithms are designed to compute matching for graphs, it is not difficult to generalize them to multichannel networks. Essentially, a link-channel pair can schedule itself if it is the "locally heaviest" [32], i.e., its weight is larger than all other conflicting link-channel pairs. (Ties can be broken using some preassigned ID numbers.) Otherwise, it will wait until the conflicting link-channel pairs with larger weights have decided. Then, it will either give up (if a conflicting link-channel pair with higher weight has been scheduled), or schedule itself (if all other conflicting link-channel pairs with higher weights have given up).

Aggregated Maximal Scheduling:

- If link l is scheduled, it will transmit over all channels simultaneously.
- For any link l that is backlogged (i.e., $q_l(t) \geq \sum_{c=1}^{C} r_l^c$), either link l is scheduled, or some other backlogged link $k \in I_l$ is scheduled.

However, the performance of Aggregated Maximal Scheduling can still be very poor. We note that the weak performance of Aggregated Maximal Scheduling has less to do with the inefficiency of single-channel Maximal Scheduling, but has more to do with the fact that all channels are aggregated into a single channel. In fact, the following example shows that, once all channels are aggregated into a single channel, regardless of the scheduling algorithm used, the efficiency ratio can be as low as $1/\bar{I}$, where $\bar{I}$ is the maximum number of links that interfere with any link l. To see this, consider a node 0 communicating with nodes $1, \ldots, \bar{I}$. Label the links between node 0 and node i as link i. Assume that each link interferes with all other links if they operate on the same channel. Hence, only one link can be assigned to each channel at any time. Assume that the total number of channels is also $\bar{I}$. Let the capacity of link i at channel i to be 1, while its capacity at all other channels is ϵ. Thus, if we aggregate all channels into a single channel, the aggregate capacity of link i is $1 + \epsilon(\bar{I} - 1)$. Since only one link can be activated at each time, the total capacity of the system is $1 + \epsilon(\bar{I} - 1)$. However, if we assign each link i to operate on channel i, there will be no interference in the system, and the total capacity of the system is thus $\bar{I}$. Hence, regardless of the scheduling algorithm used, as ϵ approaches zero, the efficiency ratio will be arbitrarily close to $1/\bar{I}$.

The above example clearly illustrates the weakness of Maximal Scheduling algorithms in multichannel networks with channel diversity. In particular, the above extensions of Maximal Scheduling to multichannel networks fail to take into account the channel diversity, and hence the performance of the scheduling decisions can be very poor.

IV. Distributed and Provably Efficient Multchannel Control Algorithm Based on Maximal Scheduling

Given the results in Section III, a natural question is then: can we develop a distributed scheduling algorithm for multichannel multiradio ad hoc wireless networks that can guarantee the same efficiency ratio as the Greedy Maximal Scheduling algorithm, but with lower complexity comparable to Maximal Scheduling? In this section, we will develop such a distributed scheduling algorithm. Interestingly, our new scheduling algorithm still uses maximal schedules. Obviously, in order to avoid the inefficiency illustrated in Section III-B, we must be able to properly take into account channel diversity. In this work, we introduce a novel *two-stage queueing* mechanism to address channel diversity, and to prevent links from using channels that are weak (i.e., with smaller capacity).

The basic idea of two-stage queueing is as follows. Packets arriving to each link l are served in two steps. The first step is a *logical* assignment of the packets to channels: packets that arrive at each link l are assigned to queues that correspond to each channel c. The second step is the *actual* scheduling of radio interfaces and links: radios are assigned to channels according to maximal schedules, and packets in the channel queues are served. The key of two-stage queueing is to ensure that, in the first step, packets are less likely to be assigned to link-channel pairs that are "weak." Thus, the "weak" links do not even participate in the maximal schedules in the second step. Clearly, the main difficulty is how to determine in a distributed and online fashion which link-channel pairs are "weak." As we will show next, our algorithm makes this decision intelligently by using the queue length information at both the per-link level and the per-channel level.

A. Single-Path Case

For ease of exposition, in this subsection we first focus on the case where each user only has one fixed path through the network. Let $H_s^l = 1$ if the path of user s uses link l, $H_s^l = 0$, otherwise. Our proposed multichannel multiradio scheduling algorithm works as follows. Each link maintains $C + 1$ queues. There is one queue q_l for each link l, which represents the backlog of packets at link l that have not been assigned to channel queues yet. At the same time, each link l maintains C channel-queues $\eta_l^c, c = 1, \ldots, C$. The per-channel queue η_l^c represents the backlog of packets assigned to channel c by link l that are still waiting to be served. Note that both q_l and η_l^c evolve as a function of time t, and in general $q_l(t) \neq \sum_{c=1}^{C} \eta_l^c(t)$. At each time slot t, the following algorithm is executed.

Algorithm SP:

- Step 1: Define $x_l^c(t)$ to be the number of packets that link l can assign to channel c at time-slot t. For each link l, let

$$x_l^c(t) = r_l^c, \text{ if } \frac{q_l(t)}{\alpha_l} \geq \frac{1}{r_l^c}\left[\sum_{k\in I_l}\frac{\eta_k^c(t)}{r_k^c} + \frac{1}{M_{b(l)}}\sum_{k\in E(b(l))}\sum_{d=1}^{C}\frac{\eta_k^d(t)}{r_k^d} + \frac{1}{M_{e(l)}}\sum_{k\in E(e(l))}\sum_{d=1}^{C}\frac{\eta_k^d(t)}{r_k^d}\right]$$
$$x_l^c(t) = 0, \text{ otherwise} \tag{3}$$

where α_l is an arbitrary positive constant chosen for link l. Then, link l drains $\min\{q_l(t), \sum_{c=1}^{C} x_l^c(t)\}$ packets from q_l, and assign them to each channel queue[2]. Let $y_l^c(t) \in [0, x_l^c(t)]$ be the actual number of packets assigned to each channel queue η_l^c. Recall that in the same time slot, link l also receives $\sum_{s=1}^{S} H_s^l A_s(t)$ new packets. The evolution of q_l is thus given by

$$q_l(t+1) = q_l(t) + \sum_{s=1}^{S} H_s^l A_s(t) - \sum_{c=1}^{C} y_l^c(t) \tag{4}$$

where

$$\sum_{c=1}^{C} y_l^c(t) = \min\left\{q_l(t), \sum_{c=1}^{C} x_l^c(t)\right\}. \tag{5}$$

[2]This assignment can be more precisely described as follows: If $q_l(t) \geq \sum_{c=1}^{C} x_l^c(t)$, then $y_l^c(t) = x_l^c(t)$ for all channel c. Otherwise, choose any $0 \leq y_l^c(t) \leq x_l^c(t)$ such that $\sum_{c=1}^{C} y_l^c(t) = q_l(t)$. For example, if there are three channels, $x_l^1(t) = 1, x_l^2(t) = 2, x_l^3(t) = 3$, and $q_l(t) = 3$, then we can choose, as an example, $y_l^1(t) = y_l^2(t) = y_l^3(t) = 1$.

- Step 2: Based on the channel queues $\eta_l^c(t)$, Multichannel Maximal Scheduling (as in Section III-B) is carried out to determine the channel-assignment and link schedules. Mathematically, this means that $\mathcal{M}^c(t)$ again consists of links l that are backlogged in channel c (which is now defined as $\eta_l^c(t) \geq r_l^c$). Furthermore, for any link-channel pair (l,c) that is backlogged, one of the statements under the Multichannel Maximal Scheduling algorithm in Section III-B must hold. Then at the end of step 2, the evolution of each channel queue $\eta_l^c(t)$ is given by

$$\eta_l^c(t+1) = \eta_l^c(t) + y_l^c(t) - r_l^c \mathbf{1}_{\{l \in \mathcal{M}^c(t)\}}. \tag{6}$$

Remark: The assignment in (3) is the key to ensure that links will only be scheduled on their "strong" channels. Roughly speaking, this rule means that packets will have a better chance to be assigned to a channel-queue if the corresponding rate r_l^c is large (note that r_l^c appears in the denominator of the right-hand side). Note that in (3) each link only needs to know the channel-queue-length $\eta_l^c(t)$ at interfering links. This rule can be best explained by interpreting the quantities $q_l(t)$ and $\eta_l^c(t)$ as "price" signals. The quantity $q_l(t)$ can be interpreted as the *congestion cost* at link l (due to the imbalance between the external arrivals and the system capacity). The quantity $\sum_{k\in I_l} \frac{\eta_k^c(t)}{r_k^c}$ can be interpreted as the *contention cost* at link l (due to the interference on channel c from links in the interference set I_l). The quantities $\frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{\eta_k^d(t)}{r_k^d}$ and $\frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{\eta_k^d(t)}{r_k^d}$ can be interpreted as the *radio costs* at the transmitter node $b(l)$ and receiver node $e(l)$, respectively, of link l. Hence, each link l will assign traffic to channel c at the maximum rate r_l^c only if the contention cost of the channel plus the radio cost, weighted by the channel capacity r_l^c, is smaller than the congestion level at the link.

