Joint Channel Assignment and Routing for Throughput Optimization in Multiradio Wireless Mesh Networks

Mansoor Alicherry, Randeep Bhatia, and Li Erran Li

***Abstract*—Multihop infrastructure wireless mesh networks offer increased reliability, coverage, and reduced equipment costs over their single-hop counterpart, wireless local area networks. Equipping wireless routers with multiple radios further improves the capacity by transmitting over multiple radios simultaneously using orthogonal channels. Efficient channel assignment and routing is essential for throughput optimization of mesh clients. Efficient channel assignment schemes can greatly relieve the interference effect of close-by transmissions; effective routing schemes can alleviate potential congestion on any gateways to the Internet, thereby improving per-client throughput. Unlike previous heuristic approaches, we mathematically formulate the joint channel assignment and routing problem, taking into account the interference constraints, the number of channels in the network, and the number of radios available at each mesh router. We then use this formulation to develop a solution for our problem that optimizes the overall network throughput subject to fairness constraints on allocation of scarce wireless capacity among mobile clients. We show that the performance of our algorithms is within a constant factor of that of any optimal algorithm for the joint channel assignment and routing problem. Our evaluation demonstrates that our algorithm can effectively exploit the increased number of channels and radios, and it performs much better than the theoretical worst case bounds.**

***Index Terms*—Approximation algorithm, channel assignment, graph theory, mathematical programming, routing, scheduling, wireless mesh networks (WMNS).**

I. Introduction

WIRELESS broadband networks are being increasingly deployed in a multihop wireless mesh network (WMN) configuration. These WMNs are being used on the last mile for extending or enhancing Internet connectivity for mobile clients located on the edge of the wired network. Commercial deployments of multihop WMNs are already in the works. For example, many U.S. cities including Medford, OR, and Chaska, MN, have deployed mesh networks. Even big cities like Philadelphia, PA, are planning to deploy citywide mesh networks. The deployed mesh networks will provide commercial Internet access to residents and local businesses.

Manuscript received March 4, 2005; revised May 1, 2005. This work was supported in part by the National Science Foundation (NSF) NRT under Grant ANI-0335244.

M. Alicherry is with the Network Software Research Department, Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974-0636 USA (e-mail: mansoor@research.bell-labs.com).

R. Bhatia and L. E. Li are with the Networking Research Laboratory, Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974-0636 USA (e-mail: randeep@research.bell-labs.com; erranlli@research.bell-labs.com).

Digital Object Identifier 10.1109/JSAC.2006.881641

In WMNs, the access points (or mesh routers) are rarely mobile and may not have power constraints. In addition, these networks behave almost like wired networks in having infrequent topology changes, limited node failures, etc. Although WMNs may be self-organizing, node additions and maintenance are still rare events. In addition, since each mesh router may aggregate traffic flows for a large number of mobile clients, the aggregate traffic load of each mesh router changes infrequently. In infrastructure wireless mesh networks (IWMNs) [3], some mesh routers are also equipped with a gateway capability through which they interface with the wired network. In such networks, traffic is mainly routed by the WMN wireless backbone between the mesh clients and the wired Internet and goes through the gateway nodes.

One of the major problems facing wireless networks is the capacity reduction due to interference among multiple simultaneous transmissions [11]. In WMNs, providing mesh routers with multiple radios can greatly alleviate this problem. With multiple radios, nodes can transmit and receive simultaneously or can transmit on multiple channels simultaneously. However, due to the limited number of channels available, the interference cannot be completely eliminated, and in addition, careful channel assignment must be done to mitigate the effects of interference. Several companies such as MeshDynamics have recently announced the availability of multiradio mesh network technology.

To make use of commodity 802.11 radios, a channel is assigned to a radio interface for an extended period of time as long as traffic demand or topology does not change. Media access control (MAC) protocols [4] where each radio interface can use different channels on a fast time scale such as on a per-packet basis are not supported in current 802.11 MAC. As observed in [20], assigning the first channel to the first radio, the second channel to the second radio and so on can be far from the optimal achievable performance. In addition, channel assignment and routing are interdependent. This is because channel assignments have an impact on link bandwidths and the extent to which link transmissions interfere. This clearly impacts the routing used to satisfy traffic demands. In the same way, traffic routing determines the traffic flows for each link which certainly affects channel assignments. Channel assignments need to be done in a way such that the communication requirements for the links can be met.

Heuristic approaches on channel assignments and load-aware routing [19], [20] are proposed to improve the aggregate throughput of WMNs and balance load among gateways. These heuristic approaches can still be far from the optimal

performance the network can offer. Because aggregate traffic demands and network topology do not change frequently in IWMNs optimizations using measured traffic demands are feasible. The system management software can compute the optimal channel assignment and routing and configure each elements periodically. Routing protocols will still need to be run to handle topology changes. In this paper, we study the joint channel assignment and routing problem in multiradio IWMNs. Our contributions are as follows.

- We present a formulation for the joint channel assignment, routing, and scheduling problem that can model the interference and fairness constraints and is also able to account for the number of radios at each of the wireless nodes.
- We establish matching necessary and sufficient conditions under which interference-free link communication schedule can be obtained, and we design an efficient algorithm to compute such a schedule.
- We use a novel flow transformation technique to design an efficient channel assignment algorithm that can assign channels to node radios, while ensuring that maximum data can be transmitted on specified traffic routes.
- We establish that our algorithm for the joint channel assignment, routing, and scheduling problem is a constant factor approximation algorithm. To the best of our knowledge, this is the first constant factor approximation algorithm for the problem.
- Our evaluation shows that our algorithm can effectively exploit the increased number of channels and radios, and it performs much better than the worst case theoretical bounds.

II. System Model and Assumptions

This section contains basic definitions and concepts used in the rest of this paper.

A. System Architecture

This work pertains to multihop IWMNs [3], an example of which is shown in Fig. 1. These networks consist of static wireless mesh routers and end mobile clients. The static wireless routers are equipped with traffic aggregation capabilities (e.g., access points) and provide network connectivity to mobile clients within their coverage areas. The wireless mesh routers themselves form a multihop wireless backbone for relaying the traffic to and from the clients. Some of the wireless mesh routers are equipped with gateway functionality to enable connectivity to the wired Internet. All infrastructure resources that the mobile client access (e.g., web servers, enterprise servers, and Internet gateways) reside on the wired Internet and can be accessed via any wireless mesh router with gateway functionality. Thus, the wireless backbone of mesh routers mainly relays mobile clients traffic to and from the Internet via the routers with gateway functionality. Each wireless mesh router may be equipped with multiple wireless interfaces (radios) each operating on a particular channel.