Note that in the definition of Multichannel Maximal Scheduling in Section III-B, we have explicitly given priority to links that are backlogged. Similarly, in Step 2 of Algorithm *SP*, we only require the link-channel pairs that are backlogged (i.e., $\eta_l^c(t) \geq r_l^c$) to participate in the schedule computation in each channel. As a result, the updated channel-queue in (6) is always nonnegative. It is possible that, in addition to a multichannel maximal schedule, one can simultaneously activate additional link-channel pairs that are *not* backlogged (i.e., $\eta_l^c(t) < r_l^c$) and that still satisfy the interference constraints and the radio interface constraints. Specifically, we can modify Step 2 as follows: If after computing multichannel maximal schedule for backlogged link-channel pairs, there are still other link-channel pairs that could be activated, we can include those link-channel pairs into the schedule as long as they do not interfere with the link-channel pairs that have been scheduled. In this case, the evolution of the channel queue becomes

$$\eta_l^c(t+1) = [\eta_l^c(t) + y_l^c(t) - r_l^c \mathbf{1}_{\{l \in \tilde{\mathcal{M}}^c(t)\}}]^+ \tag{7}$$

where $\tilde{\mathcal{M}}^c(t)$ consists of both $\mathcal{M}^c(t)$ and the additional link-channel pairs that are scheduled but are not backlogged. This modification will help to more quickly drain those queues that are not backlogged. Our main result (Proposition 2) does not require scheduling those link-channel pairs that are *not* backlogged, and hence applies to both update (6) and update (7).

The following main result shows that the efficiency ratio of Algorithm SP is identical to that of Greedy Maximal Scheduling in Section III-A.

Proposition 2: Assume that each user can only use one path, and the routing matrix is given by $[H_s^l]$. The efficiency ratio of our proposed algorithm SP is $1/(\mathcal{K}+2)$ where $\mathcal{K}$ is the interference degree defined in Section II.

Proof: We will show that for any $\vec{\lambda}$ such that some scheduling algorithm can stabilize the network at the offered load $(\mathcal{K}+2)\vec{\lambda}$, Algorithm SP will stabilize the system at the offered load $\vec{\lambda}$. We use the following Lyapunov function to establish stability

$$V(\vec{q},\vec{\eta}) = V_q(\vec{q}) + V_\eta(\vec{\eta}) \tag{8}$$

where

$$V_q(\vec{q}) = \sum_{l=1}^{L} \frac{(q_l)^2}{2\alpha_l},$$

$$V_\eta(\vec{\eta}) = \sum_{l=1}^{L} \sum_{c=1}^{C} \frac{\eta_l^c}{2r_l^c} \Bigg[\sum_{k\in I_l} \frac{\eta_k^c}{r_k^c} + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{\eta_k^d}{r_k^d} + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{\eta_k^d}{r_k^d} \Bigg].$$

Let $\Delta V(t) = V(\vec{q}(t+1),\vec{\eta}(t+1)) - V(\vec{q}(t),\vec{\eta}(t))$. Since we assume that $A_s(t)$ is bounded, we have

$$\begin{aligned} &\frac{(q_l(t+1))^2}{2} - \frac{(q_l(t))^2}{2} \\ &\quad = q_l(t)(q_l(t+1) - q_l(t)) + \frac{(q_l(t+1) - q_l(t))^2}{2} \\ &\quad \leq q_l(t) \Bigg[\sum_{s=1}^{S} H_s^l A_s(t) - \sum_{c=1}^{C} y_l^c(t) \Bigg] + C_1 \end{aligned}$$

where C_1 upper bounds $(q_l(t+1) - q_l(t))^2$ for any l and t. According to (5), $\sum_{c=1}^{C} y_l^c(t) = \sum_{c=1}^{C} x_l^c(t)$ whenever $q_l(t) \geq \sum_{c=1}^{C} r_l^c$. Hence, we can show that

$$\begin{aligned} &V_q(\vec{q}(t+1)) - V_q(\vec{q}(t)) \\ &\qquad \leq \sum_{l=1}^{L} \frac{q_l(t)}{\alpha_l} \Bigg[\sum_{s=1}^{S} H_s^l A_s(t) - \sum_{c=1}^{C} x_l^c(t) \Bigg] + C_2 \end{aligned}$$

for some constant C_2. Similarly, we can show that

$$\begin{aligned} &V_\eta(\vec{\eta}(t+1)) - V_\eta(\vec{\eta}(t)) \\ &\leq \sum_{l=1}^{L} \sum_{c=1}^{C} \frac{\eta_l^c(t)}{r_l^c} \Bigg[\sum_{k\in I_l} \frac{y_k^c(t)}{r_k^c} + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{y_k^d(t)}{r_k^d} \\ &\qquad + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{y_k^d(t)}{r_k^d} - \mu_l^c(t) \Bigg] + C_3 \\ &\leq \sum_{l=1}^{L} \sum_{c=1}^{C} \frac{\eta_l^c(t)}{r_l^c} \Bigg[\sum_{k\in I_l} \frac{x_k^c(t)}{r_k^c} + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{x_k^d(t)}{r_k^d} \\ &\qquad + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{x_k^d(t)}{r_k^d} - \mu_l^c(t) \Bigg] + C_3 \end{aligned}$$

where C_3 is a positive constant and

$$\mu_l^c(t) = \sum_{k\in I_l} \mathbf{1}_{\{k\in\mathcal{M}^c(t)\}} + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \mathbf{1}_{\{k\in\mathcal{M}^d(t)\}} + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \mathbf{1}_{\{k\in\mathcal{M}^d(t)\}}. \quad (9)$$

Note that this inequality holds even if we replace the channel-queue update (6) by (7), because the activation of link-channel pairs that are not backlogged only further decreases the difference $V_\eta(\vec{\eta}(t+1)) - V_\eta(\vec{\eta}(t))$. Therefore, the Lyapunov drift $\Delta V(t)$ can be bounded by

$$\begin{aligned}
&\mathbf{E}\Big[\Delta V(t)|\vec{q}(t),\vec{\eta}(t)\Big] \\
&\le \sum_{l=1}^{L} \frac{q_l(t)}{\alpha_l}\Big[\sum_{s=1}^{S} H_s^l\lambda_s - \sum_{c=1}^{C} x_l^c(t)\Big] \\
&\quad + \sum_{l=1}^{L}\sum_{c=1}^{C} \frac{\eta_l^c(t)}{r_l^c}\Big[\sum_{k\in I_l} \frac{x_k^c(t)}{r_k^c} + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{x_k^d(t)}{r_k^d} \\
&\qquad + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{x_k^d(t)}{r_k^d} - \mu_l^c(t)\Big] + C_4
\end{aligned}$$

for some positive constant C_4.

Since there exists some scheduling algorithm that can stabilize the system at the offered load vector $(\mathcal{K}+2)\vec{\lambda}$, there must exist some $\tilde{x}_l^c \in [0, r_l^c]$ for each (l,c) such that

$$(1+\epsilon)^2(\mathcal{K}+2)\sum_{s=1}^{S} H_s^l\lambda_s \le \sum_{c=1}^{C} \tilde{x}_l^c, \quad \text{for all link } l \quad (10)$$

$$\sum_{k\in I_l} \frac{\tilde{x}_k^c}{r_k^c} \le \mathcal{K}, \quad \text{for all link } l \text{ and channel } c \quad (11)$$

$$\sum_{k\in E(i)} \sum_{c=1}^{C} \frac{\tilde{x}_k^c}{r_k^c} \le M_i, \quad \text{for all node } i \quad (12)$$

where $\tilde{x}_l^c$ can be interpreted as the long-term average amount of service that link l received at channel c, and ϵ is a small positive number. Note that the inequality (10) is due to the rate-balance at link l. Inequality (11) is due to the interference constraint, i.e., there can be no more than $\mathcal{K}$ links activated simultaneously in any interference range I_l. The inequality (12) is due to the radio interface constraint, i.e., there can be no more than M_i link-channel pairs incident to node i that are activated simultaneously (see [8] and [9]). Let $\bar{x}_l^c = \frac{\tilde{x}_l^c}{(\mathcal{K}+2)(1+\epsilon)}$. We thus have

$$(1+\epsilon)\sum_{s=1}^{S} H_s^l\lambda_s \le \sum_{c=1}^{C} \bar{x}_l^c, \quad \text{for all link } l \quad (13)$$

$$(1+\epsilon)\Big[\sum_{k\in I_l} \frac{\bar{x}_k^c}{r_k^c} + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{\bar{x}_k^d}{r_k^d} + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{\bar{x}_k^d}{r_k^d}\Big] \le 1, \quad \text{for all } (l,c). \quad (14)$$

Therefore, we have

$$\begin{aligned}
&\mathbf{E}[\Delta V(t)|\vec{q}(t),\vec{\eta}(t)] \\
&\le \sum_{l=1}^{L} \frac{q_l(t)}{\alpha_l}\Big[\sum_{s=1}^{S} H_s^l\lambda_s - \sum_{c=1}^{C} \bar{x}_l^c\Big] \\
&\quad + \sum_{l=1}^{L} \frac{q_l(t)}{\alpha_l}\Big[\sum_{c=1}^{C} \bar{x}_l^c - \sum_{c=1}^{C} x_l^c(t)\Big] \\
&\quad + \sum_{l=1}^{L}\sum_{c=1}^{C} \frac{\eta_l^c(t)}{r_l^c}\Big[\sum_{k\in I_l} \frac{\bar{x}_k^c}{r_k^c} + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{\bar{x}_k^d}{r_k^d} \\
&\qquad + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{\bar{x}_k^d}{r_k^d} - \mu_l^c(t)\Big] \\
&\quad + \sum_{l=1}^{L}\sum_{c=1}^{C} \frac{\eta_l^c(t)}{r_l^c}\Big[\sum_{k\in I_l} \frac{x_k^c(t) - \bar{x}_k^c}{r_k^c} \\
&\qquad + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{x_k^d(t) - \bar{x}_k^d}{r_k^d} \\
&\qquad + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{x_k^d(t) - \bar{x}_k^d}{r_k^d}\Big] + C_4.
\end{aligned} \quad (15)$$

Let

$$m_l^c = \Big[\sum_{k\in I_l} \frac{\bar{x}_k^c}{r_k^c} + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{\bar{x}_k^d}{r_k^d} + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{\bar{x}_k^d}{r_k^d}\Big].$$

By definition of Multichannel Maximal Scheduling (see Section III-B), $\mu_l^c(t) \ge 1$ whenever $\eta_l^c(t) \ge r_l^c$. Using (13)–(14), we thus have

$$\begin{aligned}
&\mathbf{E}[\Delta V(t)|\vec{q}(t),\vec{\eta}(t)] \quad (16) \\
&\le -\epsilon \sum_{l=1}^{L} \frac{\sum_{s=1}^{S} H_s^l\lambda_s}{\alpha_l} q_l(t) - \epsilon \sum_{l=1}^{L}\sum_{c=1}^{C} \frac{m_l^c}{r_l^c}\eta_l^c(t) \\
&\quad + \sum_{l=1}^{L}\sum_{c=1}^{C} \frac{\bar{x}_l^c - x_l^c(t)}{r_l^c}\Big[\frac{r_l^c q_l(t)}{\alpha_l} - \Big(\sum_{k\in I_l} \frac{\eta_k^c(t)}{r_k^c} \\
&\qquad + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{\eta_k^d(t)}{r_k^d} \\
&\qquad + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{\eta_k^d(t)}{r_k^d}\Big)\Big] + C_5.
\end{aligned} \quad (17)$$

Recall that $0 \le \bar{x}_l^c \le r_l^c$. Then, by Step 1 of Algorithm SP, the third term in (17) is nonpositive. Therefore

$$\mathbf{E}[\Delta V(t)|\vec{q}(t),\vec{\eta}(t)] \le -\epsilon \sum_{l=1}^{L} \frac{\sum_{s=1}^{S} H_s^l \lambda_s}{\alpha_l} q_l(t) - \epsilon \sum_{l=1}^{L}\sum_{c=1}^{C} \frac{m_l^c}{r_l^c}\eta_l^c(t) + C_5.$$

The stability of the system then follows [16]. $Q.E.D.$

If $M_i = C$ for all node i, i.e., when there are no radio interface constraints, a tighter efficiency ratio can be shown by slightly modifying Algorithm SP. In particular, we can remove from (3) the radio costs (i.e., the last two terms). We can then use the following Lyapunov function:

$$V(\vec{q},\vec{\eta}) = \sum_{l=1}^{L} \frac{(q_l)^2}{2\alpha_l} + \sum_{l=1}^{L}\sum_{c=1}^{C} \frac{\eta_l^c}{2r_l^c}\left[\sum_{k\in I_l} \frac{\eta_k^c}{r_k^c}\right].$$

Following the line of proof of Proposition 2, and noting that the last two terms of (14) can also be removed. We can then show the following tighter result.

Proposition 3: Assume that each user can only use one path, and the routing matrix is given by $[H_s^l]$. Furthermore, assume that $M_i = C$ for all node i. The efficiency ratio of the modified Algorithm SP is $1/\mathcal{K}$.

In both cases, the distributed Algorithm SP achieves the same efficiency ratio as the higher-complexity Greedy Maximal Scheduling algorithm in Section III-A.

We briefly discuss the choice of α_l in (3). The parameter α_l will affect the balance between the link-queue $q_l(t)$ and channel-queues $\eta_l^c(t)$. If α_l is too large, Algorithm SP may not move packets from the link-queue to the channel-queues even when there is a large backlog in the link-queue $q_l(t)$. On the other hand, if α_l is too small, then the per-channel queue needs to be fairly large before the algorithm can avoid "weak" channels. In practice, we find that the algorithm performs satisfactorily when α_l is chosen to be at the same magnitude as $(\max_{l,c} r_l^c)^2$, which leads to comparable lengths of link-queues and channel-queues.

B. The Multipath Case

We next extend Algorithm SP to the case when each user can use multiple alternate paths. For the moment, we assume that each user is provided with $J(s)$ alternate paths through the network, and we will study how each user should optimally route packets among these alternate paths. Then, in Section IV-C, we will discuss how these paths should be computed. Let $H_{sj}^l = 1$ if the jth path of user s uses link l, and $H_{sj}^l = 0$, otherwise. Define P_{sj} to be the fraction of incoming packets from user s that are routed to path j. Let $\vec{P}_s = [P_{s1},\ldots,P_{s,J(s)}]$. Obviously, $P_{sj} \ge 0$ and $\sum_{j=1}^{J(s)} P_{sj} = 1$ for all user s. Let $\vec{P} = [\vec{P}_1,\ldots,\vec{P}_s]$. We can then generalize Algorithm SP to the following joint channel-assignment, scheduling and routing algorithm.

Algorithm MP: At each time slot t:

- Step 1: Each user s computes the routing fractions $P_{sj}(t)$ as the solution to the following optimization problem:

$$\max_{\vec{P}_s} -\frac{\beta_s}{2}\sum_{j=1}^{J(s)}(P_{sj})^2 - \sum_{j=1}^{J(s)} P_{sj}\left[\sum_{l=1}^{L} H_{sj}^l q_l(t) + e_{sj}\right] \tag{18}$$

$$\text{subject to } P_{sj} \ge 0, \quad \sum_{j=1}^{J(s)} P_{sj} = 1 \tag{19}$$

  where $[H_{sj}^l]$ is the routing matrix, β_s is a positive number chosen for each source s, and e_{sj} is a constant chosen for each path j of source s. Each user s then routes each arriving packet independently to path j with probability $P_{sj}(t)$. Let $\tilde{P}_{sj}(t)$ be the actual fraction of packets that user s routes to path j at time t. Note that $\mathbf{E}[\tilde{P}_{sj}(t)A_s(t)] = P_{sj}(t)\lambda_s$.
- Step 2: This step is the same as Step 1 of Algorithm SP, except that the queue evolution (4) becomes[3]

$$q_l(t+1) = q_l(t) + \sum_{s=1}^{S}\sum_{j=1}^{J(s)} H_{sj}^l \tilde{P}_{sj}(t)A_s(t) - \sum_{c=1}^{C} y_l^c(t).$$

- Step 3: This step is the same as Step 2 in Algorithm SP.