A mesh router u has $I(u)$ wireless interfaces each of which operates on a single channel selected from a set $\mathcal{F}$. Here, $\mathcal{F}$ is a set of orthogonal channels. It may be noted that due to the wireless interference constraints there is no capacity advantage in equipping two different interfaces of a node with the same

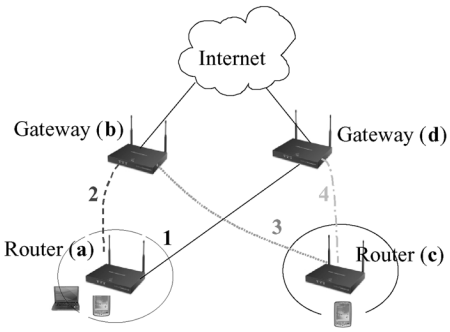

Fig. 1. A WMN with four nodes. Two of them, b and d are gateways. Each node has two interfaces each operating on a different channel among $1, 2, 3, 4$. Edge labels represent the channels used.

channel. Thus, each wireless node u can be associated with a ordered set $F(u) \subseteq \mathcal{F}$ of $I(u)$ distinct channels, where the ith interface of node u operates on the ith channel in $F(u)$. Each node u in V aggregates the user traffic from all the mesh clients that are associated with u. We denote the aggregate user traffic load on u by $l(u)$. The load $l(u)$ may be due to outgoing or incoming traffic. *However, for ease of exposition from now on we will assume that there is no incoming traffic to any wireless node from the wired Internet, and hence $l(u)$ represents only outgoing traffic*. Our results easily extend (by flow reversal) to the more general case of both outgoing and incoming traffic. We assume $l(u)$ exhibits only long term variability and any such variations can be dealt with by rerouting and readjustment of the channel assignments. Thus, we assume $l(u)$ to be a node dependent fixed value.

B. Wireless Transmission and Interference Model

For direct communication, two nodes need to be within *communication range* of each other, and need to have a common channel assigned to their interfaces. A pair of nodes that use the same channel and are within *interference range* may interfere with each other's communication, even if they cannot directly communicate. Node pairs using different channels can transmit packets simultaneously without interference. For example, in Fig. 1, each node is equipped with two Network Interface Cards. The links shown between the nodes depict direct communication between them, and the channel used by a pair of nodes is shown as the number associated with the connecting link. This example network totally uses four distinct channels.

We denote by R_T the *transmission range* and by $d(u, v)$ the distance between the nodes u and v. An edge $(u, v) \in E$ if and only if $d(u, v) \leq R_T$ and implies that mesh router u can communicate with mesh router v directly (in one hop). However, such a communication is only possible if there is a common channel among the sets $F(u)$ and $F(v)$. We assume that, the channel to radio assignment $F(u)$ for each node u is fixed as long as the traffic demands do not change. In other words, we assume there is no system or hardware support to allow a radio to switch channels on a per-packet basis. We denote by $c(e)$ the rate for edge $e = (u, v)$. This is the maximum rate at which mesh router u can communicate with mesh router v in one hop on a single channel. An edge can support multiple simultaneous communications of rate $c(e)$ each one for every channel in

common among the sets $F(u)$ and $F(v)$. Each such communication can be uniquely identified with one channel in common among the sets $F(u)$ and $F(v)$. For notational convenience, we will use $F(e)$ to denote the common channels among $F(u)$ and $F(v)$. Thus, for an edge $e = (u, v)$, k simultaneous link transmissions each of rate $c(e)$ are possible from node u to node v if there are k channels in $F(e)$. We denote by R_I the *interference range*. We assume that R_I is $q \times R_T$, where $q \geq 1$. We assume that 802.11 MAC protocol, carrier sense multiple access (CSMA) with ready-to-send/clear-to-send/acknowledgment (RTS/CTS/ACK) is used to protect unicast transmissions. Thus, as a result of carrier sensing, a transmission between u and v may block all transmissions within R_I away from either u (due to sensing RTS and DATA) or v (due to sensing CTS and ACK). In particular, simultaneous link transmissions on a common channel f on two distinct edges $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ is possible if and only if none of the four pairs of nodes $(u_1, u_2), (v_1, u_2), (u_1, v_2), (v_1, v_2)$ are at most R_I apart. *In this case, we say that edges e_1 and e_2 do not interfere. Otherwise, the two edges interfere with each other.* We denote by $I(e) \subseteq E$ the set of edges that interfere with an edge $e \in E$. Note that simultaneous link transmissions on two edges e_1 and e_2 that interfere is still possible, as long as these transmissions are on distinct channels. We would like to note that our results also extend to other commonly used interference models including the Protocol Model [11] which are based on certain geometric properties.

C. Assumptions on Scheduling, Routing and Fairness

For the algorithm presented in this paper, we assume that the system operates synchronously in a time-slotted mode. The throughput we obtain will provide an upper bound for systems using 802.11 MAC. We assume traffic between a node and the gateway nodes is routed on multiple paths to achieve the optimal load balancing and least congestion for the given WMN. Our goal is to maximize the capacity of the network for serving mesh clients. This capacity may not be measured by the total throughput of all mesh clients. Optimizing such a metric may lead to starvation of mesh clients which are far from gateways. We therefore need to consider fairness constraints to prevent such starvation. In this paper, we first focus on the fairness constraint that, for each node $u \in V$, demands be routed in proportion to its aggregate user traffic load $l(u)$. In other words, we consider the problem of maximizing λ such that $\lambda l(u)$ fraction of each node u's demand can be routed. For this problem, the fraction of demand $\lambda(u)$ that can be routed is the same λ for each node. Note that, the nodes in V correspond to wireless routers. Our solution works with no change if V also includes mesh clients.