Remark: The optimization problem (19) is formulated in such a way that the routing fraction P_{sj} is larger for a path j with a smaller congestion cost $\sum_{l=1}^{L} H_{sj}^l q_l(t) + e_{sj}$. Note that each user only needs to know the sum of the queue length q_l along its own paths. An efficient algorithm for solving this optimization problem with $O(J(s)\log J(s))$-complexity is provided in [38]. The parameter e_{sj} is optional and it allows each user to give preference to certain paths. For example, e_{sj} can be equal to a positive constant multiplied by the number of hops along each path j of user s, in which case the paths with a smaller number of hops are given preference. The quadratic term in the objective function of (19) is essential to prevent potential routing oscillation. To see this, note that if $\beta_s = 0$, then when (19) is solved for any user s that has multiple alternate paths, only paths that have the smallest cost $\sum_{l=1}^{L} H_{sj}^l q_l(t) + e_{sj}$ will have positive P_{sj}. This property can easily lead to oscillation of the routing fractions P_{sj} when the queue length q_l is being updated [39]. On the other hand, with the addition of a quadratic term, the objective function of (19) becomes strictly concave. The optimal routing fraction then becomes a continuous function of the queue length q_l. Thus, routing oscillation is eliminated. The parameter β_s can also control how sensitive the routing fraction is with respect to the queue length. By setting β_s to be a larger constant, the routing fraction will become less sensitive to the transient queue dynamics.

[3]Note that here again we use the assumption that packets from each user s are applied to all links l along the path of user s simultaneously. Correspondingly, the choice of the routing probability in (19) is also based on this "virtual" queue.

The following result shows that Algorithm MP can achieve at least $1/(\mathcal{K}+2)$ of the maximum system capacity, compared with a throughput-optimal algorithm using the same set of alternate paths.

Proposition 4: Assume that the set of alternate paths are given. If $\alpha_l = \alpha$ for all links l, then the efficiency ratio of Algorithm MP is $1/(\mathcal{K}+2)$. Furthermore, if $M_i = C$ for all node i, then the efficiency ratio can be improved to $1/\mathcal{K}$.

Remark: The authors of [8] also developed a joint channel assignment, scheduling and routing algorithm, which has been shown to achieve an efficiency ratio of $(\min_i M_i)/(\mathcal{K}C)$ (see [8, Theorem 2]). The algorithm in [8] is an offline and centralized algorithm. In contrast, Algorithm MP that we developed in this paper is distributed and much simpler. Furthermore, when Algorithm MP uses the same set of paths as computed by the Linear-Programming based algorithm in [8], it can guarantee an efficiency ratio of $1/(\mathcal{K}+2)$, which is higher than that of [8] if some network nodes only have a small number of radio interfaces. We refer the readers to the discussions in the Introduction regarding the difference between our work and [8], and the potential implications.

Proof of Proposition 4: We only provide the proof for the efficiency ratio $1/(\mathcal{K}+2)$. We use the same Lyapunov function in (8). Following the steps in the proof of Proposition 2, we can bound the Lyapunov drift as

$$\begin{aligned}
&\mathbf{E}\big[\Delta V(t)|\vec{q}(t),\vec{\eta}(t)\big] \\
&\le \sum_{l=1}^{L}\frac{q_l(t)}{\alpha}\Big[\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}P_{sj}(t)\lambda_s-\sum_{c=1}^{C}x_l^c(t)\Big] \\
&\quad+\sum_{l=1}^{L}\sum_{c=1}^{C}\frac{\eta_l^c(t)}{r_l^c}\Big[\sum_{k\in I_l}\frac{x_k^c(t)}{r_k^c} \\
&\qquad+\frac{1}{M_{b(l)}}\sum_{k\in E(b(l))}\sum_{d=1}^{C}\frac{x_k^d(t)}{r_k^d} \\
&\qquad+\frac{1}{M_{e(l)}}\sum_{k\in E(e(l))}\sum_{d=1}^{C}\frac{x_k^d(t)}{r_k^d}-\mu_l^c(t)\Big]+C_4.
\end{aligned}$$

Since $(\mathcal{K}+2)\vec{\lambda}$ can be supported by the optimal policy, similar to (13) and (14), there must exist some $[\bar{x}_l^c]$ and $[\bar{P}_{sj}]$, such that

$$\begin{aligned}
&0\le\bar{P}_{sj}\le 1,\quad \sum_{j=1}^{J(s)}\bar{P}_{sj}=1,\quad 0\le\bar{x}_l^c\le r_l^c \\
&(1+\epsilon)\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}\bar{P}_{sj}\lambda_s\le\sum_{c=1}^{C}\bar{x}_l^c,\quad \text{for all link } l \\
&(1+\epsilon)\Big[\sum_{k\in I_l}\frac{\bar{x}_k^c}{r_k^c}+\frac{1}{M_{b(l)}}\sum_{k\in E(b(l))}\sum_{d=1}^{C}\frac{\bar{x}_k^d}{r_k^d} \\
&\qquad+\frac{1}{M_{e(l)}}\sum_{k\in E(e(l))}\sum_{d=1}^{C}\frac{\bar{x}_k^d}{r_k^d}\Big]\le 1,\quad \text{for all } (l,c).
\end{aligned}$$

We thus have

$$\begin{aligned}
&\mathbf{E}[\Delta V(t)|\vec{q}(t),\vec{\eta}(t)] \\
&\le\sum_{l=1}^{L}\frac{q_l(t)}{\alpha}\Big[\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}P_{sj}(t)\lambda_s \\
&\qquad-\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}\bar{P}_{sj}\lambda_s\Big] \qquad (20)\\
&\quad+\sum_{l=1}^{L}\frac{q_l(t)}{\alpha}\Big[\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}\bar{P}_{sj}\lambda_s-\sum_{c=1}^{C}\bar{x}_l^c\Big] \qquad (21)\\
&\quad+\sum_{l=1}^{L}\frac{q_l(t)}{\alpha}\Big[\sum_{c=1}^{C}\bar{x}_l^c-\sum_{c=1}^{C}x_l^c(t)\Big] \\
&\quad+\sum_{l=1}^{L}\sum_{c=1}^{C}\frac{\eta_l^c(t)}{r_l^c}\Big[\sum_{k\in I_l}\frac{\bar{x}_k^c}{r_k^c}+\frac{1}{M_{b(l)}}\sum_{k\in E(b(l))}\sum_{d=1}^{C}\frac{\bar{x}_k^d}{r_k^d} \\
&\qquad+\frac{1}{M_{e(l)}}\sum_{k\in E(e(l))}\sum_{d=1}^{C}\frac{\bar{x}_k^d}{r_k^d}-\mu_l^c(t)\Big] \\
&\quad+\sum_{l=1}^{L}\sum_{c=1}^{C}\frac{\eta_l^c(t)}{r_l^c}\Big[\sum_{k\in I_l}\frac{x_k^c(t)-\bar{x}_k^c}{r_k^c} \\
&\qquad+\frac{1}{M_{b(l)}}\sum_{k\in E(b(l))}\sum_{d=1}^{C}\frac{x_k^d(t)-\bar{x}_k^d}{r_k^d} \\
&\qquad+\frac{1}{M_{e(l)}}\sum_{k\in E(e(l))}\sum_{d=1}^{C}\frac{x_k^d(t)-\bar{x}_k^d}{r_k^d}\Big]+C_6
\end{aligned}$$

where C_6 is some constant. Note that except the first two terms (20) and (21), the remaining terms are the same as (15) in the proof of Proposition 2. To show stability, it then suffices to ensure that (20) and (21) have negative drifts. From (20), we have

$$\begin{aligned}
&\sum_{l=1}^{L}q_l(t)\Big[\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}P_{sj}(t)\lambda_s-\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}\bar{P}_{sj}\lambda_s\Big] \\
&\quad=\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\frac{\lambda_s\beta_s(P_{sj}(t))^2}{2} \\
&\qquad+\sum_{l=1}^{L}q_l(t)\Big[\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}P_{sj}(t)\lambda_s\Big] \\
&\qquad-\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\frac{\lambda_s\beta_s(\bar{P}_{sj})^2}{2} \\
&\qquad-\sum_{l=1}^{L}q_l(t)\Big[\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}\bar{P}_{sj}\lambda_s\Big] \\
&\qquad-\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\frac{\lambda_s\beta_s(P_{sj}(t))^2}{2}+\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\frac{\lambda_s\beta_s(\bar{P}_{sj})^2}{2}.
\end{aligned}$$