III. Problem Formulation and Algorithm Overview

Formally, we are given a wireless mesh backbone network modeled as a graph (V, E). The node $t \in V$ represents the wired network. The set $V_G \subseteq V$ represents the set of gateway nodes. We have a total of K channels. Each node $u \in V$ has $I(u)$ network interface cards, and has an aggregated demand $l(u)$ from its associated users. For each edge e, the set $I(e) \subset E$ denotes the set of edges that it interferes with. We seek to maximize λ, where at least $\lambda l(u)$ amount of throughput can be routed from each node u to the Internet (represented by a node t). In order to achieve $\lambda l(u)$ throughput for each node u, we need to compute: 1) a network flow that associates with each edge $e = (u, v)$ values $f(e(i)), 1 \leq i \leq K$, where $f(e(i))$ is the rate at which traffic is transmitted by node u for node v on channel i; 2) a feasible channel assignment $F(u)$ (recall that $F(u)$ is an ordered set, where the ith interface of u operates on the ith channel in $F(u)$) such that, whenever $f(e(i)) > 0$, $i \in F(u) \cap F(v)$; in this case, we say edge e uses channel i; and 3) a *feasible* schedule S that decides the set of edge channel pair (e, i) (edge e using channel i) scheduled at time slot τ, for $\tau = 1, 2, \ldots, T$, where T is the period of the schedule. A schedule is feasible if the edges of no two edge pairs $(e_1, i), (e_2, i)$ scheduled in the same time slot for a common channel i interfere with each other ($e_1 \notin I(e_2)$ and $e_2 \notin I(e_1)$). Thus, a feasible schedule is also referred to as an interference-free edge schedule. We use an indicator variable $X_{e,i,\tau}, e \in E, i \in F(e), \tau \geq 1$ which is assigned 1 if and only if link e is active in slot τ on channel i. Note that $(1/T) \sum_{1 \leq \tau \leq T} X_{e,i,\tau} c(e) = f(e(i))$. This is because communication at rate $c(e)$ happens in every slot that link e is active on channel i and since $f(e(i))$ is the average rate attained on link e for channel i. This implies $(1/T) \sum_{1 \leq \tau \leq T} X_{e,i,\tau} = (f(e(i))/c(e))$.

Even the interference-free edge scheduling subproblem given the edge flows is NP-hard [17]. We present an approximation algorithm for the overall problem. We refer to this algorithm as the joint routing, channel assignment, and link scheduling (RCL) algorithm. We now give an overview of the RCL algorithm. The algorithm performs the following five steps in the given order.

1) **Solve LP**: We first optimally solve a LP relaxation of the problem This results in a flow on the flow graph along with a not necessarily feasible channel assignment for the node radios. Specifically, a node may be assigned more channels than the number of its radios. However, this channel assignment is "optimal" in terms of ensuring that the interference for each channel is minimum. This step also yields a lower bound on the λ value which we use in establishing the worst case performance guarantee of the overall algorithm.
2) **Channel Assignment**: In this step, we present a channel assignment algorithm which is used to adjust the flow on the flow graph (routing changes) to ensure a feasible channel assignment. This flow adjustment also strives to keep the increase in interference for each channel to a minimum.
3) **Post Processing and Flow Scaling**: In this step, the flow on the flow graph is readjusted (routing changes) to ensure that the maximum interference over all channels is minimized. Next, the flow is scaled to eliminate all interference for all channels, thus yielding a feasible routing and channel assignment.
4) **Interference-Free Link Scheduling**: In this step, for the edge flows corresponding to the flow on the flow graph, we obtain an interference-free link schedule.

In subsequent sections, we will first describe each step in detail. We then present the analysis on the algorithm and show that it achieves a constant factor approximation. We use the example network shown in Fig. 1 to illustrate the steps of our algorithm.

A. A Linear Programming Based Routing Algorithm

In this section, we show how to efficiently find such a routing that also satisfies all the necessary constraints for the joint channel assignment, routing, and interference-free link scheduling problem (which will be discussed in subsequent sections).

We now formulate a linear program (LP) to find a flow that maximizes λ. The LP (LP1) is given below

$$\max \lambda \tag{1}$$

Subject to

$$\lambda l(v) + \sum_{e=(u,v)\in E} \sum_{i=1}^{K} f(e(i)) = \sum_{e=(v,u)\in E} \sum_{i=1}^{K} f(e(i)), \quad \forall v \in V - V_G \tag{2}$$

$$f(e(i)) \le c(e), \quad \forall e \in E \tag{3}$$

$$\sum_{1\le i\le K} \left(\sum_{e=(u,v)\in E} \frac{f(e(i))}{c(e)} + \sum_{e=(v,u)\in E} \frac{f(e(i))}{c(e)} \right) \le I(v), \quad v \in V \tag{4}$$

$$\frac{f(e(i))}{c(e)} + \sum_{e'\in I(e)} \frac{f(e'(i))}{c(e')} \le c(q), \quad \forall e \in E, \quad 1 \le i \le K. \tag{5}$$

The first two constraints are *flow constraints*. The first one is the flow conservation constraint; the second one ensures no link capacity is violated. The third constraint is the *node radio constraints*. Recall that a WMN node $v \in V$ has $I(v)$ radios, and hence can be assigned at most $I(v)$ channels from $1 \le i \le K$. One way to model this constraint is to observe that due to interference constraints v can be involved in at most $I(v)$ simultaneous communications (with different one hop neighbors). In other words, this constraint follows from $\sum_{1\le i\le K} \sum_{e=(u,v)\in E} X_{e,i,\tau} + \sum_{1\le i\le K} \sum_{e=(v,u)\in E} X_{e,i,\tau} \le I(v)$. The fourth constraint is the *link congestion constraints*, which we will discuss in detail in Section VI. Note that all the constraints listed above are necessary conditions for any feasible solution. However, these constraints are not necessarily sufficient. Hence, if a solution is found that satisfies these constraints, it may not be a feasible solution. Our approach is to start with a "good" but not necessarily feasible solution that satisfies all of these constraints and use it to construct a feasible solution without impacting the quality of the solution. A solution to this LP can be viewed as a flow on a *flow graph* $H = (V, E^H)$ where $E^H = \{e(i) | \forall e \in E, 1 \le i \le K\}$. Although the optimal solution to this LP yields the best possible λ (say λ^*) from a practical point of view, some more improvements may be possible.

- The flow may have directed cycles. This may be the case since the LP does not try to minimize the amount of interference directly. By removing the flow on the directed cycle (equal amount off each edge), flow conservation is maintained, and in addition since there are fewer transmissions, the amount of interference is reduced.
- Flow may be going on long paths when shorter paths are available. Note that longer paths imply more link transmissions. In this case, many times by moving the flow to shorter paths, system interference may be reduced.

The above arguments suggests that it would be practical to find among all solutions that attain the optimal λ value of λ^*, the one for which the total value of the following quantity is minimized:

$$\sum_{1\le i\le K} \sum_{e=(v,u)\in E} \frac{f(e(i))}{c(e)}.$$

We thus resolve the LP with this objective function and with λ fixed at λ^*. We refer this new LP as LP2.

The optimal solution to LP2 is a flow on the flow graph H that maximizes λ, satisfies all of the above mentioned constraints, and also tries to minimize the maximum interference per channel.

We now illustrate the routing step using the four-node example network shown in Fig. 1. Suppose there are a total of four channels and each node has two radio interfaces. Suppose the edge capacities are 1 unit each and $q = 2$ ($C_q = 8$). We assume that the interference range is large enough so that only one node can transmit at any given time on a given channel. We assume both nodes a and c have a demand of 2 units each. Let us denote $e_1 = (a, b)$, $e_2 = (a, d)$, $e_3 = (c, b)$, and $e_4 = (c, d)$. After completing the routing step, we may have the following set of edge flows as an optimal solution to the LP-based routing algorithm: $f(e_j(i)) = 1/4$, $\forall i, j = 1, 2, 3, 4$. It is easy to see that this edge flows satisfies all the linear constraints in LP1 and is optimal. However, this flow is not feasible to the original throughput optimization problem since the channel assignment is not feasible (two radios utilize four channels).

IV. A Channel Assignment Algorithm

In this section, we present a channel assignment algorithm that operates on a flow on the flow graph H for a traffic routing that satisfies the loads $\lambda^* l(v)$ at all nodes $v \in V$. Although this given flow satisfies the link congestion constraints (5) for all the channels, the *induced* channel assignment may not be feasible. The channel assignment algorithm transforms the given flow to fix this infeasibility. In other words, it ensures that for each v, the number of channel i such that $f(e(i)) < 0, \forall e \in E$, and e incident on v, is no more than $I(v)$. When we scale the resulting flow of the channel assignment step, it is at least a $\phi = (c(q)K)/(I)$ fraction of the original flow, and hence satisfies at least the load $l(v)\lambda/\phi$ at all nodes $v \in V$. The scaled flow also satisfies the link schedulability constraint [(11) discussed later in Section VI] for all channels, thus implying an interference-free schedule for each channel can be obtained. We can assume without loss of generality (proof omitted) that $f(e(i)) > 0$ for some channel i for every edge $e \in E$. The algorithm works in phases and on termination ensures that the number of channels assigned to any node v is at most the number of its interfaces $I(v)$. The basic operation used by the algorithm is to move some flow from edge $e(i)$ to edge $e(j)$ for some link $e = (u, v) \in E$ and distinct channels i and j. This flow adjustment step disregards edge capacities, however, any potential capacity violations are corrected by subsequent flow scaling. The algorithm

also strives to minimize the interference for each channel. We use the following expression, which forms the basis of a necessary and sufficient condition for interference-free link schedulability (see Section VI), for measuring the interference on a link $e \in E$ for a given channel i:

$$\mathrm{Int}(e,i) = \frac{f(e(i))}{c(e)} + \sum_{e' \in I(e)} \frac{f(e'(i))}{c(e')}. \tag{6}$$

Based on this expression, we define the interference on a channel i as

$$\mathrm{Int}(i) = \max_{e \in E} \mathrm{Int}(e,i). \tag{7}$$

1) Algorithm Intuition: The main idea behind the algorithm is the observation that channel assignment is trivial if the number of available channels is at most the minimum number of radios per node (I). Since in this case every node must be assigned all the channels. The algorithm thus initially operates with I channels. However, instead of assigning one channel to a single interface of a node u, it assigns approximately $I(u)/I$ channels per interface. To this end, the algorithm first transforms the given network (by creating approximately $I(u)/I$ copies of each node) and reassigns the edge flows so that after the transformation every node has between I and $2I$ radios and the edge flows still satisfy the node radio constraint (5) in the transformed network. This is done in Phase 1 of the algorithm. Next, the algorithm assigns channels $1, 2, \ldots, I$ to every node in the new network. At this point, the network may be very imbalanced in terms of the interference within each channel. The algorithm, therefore, attempts to spread the interference uniformly so that within each channel the maximum interference is bounded. In addition, the algorithm tries to minimize the interference within connected components of each channel. Here, the connected components are formed by the edges that have positive flows on the given channel. This is done in Phase 2 of the algorithm. So far, the algorithm has only used I out of K channels and may have assigned a channel to multiple node interfaces. Next, the algorithm tries to change the channel assignment so that all K channels are used and only one channel is assigned per node interface. The algorithm does this channel reassignment to further reduce the intrachannel interference. This is done by forming groups of connected components consisting of connected components from the same channel. All the edges (and their end-nodes) belonging to each group are then assigned a common channel (from 1 to K) in order to minimize the maximum interference within each channel. At this point, all available channels are effectively utilized by the algorithm. The algorithm then maps this channel assignment and flow solution back to the original network. All this is done in Phase 3 of the algorithm.

We now formally describe the channel assignment algorithm. We say a node v is assigned to channel i if there exists an edge e incident on node v for which $f(e(i)) > 0$. Let $I \geq 1$ denote the minimum number of radios at each wireless node. Let $\mu(e)$ denote the "aggregate fractional flow" on an edge e

$$\mu(e) = \sum_{1 \leq i \leq K} \frac{f(e(i))}{c(e)}.$$

Let $\mu(v)$ denote the total "aggregate fractional flow" on the edges incident on node v