According to Step 1 of Algorithm MP

$$\begin{aligned}
&\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\frac{\lambda_s\beta_s(P_{sj}(t))^2}{2}\\
&\quad+\sum_{l=1}^{L}q_l(t)\left[\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}P_{sj}(t)\lambda_s\right]\\
&\quad+\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\lambda_s P_{sj}(t)e_{sj}\\
&\le\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\frac{\lambda_s\beta_s(\bar{P}_{sj})^2}{2}+\sum_{l=1}^{L}q_l(t)\left[\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}\bar{P}_{sj}\lambda_s\right]\\
&\quad+\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\lambda_s\bar{P}_{sj}e_{sj}.
\end{aligned}$$

Hence, the term (20) can be bounded by

$$\begin{aligned}
&\sum_{l=1}^{L}\frac{q_l(t)}{\alpha}\left[\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}P_{sj}(t)\lambda_s-\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}\bar{P}_{sj}\lambda_s\right]\\
&\le-\frac{1}{\alpha}\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\lambda_s P_{sj}(t)e_{sj}-\frac{1}{\alpha}\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\frac{\lambda_s\beta_s(P_{sj}(t))^2}{2}\\
&\quad+\frac{1}{\alpha}\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\lambda_s\bar{P}_{sj}e_{sj}+\frac{1}{\alpha}\sum_{s=1}^{S}\sum_{j=1}^{J(s)}\frac{\lambda_s\beta_s(\bar{P}_{sj})^2}{2}\\
&\le\frac{1}{\alpha}\sum_{s=1}^{S}\sum_{j=1}^{J(s)}2\lambda_s|e_{sj}|+\frac{1}{\alpha}\sum_{s=1}^{S}\lambda_s\beta_s J(s).
\end{aligned}$$

On the other hand, the second term (21) is less than

$$-\sum_{l=1}^{L}\frac{\epsilon\sum_{s=1}^{S}\sum_{j=1}^{J(s)}H_{sj}^{l}\bar{P}_{sj}\lambda_s}{\alpha}q_l(t).$$

The rest of the proof then follows along the same line as the proof of Proposition 2. $Q.E.D.$

It is also possible to combine Greedy Maximal Scheduling with Step 1 of Algorithm MP to obtain a multipath version of Greedy Maximal Scheduling. By combining the Lyapunov function in the proof of Proposition 1 (see [31]) with the techniques in the proof of Proposition 4, we can show that the multipath version of Greedy Maximal Scheduling also guarantees an efficiency ratio of $1/(\mathcal{K}+2)$ in general, and an efficiency ratio of $1/\mathcal{K}$ when $M_i = C$ for all node i. In Section V, we will compare the performance of these two distributed algorithms through simulation.

C. How to Generate Alternate Paths

The set of alternate paths, denoted by the matrix $[H_{sj}^{l}]$, could potentially be the enumeration of all possible paths between each source-destination pair. In practice, however, a much smaller set of alternate paths suffices. We next describe options to compute and maintain this set of alternate paths. First, we could use the paths computed by any of the routing algorithms in [2], [6], [8]–[12]. Proposition 4 then ensures that the capacity region of Algorithm MP is no worse that $1/(\mathcal{K}+2)$ times the maximum capacity that can be achieved by using the same set of paths.

A more attractive alternative is to discover paths *online*. Note that the queue-length $q^l(t)$ also provides us with the signal to discover potentially better paths. Given a set of alternate paths, we can easily verify the following property for Step 1 of Algorithm MP. For each user s, those paths j with positive routing fractions P_{sj} must satisfy the following condition:

$$\begin{aligned}
&\beta_s P_{sj}+\sum_{l=1}^{L}H_{sj}^{l}q_l(t)+e_{sj}\\
&=\min_{k=1,\ldots,J(s)}\beta_s P_{sk}+\sum_{l=1}^{L}H_{sk}^{l}q_l(t)+e_{sk}\triangleq q_{s,\min}(t). \quad (22)
\end{aligned}$$

Therefore, adding paths with congestion costs $\sum_{l=1}^{L}H_{sj}^{l}q_l(t)+e_{sj}$ larger than $q_{s,\min}(t)$ will not yield any gain. We can use this property to iteratively generate the candidate paths online. Consider the case when $e_{sj}=0$ for all paths. Starting from any initial set of candidate paths, we execute Algorithm MP for joint routing, channel-assignment and scheduling. Then, every T time-slots (our simulation uses $T=200$), we can run a minimal cost routing algorithm using the queue-length as the cost-metric for each link. If the minimal cost is smaller than $q_{s,\min}(t)$ defined in (22), we add this new path into the set of alternate paths, and continue. Otherwise, we can conclude that no further alternate paths need to be added. In practice, in order to reduce the number of paths added, we may add new paths only when the minimal cost of the new path is smaller than $(1-\delta)\,q_{s,\min}(t)$. Our simulation results in the next section use $\delta=0.25$ and still exhibit satisfactory performance.

D. How to Compute Maximal Schedules

For single-channel systems and under the node-exclusive interference model, efficient algorithms for computing a maximal schedule (which reduces to maximal matching) in $O(\log L)$-time have been provided in [37]. We believe that the idea there can also be generalized to multichannel systems and to more general interference models. Below we sketch one algorithm that can compute a maximal schedule for multichannel systems in logarithmic-time, assuming that the size of the interference sets is bounded, i.e., $|I_l|\le I_{\max}$ for all links l. For ease of exposition, we assume that each node has only one radio interface [because the multiple-interface case can be mapped to the single-interface case by mapping each physical node to multiple copies of logical nodes (see [40])]. Like [37], this algorithm proceeds in phases. Each phase k computes a (not necessarily maximal) schedule from a system G_k. At the initial phase $k=1$, the system G_k is the original network. After each phase k, the system at the next phase G_{k+1} is obtained from G_k by removing those link-channel pairs that either have already been scheduled, or cannot be scheduled any more due to interference constraints or radio interface constraints. The algorithm ends when G_k have no link-channel pairs left, and

the maximal schedule is the union of the schedules computed in all phases.

The algorithm in each phase k proceeds as follows. First, a collapsed graph G'_k is constructed from G_k such that an edge (i,j) is in G'_k if and only if there is a link (i,j) in *any* channel in system G_k. We assume that there exists a separate control channel with which each node can communicate with its adjacent nodes. We then use the algorithm from [37] to compute a (not necessarily maximal) matching in this collapsed graph G'_k. Then, for each edge belonging to the matching, the two end-points (say, i and j) decide on a channel c such that link (i,j) is available for scheduling in channel c in system G_k. The two end-points then tune one radio to channel c. One of the end-points (say i) picks a random backoff time between 1 to $(I_{\max}+1)(I_{\max}+2)$. If node i does not hear any other transmissions in channel c from its interfering links before its own backoff time expires, it sends an RTS (Request-to-Send) to the other end-point (in this case j). Node j then responds with a CTS (Clear-to-Send) if it also has not heard other transmissions in channel c from its interfering links. Using this random-backoff procedure, in each channel any link with a smaller backoff time than all of its interfering links will win. All link-channel pairs that win will be added to the schedule computed in this phase.