$$\mu(v) = \sum_{e=(u,v) \in E} \mu(e) + \sum_{e=(v,u) \in E} \mu(e).$$

The algorithm operates in three phases. In the first phase, the given network $G = (V, E)$ is transformed into a network $G' = (V', E')$ so that all nodes in G' have approximately I (between I and $2I - 1$) radios each and so that the total aggregate fractional flow (as determined by the routing step) on the edges incident on a node $v \in V'$ is at most $I(v)$. The network G' is created as follows. Any node $v \in V$ with $Ir_1 + r_2$ radios, for $r_1 \geq 2$ and $I > r_2 \geq 0$, is replaced by r_1 nodes in G'. All these nodes except at most one has I radios and the one exceptional node has $I + r_2$ radios. Let these nodes (in V') be denoted by $v_1, v_2 \ldots v_{r_1}$. Next, the edges incident on node $v \in V$ are distributed among these nodes so as to assign approximately the same fractional flow $\mu(v_i)$ to all nodes v_i. This is done while maintaining the constraint that $\mu(v_i) \leq I(v_i)$ for all v_i. In this step, the algorithm iterates over the edges incident on node v. When considering an edge e, let v_i denote the node with minimum current value for $\mu(v_i)$ such that $\mu(v_i) < I(v_i)$. In case $I(v_i) - \mu(v_i) \geq \mu(e)$, then in G' edge e is made incident on node v_i. Otherwise, a new copy e' of edge e is created. We set $\mu(e') = \mu(e) - (I(v_i) - \mu(v_i))$, and then set $\mu(e) = I(v_i) - \mu(v_i)$. Edge e is made incident on node v_i and e' is the next edge considered in the edge iteration by the algorithm. Pseudocode for Phase 1 is given in Algorithm 1.

Algorithm 1: Phase 1—Channel Assignment Algorithm

Input: Network $G = (V, E)$ with $\mu(e), e \in E$ and $\mu(v), v \in V$
  $V' = \emptyset, E' = E$
  for $v \in V$
    Let $I(v) = Ir_1 + r_2, 0 \leq r_2 < I$
      $V' = V' \cup \{v_1, v_2, \ldots v_{r_1}\}$
      $I(v_1) = I + r_2, I(v_k) = I, 2 \leq k \leq r_1$
    $\mu(v_i) = 0, \forall 1 \leq i \leq r_1$
    Stack $Q = \{e \in E', e = (u,v) \text{ or } e = (v,u)\}$.
    While $Q \neq \emptyset$
      Pop Stack Q *to get edge* $e = (u,v)$
      Let v_i *has minimum* $\mu(v_i)$ *and* $\mu(v_i) < I(v_i)$
      $\mu = \min\{\mu(e), I(v_i) - \mu(v_i)\}$
      if $\mu < \mu(e)$
        Copy edge e *to* e', *Set* $E' = E' \cup \{e'\}$
        $\mu(e') = \mu(e) - \mu$
        Push e' *on top of Stack* Q
      $\mu(v_i) = \mu(v_i) + \mu$
      $e = (u, v_i)$, $\mu(e) = \mu$

Note that at the end of Phase 1 of the algorithm, the number of radios for each node is in the range I to $2I - 1$. In Phase

2 of the algorithm, the node radios are assigned channels between 1 and I such that the channel interference $\mathrm{Int}(i)$ is at most $(Kc(q))/(I)$, for all channels i. The factor $(Kc(q))/(I)$ is the best we can hope for given that the flow satisfies the constraint (5), and hence $\mathrm{Int}(e,i) \leq c(q)$ for every edge e and every $1 \leq i \leq K$ channels. Thus, the total interference over all K channels for edge e is $\sum_{1\leq i\leq K} \mathrm{Int}(e,i) \leq Kc(q)$. Hence, when we restrict to only I channels, the interference on some channel for edge e may be as large as $(Kc(q))/(I)$. In addition, the goal of Phase 2 is to try to have a large number of connected components, with small intracomponent interference, among the set of edges e with $f(e(i)) > 0$ for every channel i. This property is useful for Phase 3 of the algorithm. In the following, for ease of presentation, we denote by $G = (V, E)$ also the network output by Phase 1 with aggregate fractional flow values $\mu(e), e \in E$ and $\mu(v), v \in V$, and number of radios $I(v), v \in V$. For a given channel and flow assignment $f(e(i))$, let A be a connected component of the network formed by edges $e \in E$ with $f(e(i)) > 0$. We denote by $\mathrm{Int}(e,i,A)$ the interference on edge $e \in A$ for channel i by only considering the edges in A. We can then define the interference $\mathrm{Int}(i,A)$ for component A for channel i as the maximum value of $\mathrm{Int}(e,i,A)$ over all edges e in A. Finally, we can define the component interference $\mathrm{CompInt}(i)$ for channel i as the maximum value of $\mathrm{Int}(i,A)$ for all connected components A for the network formed by edges $e \in E$ with $f(e(i)) > 0$. The algorithm starts out with an empty channel assignment: it sets $f(e(i)) = 0$ for all edges e and channels i. The algorithm iterates over the nodes of the network in nonincreasing order of the $\mu(v)$ value. When considering a node v, it iterates over the unconsidered edges e incident on node v in the nonincreasing order of $\mu(e)$ values. When considering an edge e, the algorithm makes I copies of edge e: $e(1), e(2), \ldots e(I)$ and partitions the total edge e flow $\mu(e)c(e)$ among these I copies as follows. For each channel $i, 1 \leq i \leq I$, the algorithm independently computes the maximum possible flow increase $\gamma(e(i)) \leq c(e)$ on edge $e(i)$ such that the resulting total flow on edges $e(k)$ for all channels $k, 1 \leq k \leq I$ is at most $\max\{c(e), \mu(e)c(e)\}$ and such that for this new flow the channel i interference $\mathrm{Int}(i)$ does not exceed $(Kc(q)/I)$. Let I' be the set of channels $i, 1 \leq i \leq I$ for which $\gamma(e(i)) > 0$. If $I' = \emptyset$, the algorithm proceeds to consider the next edge. Otherwise, let γ be the minimum value of $\gamma(e(i))$ among channels $i \in I'$. Let $k \in I'$ be a channel for which an increase in the flow $f(e(k))$ by γ results in the minimum (among all the channels in I') component interference $\max_{i\in I'} \mathrm{CompInt}(i)$. The algorithm increments $f(e(k))$ by $\gamma(e(k))$ for channel k. The above is repeated for edge e until (as mentioned above) $I' = \emptyset$. Algorithm 2 depicts Phase 2.

Algorithm 2: Phase 2—Channel Assignment Algorithm

Input: Network $G = (V, E)$ with Aggregate Fractional Flow Values

$\mu(e), e \in E$ and $\mu(v), v \in V$

$f(e(i)) = 0, \forall e \in E$ *and* $1 \leq i \leq I$.