We can show that this algorithm has expected time-complexity of $O(\frac{I_{\max}^2}{\log I_{\max}} \log L)$. To see this, note that according to the result in [37], at least a third of the edges in G'_k are "good edges" (readers can refer to [37] for the definition of "good edges"). For each good edge, in each phase with probability at least $\frac{1}{2}(1-e^{-1/3})$ it is either included in the matching computed by the algorithm in [37], or adjacent to another edge in the matching. For each edge in the matching, the probability that its corresponding link can win in the above random-backoff procedure is at least $1/(I_{\max}+2)$ (see [23]). Since we assume that there is only one interface at each node, whenever a link wins, all links that share a common node with this link will be removed from the system. Using these facts, we can then conclude that in each phase, the average fraction of edges that will be removed is bounded from below by a constant that is inversely proportional to $(I_{\max}+2)$. Hence, the expected number of phases to completion is $O(\frac{\log L}{\log I_{\max}})$, and the total time to compute a multichannel maximal schedule is $O(\frac{I_{\max}^2}{\log I_{\max}} \log L)$.

V. Simulation Results

In this section, we evaluate the performance of our proposed algorithms through simulation. Our simulations are based on the network topology shown in Fig. 1. There are 16 nodes (represented by circles) and 24 links (represented by dashed lines). For simplicity, we assume a node-exclusive interference model in our simulations, i.e., only links that share a common node interfere with each other [17], [28]. We also use the simplified equation (1) to model the queue evolution.

We first consider a single-hop scenario. We will compare the performance of the fixed-path Algorithm SP, the Greedy Maximal Scheduling algorithm and the Aggregated Maximal Scheduling algorithm in Section III. We assume that there are 8 available channels in the system, and each node has 8 radios so that

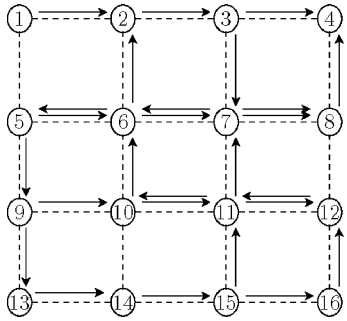

Fig. 1. Network topology.

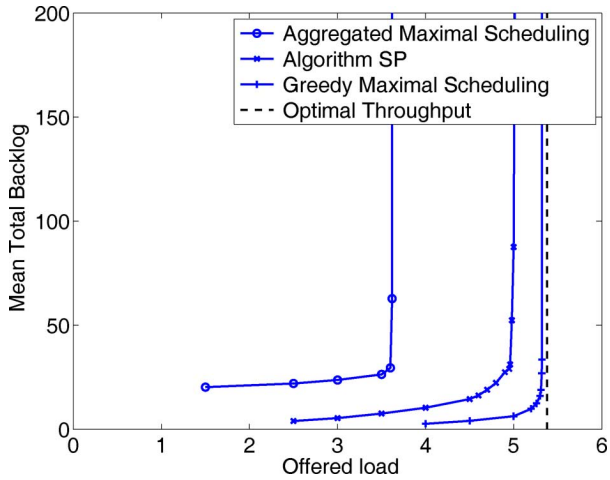

Fig. 2. Comparison of Algorithm SP with Greedy Maximal Scheduling and Aggregated Maximal Scheduling.

it can transmit/receive on 8 channels simultaneously. In our simulation, the capacity r_l^c of each channel for each link is randomly chosen as an integer from 1 to 5, which models channel-diversity. The single-hop flows are represented by arrows in Fig. 1. We let the rate of each flow be λ. Note that although the rates of the flows are the same, the composition of the flows has been chosen to avoid uniform patterns. For Algorithm SP, we have chosen the parameter α_l in (3) to be 100 for all links, so that the per-link queues and the channel-queues are on the same order of magnitude. Furthermore, in order to reduce the queue backlog as much as possible, after computing the multichannel maximal schedule in Step 2 of Algorithm SP, we do schedule additional link-channel pairs that are not backlogged if they do not interfere with the multichannel maximal schedule. In Fig. 2, we plot the mean total backlog in the network divided by the total number of flows as the packet arrival rate λ increases. When λ approaches a certain limit, the mean backlog will increase to infinity. This limit can then be viewed as the boundary of the capacity region under the particular channel-assignment and scheduling algorithm. Furthermore, we modify the rate control algorithm in [41] to compute the highest value of λ that the network can support under the throughput-optimal (and centralized) channel-assignment and scheduling algorithm (2). In Fig. 2, this highest value of λ is shown as the right-most vertical line. We observe from Fig. 2 that both Algorithm SP and Greedy Maximal Scheduling achieve approximately 40% higher capacity than Aggregated Maximal Scheduling, and their capacity on this network topology is fairly close

TABLE II
SIMULATIONS OF 10 RANDOM CASES: THE VALUES IN THE TABLE REPRESENT THE CAPACITY ACHIEVED BY EACH ALGORITHM

Case #	1	2	3	4	5	6	7	8	9	10
Aggregated Maximal Scheduling	3.60	3.78	3.30	3.76	3.48	3.40	3.45	3.78	3.84	3.48
Algorithm SP	5.00	5.16	4.20	5.00	5.06	4.70	4.65	5.20	5.05	4.88
Greedy Maximal Scheduling	5.32	5.28	4.25	5.36	5.36	5.03	5.00	5.68	5.31	5.18
Throughput-Optimal Algorithm	5.321	5.283	4.264	5.360	5.366	5.039	5.000	5.691	5.316	5.198

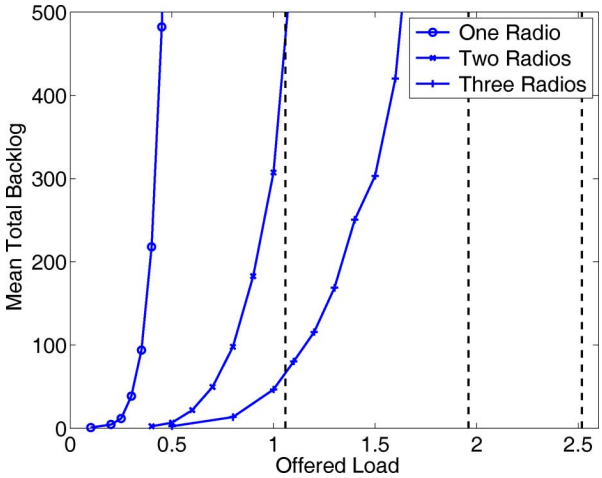

Fig. 3. Performance of Algorithm MP.

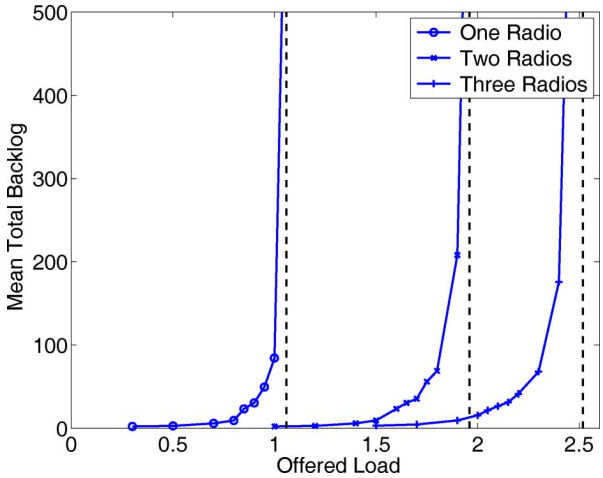

Fig. 4. Performance of the Greedy Maximal Scheduling with multipath routing.

to the optimal capacity. Therefore, by exploiting channel-diversity, our new proposed algorithms can significantly improve the system capacity. We have also randomly simulated ten cases, each with different assignments of the link-channel capacities r_l^c. In Table II we have listed the maximum capacity achieved by Aggregated Maximal Scheduling, Algorithm SP, and Greedy Maximal Scheduling, respectively, and compare with the optimal capacity computed by the rate-control algorithm of [41]. The relative performance of the algorithms are consistent with the observations in Fig. 2.