$C = \emptyset$/* *Set of edges considered so far* */

for $v \in V$ *in non-increasing order of* $\mu(v)$

for $e \notin C$, e *incident on* v, *in nonincreasing order of* $\mu(e)$

Add e *to* C

while(true)

For all $1 \leq i \leq I$

Compute $\gamma(e(i)) \leq c(e)$, *the maximum possible*

flow increase on $e(i)$ *such that* $\sum_{k=1}^{I} f(e(k)) + \gamma(e(i)) \leq \mu(e)c(e)$ *and* $\mathrm{Int}(i) \leq (Kc(q))/(I)$ *even with* $f(e(i))$ *increased by* $\gamma(e(i))$

Let $I' = \{i \,|\, \gamma(e(i)) > 0\}$

If $I' = \emptyset$ *then break*

Let $\gamma = \min_{i\in I'} \gamma(e(i))$

For $j \in I'$, *let* $\nu(j) = \max_{i\in I'} \mathrm{CompInt}(i)$ *when*

$f(e(j))$ *increased by* γ

Let $\nu(k)$ *be minimum among* $\nu(j), j \in I'$

$f(e(k)) = f(e(k)) + \gamma(e(k))$

Note that in the channel assignment obtained in Phase 2 nodes (each of which has at least I radios) are assigned at most I channels $(1, 2, \ldots, I)$ each. In Phase 3 of the algorithm, the channel assignment is further modified such that each node is still assigned at most I channels. However, these channels may range anywhere from 1 to K now.

For the channel and flow assignment $f(e(i))$ that results from Phase 2, consider a connected component A of the network formed by edges $e \in E$ with $f(e(i)) > 0$ for some channel i. Note that all nodes in A are assigned channel i in this channel assignment. We say A is assigned channel i in this channel assignment. Consider reassigning channel $k \neq i$ to A. This entails moving $f(e(i))$ flow from edge e's copy $e(i)$ to copy $e(k)$ for all edges e in A. Note that after this transformation all edges e incident on node $u \in A$ have $f(e(i)) = 0$. Thus, after the reassignment, no node $u \in A$ is assigned channel i anymore but is assigned channel k. Thus, the number of distinct channels assigned to any node does not increase with this reassignment.

If after Phase 2 of the algorithm there are at most K connected components A among all channels $(1, 2, \ldots, I)$, then in Phase 3 each of the connected components is assigned one of the K distinct channels. Otherwise, the connected components within the channels are grouped to make K groups. This grouping is done as follows. Initially, each connected component is in a group of its own. Analogous to the interference $\mathrm{Int}(i,A)$ within a component A for channel i, we can compute $\mathrm{Int}(i,P)$ the interference within a group P for channel i, and we can define $\mathrm{GroupInt}(i)$ as the maximum value of $\mathrm{Int}(i,P)$ for all groups P in channel i. The algorithm greedily merges pairs of groups belonging to the same channel to a single group such that the merging causes the least increase in $\max_{i=1}^{I} \mathrm{GroupInt}(i)$ and until there are K

groups. The connected components of the ith group are then assigned channel i for $1 \leq i \leq K$.

In the last step of Phase 3, the channel and flow assignment is mapped back to the original network G and its flow graph H. Recall that in Phase 1 an edge e may have been split into multiple edges. Thus, after the channel assignment in Phase 3, multiple copies (say $e_1, e_2, \ldots e_m$) of edge e may have positive flow in a given channel i. The flow on edge $e(i)$ is then set as the sum total $f(e(i)) = \sum_{k=1}^{m} f(e_k(i))$. Algorithm 3 depicts Phase 3.

Algorithm 3: Phase 3—Channel Assignment Algorithm

Input: Network $G' = (V', E')$ with $f(e(i))$ values for all channels

$i \leq I$ and original network $G = (V, E)$

Let $\theta(i) = \{A \mid A$ *is a connected component assigned channel* $i\}$

while$(\sum_{i=1}^{I} |\theta(i)| > K)$

Let $\eta_1, \eta_2 \in \theta(i)$ *such that*

Removing the groups η_1, η_2 *and adding the group* $\eta_1 \cup \eta_2$

to channel i *causes least increase in* $\max_{i=1}^{I} \mathrm{GroupInt}(i)$

Remove η_1, η_2 *from* $\theta(i)$, *Add* $\eta_1 \cup \eta_2$ *to* $\theta(i)$

$\theta = \cup\theta(i)$

Assign channel i *to the* ith *group in* θ

For all $e \in E$ *and* $1 \leq i \leq K$

Let $e_1, e_2, \ldots e_m \in E'$ *correspond to edge* e,

$f(e(i)) = \sum_{k=1}^{m} f(e_k(i))$

Now, let us continue our RCL algorithm on the four-node example network from the routing step. Since all four nodes have the same number of radios, Phase 1 is not needed. Now, let us consider Phase 2. From the routing step, we have $\mu(e_i) = 1, \forall i = 1, 2, 3, 4$ and $\mu(v) = 2, \forall v = a, b, c, d$. Since $\mu(e_i)$ and $\mu(v)$ are all the same, the algorithm picks nodes in the order a, b, c, d and edges in the order e_1, e_2, e_3, e_4. Note that for the edges e_1, e_2 that are incident on node a, $\gamma(e_1(i)) = \gamma(e_2(i)) = \mu(e_1) = \mu(e_2)$, for $i = 1, 2 (\mathrm{Int}(i) = 1 < KC_q/I = 2C_q)$. Therefore, the algorithm sets $f(e_1(1)) = 1$. For the second iteration of the while loop for edges incident on a, e_2 causes the least intracomponent interference if assigned channel 2 ($\mathrm{CompInt}(1) = 2, \mathrm{CompInt}(2) = 1$). Thus, the algorithm sets $f(e_2(2)) = 1$. Similarly, the algorithm sets $f(e_3(2)) = 1$ since e_3 would increase the component interference for channel 1 more than channel 2. Finally, we have $f(e_4(1)) = 1$. Note that there are four connected components in total after Phase 2: $\{e_1\}, \{e_4\}$ corresponding to channel 1 and $\{e_2\}, \{e_3\}$ corresponding to channel 2. Since $K = 4$, Phase 3 assigns each edge e_i a separate channel i. Thus, the only nonzero edge flows are $f(e_1(1)) = f(e_2(2)) = f(e_3(3)) = f(e_4(4)) = 1$. This implies that nodes a, b, c, d are assigned channel pairs $(1, 2), (1, 3), (3, 4), (2, 4)$ respectively. Note that, this flow is already feasible with maximum interference 1. Thus, the flow scaling step in 5 has no effect.

V. Post Processing and Flow Scaling

A. Post Processing

In this step of the algorithm, the aim is to reduce the maximum interference for the K channels. This is done by redistributing for each edge $e = (u, v) \in E$ the flows among its copies $e(i)$ for which $f(e(i)) > 0$, subject to the constraint that the total flow on the copies $\sum_{1 \leq i \leq K} f(e(i))$ does not change. Note that this redistribution does not have any effect on the feasibility of the flow or the channel assignment. The latter is due to the fact that if an end-node u of edge e is not assigned channel i before this step, then $f(e(i)) = 0$ after the step, and hence node u is not assigned channel i after this step. Thus, the number of channels assigned to a node can only decrease after this step.

We formulate the flow redistribution problem as a LP shown below. Here, we denote the total flow assigned to the copies of edge e by the previous step by $\rho(e)$

$$\min \beta \tag{8}$$

$$\begin{aligned}
&\text{Subject to} \\
&f((s, s_v)) = \lambda^* l(v), \forall v \in V \\
&f(e(i)) \leq \beta c(e), \quad \forall e \in E,\ 1 \leq i \leq K \\
&f(e(i)) = 0, \quad \forall e = (u, v) \in E, \quad \forall i \notin F(u) \cup F(v) \\
&\sum_{1 \leq i \leq K} f(e(i)) = \rho(e), \quad \forall e = (u, v) \in E \\
&\frac{f(e(i))}{c(e)} + \sum_{e' \in I(e)} \frac{f(e'(i))}{c(e')} \leq \beta, \quad \forall e \in E,\ 1 \leq i \leq K
\end{aligned} \tag{9}$$

Note that in the RHS of the constraint (9) we use the term $\beta c(e)$ as the capacity of edge e. This is done to deal with edge capacity violations due to the channel assignment algorithm.

B. Flow Scaling

In this step, the algorithm computes the maximum value of interference for the K channels, namely it computes a scale L given by: $\zeta = \max\{1, \max_{1 \leq i \leq K} \mathrm{Int}(i)\}$. Next, the algorithm scales all flow values in the flow graph H by ζ: thus the new flow value for any edge e in the flow graph is set to: $f(e) = (f'(e))/(\zeta)$, where $f'(e)$ is the flow value on edge e of the flow graph after the post processing step of the algorithm. Note also that $\lambda^A = (\lambda^*)/(\zeta)$,, where λ^A is the new λ value corresponding to this scaled flow. Recall that λ^* is the optimal value for LP1 in Section III-A. Finally note that at the end of this step it is guaranteed that for each channel i the interference $\mathrm{Int}(i) \leq 1$. This also implies that for any edge $e \in E$ and any channel i the interference $\mathrm{Int}(i, e) \leq 1$.

VI. Link Flow Scheduling

In this section, we present necessary and sufficient conditions for interference-free link scheduling to achieve the link flows

for a given node radio channel assignment and a given traffic routing. In addition, we design an algorithm that outputs such an interference-free link scheduling whenever the sufficient condition is satisfied. Our results are obtained by extending those of [16] for the single-channel case and for the protocol model of interference [11].

Recall that we are assuming a periodic (with period T) time slotted schedule S, where the indicator variable $X_{e,i,\tau}, e \in E, i \in F(e), \tau \geq 1$ is 1 if and only if link e is active in slot τ on channel i and i is a channel in common among the set of channels assigned to the end-nodes of edge e.

A. Link Flow Scheduling: Necessary and Sufficient Conditions

Directly applying the result (Claim 2) in [16], it follows that a necessary condition for interference-free link scheduling is that for every $e \in E, i \in F(e), \tau \geq 1 : X_{e,i,\tau} + \sum_{e' \in I(e)} X_{e',i,\tau} \leq c(q)$ Here, $c(q)$ is a constant that only depends on the interference model. In our interference model this constant is a function of the fixed value q, the ratio of the interference range R_I to the transmission range R_T, and we derive it below for a particular value $q = 2$. Proofs for other values of q can be derived along similar lines.

Lemma 1: $c(q) = 8$ for $q = 2$.

Proof: Recall that an edge $e' \in I(e)$ if there exist two nodes $x, y \in V$ which are at most $2R_T$ apart and such that edge e is incident on node x and edge e' is incident on node y. Let $e = (u, v)$. Consider the region C formed by the union of two circles C_u and C_v of radius $2R_T$ each, centered at node u and node v respectively. Then, $e' = (u', v') \in I(e)$ if an only if at least one of the two nodes u', v' is in C; Denote such a node by $C(e')$.

Given two edges $e_1, e_2 \in I(e)$ that do not interfere with each other, we must have that the nodes $C(e_1)$ and $C(e_2)$ are at least $2R_T$ apart. Thus, an upper bound on how many edges in $I(e)$ do not pairwise interfere with each other can be obtained by computing how may nodes can be put in C that are pairwise at least $2R_T$ apart. For an even looser upper bound, we can extend C to a circle C_e of radius $3.5R_T$ which is centered in the middle of the line joining the endpoints of edge e and reformulate the above question as a circle packing problem: how many maximum circles of radius R_T can be packed (without overlap) in the circle C_e of radius $3.5R_T$? From [15], it follows that this number is 8. Thus, among the edges in $I(e)$ every "independent" set is of size at most 8. Thus, in schedule S in a given slot only one of the two possibilities exist: either edge e is scheduled or an "independent" set of edges in $I(e)$ of size at most 8 is scheduled implying the claimed bound. □

A Necessary Condition: (*Link Congestion Constraint*) Recall that $(1/T)\sum_{1 \leq \tau \leq T} X_{e,i,\tau} = (f(e(i))/c(e))$. Thus, it follows:

Lemma 2: Any valid "interference-free" edge flows must satisfy for every link e and every channel i the link congestion constraint

$$\frac{f(e(i))}{c(e)} + \sum_{e' \in I(e)} \frac{f(e'(i))}{c(e')} \leq c(q). \tag{10}$$

Next, we formulate a matching sufficient condition.