Our second set of simulations consider the same network topology in Fig. 1, but now with three available channels and with multipath routing. We randomly pick eight source-destination pairs. We then run Algorithm MP with one, two and three radios for each node. As discussed in Section IV-C, in these simulations we maintain at most four alternate paths for each source-destination pair. Every $T = 200$ time-slots, we update the alternate paths according to the online strategy described in Section IV-C. Specifically, we first compute the minimum-cost path using the queue length as the cost metric. (The optional parameter e_{sj} in (22) is chosen to be zero.) Then, the new minimum-cost path is added into the set of alternate paths if its cost is less than 3/4 of $q_{s,\min}$ in (22). If there are already four alternate paths between a source-destination pair, the existing path with the smallest routing fraction is replaced by the new path. We also choose the parameter β_s in (19) to be 1000, so that the routing fractions are not too sensitive to the transient queue updates. In Fig. 3, we plot the mean total backlog in the network divided by the total number of users when there are one, two, and three radios for each node. As expected, the network capacity increases as the number of radios increases. We also show in Fig. 4 the performance of Greedy Maximal Scheduling when used with a multipath routing algorithm as in Step 1 of Algorithm MP. In both figures, we have drawn vertical lines that correspond to the largest values of λ that the network can support with one, two, and three radios per node, respectively. These largest values of λ are computed by the joint rate control and routing algorithm in [41], using the throughput-optimal channel-assignment and scheduling algorithm (2), along with back-pressure based routing. We observe that with multipath routing, the performance of Greedy Maximal Scheduling on this network topology is still quite close-to-optimal. The performance of Algorithm MP is worse, although it is still above the lower bound guaranteed by the efficiency ratios in Proposition 4. (Note that $\mathcal{K} = 2$ under the node-exclusive interference model used in these simulations.)

VI. CONCLUSION

In this paper, we develop fully distributed algorithms that jointly solve the channel-assignment, scheduling and routing problem for multichannel multiradio ad hoc wireless networks. The algorithms that we developed, in particular Algorithm SP and Algorithm MP, are amenable to distributed implementations. They do not require prior information on the offered load to the network, and can thus adapt automatically to the changes in the network topology and offered load. We show that these algorithms are provably efficient. That is, even compared with the optimal centralized and offline algorithm, our proposed algorithms can achieve a provable fraction of the maximum system capacity. Furthermore, the achievable fraction that we can guarantee is larger than that of the centralized and offline algorithm in [8].

The results in this paper have a number of interesting implications to the design of efficient and channel-aware control protocols in broadband wireless networks. Firstly, they provide

additional insights that motivate the study of multichannel control protocols. Note that in many protocol settings, multichannel systems arise because the protocol *a priori* divides the spectrum into multiple channels. One could argue that, if we can redesign the system so that the entire spectrum is used as a single channel, then single-channel protocols will suffice. The results of this paper indicate that this is not always the case. In particular, when there is frequency diversity in the system (e.g., due to frequency-selective multipath fading), it will be more advantageous to divide the spectrum into multiple channels, and to use channel-aware multichannel control algorithms in order to exploit channel diversity. In contrast, if the entire spectrum is used as a single channel, then as illustrated by the example in Section III-B, the system performance can be much worse when there is channel diversity.

A second closely related question is how to best use OFDM. As discussed in the Introduction, OFDM can be viewed as a special case of multichannel systems. Current wireless LAN protocols (such as IEEE 802.11a/g) use OFDM by grouping all subcarriers into one single channel, which again does not fully exploit channel diversity. Perhaps in future protocols, we should consider incorporating subcarrier scheduling capabilities into the protocol.

Third, a main difference between the protocol developed in this paper and that of some earlier work (e.g., [8]) is that we allow the network nodes to switch channel dynamically. Current state-of-art IEEE 802.11 hardware may take up to a few milliseconds to switch channels [2], [42], [43]. In addition, our algorithm requires some protocol overhead to collect local queue-length information, and to instruct the transmitter and receiver nodes which channel(s) they should switch to. However, by allowing this channel-switching capability, the algorithm that we have developed are much simpler and can be shown to guarantee a higher efficiency ratio. Therefore, we believe that our results provide a strong motivation to pursue such improved channel-switching hardwares and protocols in the future.

The solutions in this paper are related to the duality approach for solving some underlying optimization problems that approximate the interference constraints by linear inequality constraints [44]. In particular, the queues $q_l(t)$ and $\eta_l^c(t)$ can be viewed as Lagrange multipliers for the underlying linear constraints. There are other distributed algorithms [45]–[47] that may be used to provide fast solutions to linear programs, such as fractional packing problems. However, these studies do not directly address the problem of how to schedule the links in order to satisfy the interference constraints in wireless networks. Nonetheless, it would still be useful to explore joint channel-assignment, scheduling, and routing algorithms based on the ideas in these works.

In our future work, we plan to study practical issues for implementing the proposed algorithms. Note that in real implementations, additional protocols need to be carefully designed in order to: a) exchange channel-queue-length information between interfering links so that each link can assign traffic to channel queues according to (3); b) feedback the queue-length information to the source so that the source can make routing decisions according to (19); and c) account for actual link-by-link packet dynamics [see the remarks under (1)]. Past studies in the literature have suggested that these protocol overheads could be substantially reduced without significantly affecting the actual performance of the algorithm [19], [20], [23], [48]–[50]. For future work, we plan to carefully address these protocol issues and study their impact on the actual system performance. Finally, the performance gain due to multichannel scheduling will likely depend on the level of channel diversity. How to characterize this gain in realistic settings is also an interesting direction for future work (see [40] for some related results).

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their constructive comments on our paper and Huanren Zhang for obtaining the simulation results in the paper.

REFERENCES

[1] X. Lin and S. Rasool, "A distributed joint channel-assignment, scheduling, and routing algorithm for multi-channel ad hoc wireless network," in *Proc. IEEE INFOCOM*, 2007, pp. 1118–1126.

[2] P. Kyasanur and N. H. Vaidya, "Routing and link-layer protocols for multi-channel multi-interface ad hoc wireless networks," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 10, no. 1, pp. 31–43, Jan. 2006.

[3] P. Kyasanur, J. So, C. Chereddi, and N. H. Vaidya, "Multi-channel mesh networks: Challenges and protocols," *Wireless Commun.*, vol. 13, no. 2, pp. 30–36, Apr. 2006.

[4] C. Chereddi, P. Kyasanur, and N. H. Vaidya, "Design and implementation of a multi-channel multi-interface network," in *Proc. REALMAN*, May 2006, pp. 23–30.

[5] P. Kyasanur and N. H. Vaidya, "Capacity of multi-channel wireless networks: Impact of number of channels and interfaces," in *Proc. ACM Mobicom*, Cologne, Germany, Aug. 2005, pp. 43–57.

[6] J. So and N. H. Vaidya, "Load-balancing routing in multi-channel hybrid wireless networks with single network interface," in *Proc. QShine*, Aug. 2005.

[7] J. So and N. H. Vaidya, "Multi-channel MAC for ad hoc networks: Handling multi-channel hidden terminals using a single transceiver," in *Proc. ACM MobiHoc*, May 2004, pp. 222–233.

[8] M. Alicherry, R. Bhatia, and L. Li, "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *Proc. ACM MobiCom*, Cologne, Germany, Aug. 2005, pp. 58–72.

[9] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *Proc. ACM MobiCom*, Cologne, Germany, Aug. 2005, pp. 73–87.

[10] A. Raniwala and T. Chiueh, "Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 2005, pp. 2223–2234.

[11] A. Raniwala, K. Gopalan, and T. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 8, no. 2, pp. 50–65, Apr. 2004.

[12] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *Proc. ACM MobiCom*, Philadelphia, PA, Sep. 2004, pp. 114–128.

[13] A. Adya, P. Bahl, J. Padhye, A. Wolman, and L. Zhou, "A multi-radio unification protocol for IEEE 802.11 wireless networks," in *Proc. IEEE Broadnets 2004*, San Jose, CA, Oct. 2004, pp. 344–354.

[14] P. Bahl, R. Chandra, and J. Dunagan, "SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks," in *Proc. ACM MobiCom*, Philadelphia, PA, Sep. 2004, pp. 216–230.

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

[16] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic power allocation and routing for time varying wireless networks," in *Proc. IEEE INFOCOM*, San Francisco, CA, Apr. 2003, pp. 745–755.

[17] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer rate control in wireless networks," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 2005, pp. 1804–1814.
[18] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 302–315, Apr. 2006.
[19] X. Wu and R. Srikant, "Scheduling efficiency of distributed greedy scheduling algorithms in wireless networks," in *Proc. IEEE INFOCOM*, Barcelona, Spain, Apr. 2006, pp. 1–12.
[20] X. Wu, R. Srikant, and J. R. Perkins, "Queue-length stability of maximal greedy schedules in wireless network," presented at the Inf. Theory and Appl. Inaugural Workshop, Univ. Calif., San Diego, Feb. 2006.
[21] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput guarantees through maximal scheduling in wireless networks," in *Proc. 43rd Ann. Allerton Conf. Commun., Control and Comput.*, Monticello, IL, Sep. 2005, pp. 1557–1567.
[22] P. Chaporkar, K. Kar, and S. Sarkar, "Achieving queue length stability through maximal scheduling in wireless networks," presented at the Inf. Theory and Appl. Inaugural Workshop, Univ. Calif., San Diego, Feb. 2006.
[23] X. Lin and S. Rasool, "Constant-time distributed scheduling policies for ad hoc wireless networks," in *Proc. IEEE CDC*, San Diego, CA, Dec. 2006, pp. 1258–1263.
[24] X. Wu and R. Srikant, "Regulated maximal matching: A distributed scheduling algorithm for multi-hop wireless networks with node-exclusive spectrum sharing," in *Proc. IEEE CDC*, Seville, Spain, Dec. 2005, pp. 5342–5347.
[25] Y. Liu and E. Knightly, "Opportunistic fair scheduling over multiple wireless channels," in *Proc. IEEE INFOCOM*, San Francisco, CA, Apr. 2003, pp. 1106–1115.
[26] P. Bender, P. Black, M. Grob, R. Padovani, N. Sindhushayana, and A. Viterbi, "CDMA/HDR: A bandwidth-efficient high-speed wireless data service for nomadic users," *IEEE Commun. Mag.*, vol. 38, no. 7, pp. 70–77, Jul. 2000.
[27] X. Liu, E. K. P. Chong, and N. B. Shroff, "A framework for opportunistic scheduling in wireless networks," *Comput. Netw.*, vol. 41, no. 4, pp. 451–474, Mar. 2003.
[28] B. Hajek and G. Sasaki, "Link scheduling in polynomial time," *IEEE Trans. Inf. Theory*, vol. 34, no. 5, pp. 910–917, Sep. 1988.
[29] S. H. Lu and P. R. Kumar, "Distributed scheduling based on due dates and buffer priorities," *IEEE Trans. Autom. Control*, vol. 36, no. 12, pp. 1406–1416, Dec. 1991.
[30] E. Leonardi, M. Mellia, F. Neri, and M. A. Marsan, "On the stability of input-queued switches with speed-up," *IEEE/ACM Trans. Netw.*, vol. 9, no. 1, pp. 104–118, Feb. 2001.
[31] X. Lin and S. Rasool, "A distributed and provably efficient joint channel-assignment, scheduling and routing algorithm for multi-channel multi-radio wireless mesh networks," Purdue Univ., Tech. Rep., 2006 [Online]. Available: http://min.ecn.purdue.edu/~linx/papers.html
[32] J.-H. Hoepman, "Simple distributed weighted matchings," 2004 [Online]. Available: http://arxiv.org/abs/cs.DC/0410047
[33] R. Preis, "Linear time 1/2-approximation algorithm for maximum weighted matching in general graphs," in *Proc. STACS*, C. Meinel and S. Tison, Eds., Trier, Germany, 1999, Lecture Notes in Computer Science, pp. 259–269.
[34] M. Wattenhofer and R. Wattenhofer, "Distributed weighted matching," in *Proc. 18th Annu. Conf. Distrib. Comput. (DISC)*, Amsterdam, The Netherlands, Oct. 2004, pp. 335–348.
[35] S. Ray and S. Sarkar, "Arbitrary throughput versus complexity tradeoffs in wireless networks using graph partitioning," *IEEE Trans. Autom. Control*, vol. 53, no. 10, pp. 2307–2323, Nov. 2008.
[36] K. Jung and D. Shah, "Low delay scheduling in wireless network," in *Proc. IEEE ISIT*, Nice, France, Jun. 2007, pp. 1396–1400.
[37] A. Israeli and A. Itai, "A fast and simple randomized parallel algorithm for maximal matching," *Inf. Process. Lett.*, vol. 22, no. 2, pp. 77–80, Jan. 1986.
[38] X. Lin and N. B. Shroff, "An optimization based approach for quality of service routing in high-bandwidth networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. 6, pp. 1348–1361, Dec. 2006.
[39] X. Lin and N. B. Shroff, "Utility maximization for communication networks with multi-path routing," *IEEE Trans. Autom. Control*, vol. 51, no. 5, pp. 766–781, May 2006.
[40] V. Bhandari and N. H. Vaidya, "Scheduling in multi-channel wireless networks with limited information," Univ. Illinois, Urbana-Champaign, Tech. Rep., 2008.
[41] X. Lin and N. B. Shroff, "Joint rate control and scheduling in multihop wireless networks," in *Proc. IEEE CDC*, Paradise Island, Bahamas, Dec. 2004, vol. 2, pp. 1484–1489.
[42] P. Kyasanur and N. H. Vaidya, "Routing and interface assignment in multi-channel multi-interface wireless networks," in *Proc. IEEE WCNC*, New Orleans, LA, Mar. 2005, pp. 2051–2056.
[43] R. Chandra, R. Mahajan, T. Moscibroda, R. Raghavendra, and P. Bahl, "A case for adapting channel width in wireless networks," in *Proc. ACM SIGCOMM*, Seattle, WA, Aug. 2008, pp. 135–146.
[44] X. Lin, N. B. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1452–1463, Aug. 2006.
[45] N. E. Young, "Randomized rounding without solving the linear program," in *Proc. 6th ACM-SIAM SODA*, San Francisco, CA, 1995, pp. 170–178.
[46] N. Garg and J. Konemann, "Faster and simpler algorithms for multicommodity flow and other fractional packing problems," in *Proc. 39th Ann. Symp. Found. Comput. Sci.*, Palo Alto, CA, Nov. 1998, pp. 300–309.
[47] B. Awerbuch and R. Khandekar, "Stateless distributed gradient descent for positive linear programs," in *Proc. 40th Ann. ACM STOC*, Victoria, BC, Canada, May 2008, pp. 691–700.
[48] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu, "Joint asynchronous congestion control and distributed scheduling for multihop wireless networks," in *Proc. IEEE INFOCOM*, Barcelona, Spain, Apr. 2006, pp. 1–12.
[49] A. Gupta, X. Lin, and R. Srikant, "Low-complexity distributed scheduling algorithms for wireless networks," in *Proc. IEEE INFOCOM*, Anchorage, AK, May 2007, pp. 1631–1639.
[50] C. Joo and N. B. Shroff, "Performance of random access scheduling schemes in multi-hop wireless networks," in *Proc. IEEE INFOCOM*, Anchorage, AK, May 2007, pp. 19–27.

Xiaojun Lin (S'02–M'05) received the B.S. degree from Zhongshan University, Guangzhou, China, in 1994, and the M.S. and Ph.D. degrees from Purdue University, West Lafayette, IN, in 2000 and 2005, respectively.

He is currently an Assistant Professor of Electrical and Computer Engineering at Purdue University. His research interests are resource allocation, optimization, network pricing, routing, congestion control, network as a large system, cross-layer design in wireless networks, mobile ad hoc and sensor networks.

Dr. Lin received the IEEE INFOCOM 2008 Best Paper Award and 2005 Best Paper of the Year Award from the *Journal of Communications and Networks*. His paper was also one of two runner-up papers for the Best Paper Award at IEEE INFOCOM 2005. He received the NSF CAREER Award in 2007. He was the Workshop Co-Chair for IEEE GLOBECOM 2007, the Panel Co-Chair for WICON 2008, and the TPC Co-Chair for ACM MobiHoc 2009.

Shahzada B. Rasool (S'04) received the B.S. (with honors) degree from the University of Engineering and Technology, Lahore, Pakistan, and the M.S. degree from King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, in 2000 and 2005, respectively.

He is currently working toward the Ph.D. degree at the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN. His research interests include analysis of medium access control and scheduling algorithms for wireless networks, and signal design and processing for sensors.