A Sufficient Condition: (*Link Congestion Constraint*).

Lemma 3: If the edge flows satisfy for every link e and every channel i the following *link schedulability constraint* than an interference-free edge communication schedule can be found

$$\frac{f(e(i))}{c(e)} + \sum_{e' \in I(e)} \frac{f(e'(i))}{c(e')} \leq 1. \tag{11}$$

The proof of this Lemma is established by demonstrating an algorithm which can find an interference-free edge communication schedule and is presented next.

B. Link Flow Scheduling: An Algorithm

In this section, we present a centralized version of the interference-free link scheduling algorithm following the work of [16]. It is also possible to design a *distributed* version of this algorithm along similar lines, as in [16].

The algorithm starts out by choosing a large number T such that all $N(e,k) = T(f(e(k)))/(c(e))$ are integral. Here, T is the period of the resulting schedule. This integrality condition can be further relaxed, however, we omit the details for lack of space. Next, the algorithm independently schedules the edges for each channel to obtain the schedule S_i for all $1 \leq i \leq K$. In this schedule, let $S_i(\tau)$ denote the set of edges scheduled at time slot τ by S_i. The algorithm then "merges" these K schedules to obtain an overall interference-free link schedule S, where the multiset of links (and channels) scheduled in slot τ is $S(\tau) = \cup_{1 \leq i \leq K} \cup_{e \in S_i(\tau)} (e, i)$. Thus, $(e, i) \in S(\tau)$ implies that link e is scheduled for communication using channel i in time slot τ in S. It is easily seen that if S_i for all $1 \leq i \leq K$ are feasible interference-free link schedules, then so is S. Thus, it is sufficient to design the algorithm for a particular channel i (to output schedule S_i).

Algorithm 4 presents the pseudocode for the scheduling algorithm for a given channel i. We assume edges in E are ordered as $e_1, e_2 \ldots e_m$. We denote by $S(e_i)$ the set of slots in which edge e_i is scheduled by the algorithm. The algorithm initializes all these sets to empty sets. The algorithm considers the edges in E in order and when considering an edge e_i, schedules it in the first set of $N(e_i, k)$ slots, where edge e_i can be scheduled without causing interference to any of the edges already scheduled in those slots by the algorithm. Since S is periodic, the algorithm only outputs schedule for first T slots.

Algorithm 4: Link Scheduling—For a Single-Channel k

Set Available slots to $1, 2 \ldots T$.

Initialize $S(e_i) = \emptyset, \forall i = 1, \ldots, m$

for $i = 1, \ldots, m$

Set $S(e_i)$ to first available $N(e_i, k)$ slots such that $S(e_i) \cap \cup_{e_j \in I(e_i)} S(e_j) = \emptyset$.

Note that by construction Algorithm 4 outputs an interference-free schedule. We omit the proof of the following Lemma which establishes that the Algorithm 4 is technically sound.

Lemma 4: If the edge flows satisfy the link schedulability constraint (11), then an interference-free schedule for the edges for any channel k can be found by the Algorithm 4.

The interference-free edge communication scheduling problem, given the set of edge flows is as hard as edge coloring even for very simple interference models and hence is NP-hard in general [17]. Thus, we cannot hope for an optimal algorithm. However, we can establish the following performance bound for our interference-free link scheduling algorithm (proof omitted).

1) Theorem 1: Given a set of "feasible" link flows, the algorithm presented in this section can be used to design a $c(q)$-approximation algorithm for finding interference-free edge communication schedule, where $c(q)$ is a constant defined in Lemma 1.

VII. Algorithm Analysis

We now show that the algorithm RCL outlined in the overview section finds a feasible solution to the joint channel assignment, routing, and interference-free edge communication scheduling problem, is computationally efficient and has a provable worst case performance bound (a constant that depends only on the total number of channels). Since it is clear that routing, scheduling, post processing, and scaling takes polynomial time, we only need to show that channel assignment step takes polynomial time in order to show that RCL runs in polynomial time. Due to space limitations, the proofs of all lemmas are omitted. Interested readers can refer to [2].

Lemma 5: Algorithm 1 (Phase 1) runs in time polynomial in $|V|, |E|, K/I$ and ensures that the total aggregate fractional flow on all the edges introduced in E' for every edge $e \in E$ in Phase 1 equals the aggregate fractional flow on edge e in the original network G.

Lemma 6: Algorithm 2 (Phase 2) runs in time polynomial in $K, |V|, |E|$.

Lemma 7: If in the original flow graph H the flow satisfies the link congestion constraint (10) for every channel $i, 1 \leq i \leq K$, then in the Algorithm 2 (Phase 2) the flow $\mu(e)c(e)$ on every edge $e \in E$ gets assigned to the edges $e(k)$ for channels $1 \leq k \leq I$. In other words, on termination the following holds for all edges:

$$\mu(e)c(e) = \sum_{i=1}^{I} f(e(i)).$$

It is easy to see that Phase 3 (Algorithm 3) runs in polynomial time since in each iteration the number of groups are reduced by at least one. Hence, the total running time is bounded by the number of connected components in the K channels, and is therefore bounded by $|E|K$.

Lemma 8: After flow scaling, the resulting flow satisfies the link capacity constraints for each channel.

Lemma 9: At the end of Phase 3 (Algorithm 3), the resulting channel assignment is feasible.

Theorem 2: The RCL algorithm is a $(Kc(q))/(I)$ approximation algorithm for the joint routing and channel assignment with interference-free edge scheduling problem.

Proof: Referring to the pseudocode for Algorithm 2, it follows that on termination of Phase 2 the interference on all channels is bounded as $\mathrm{Int}(i) \leq (Kc(q))/(I)$. Phase 3 (Algorithm 3) redistributes the edge flows over the K channels without increasing the interference on any channel. This is because in Phase 3 the flows moved to a channel j all come from the edges $e(i)$ in a single-channel i. Thus, $\mathrm{Int}(j) \leq \mathrm{Int}(i) \leq (Kc(q))/(I)$. Hence, in Phase 3, we must have $\zeta \leq (Kc(q))/(I)$. Postprocessing only reduces the maximum interference. In the flow scaling step, the flow is scaled by ζ. Therefore, the scaled flow corresponds to a λ value which is at least $\lambda^A = (\lambda^*)/(c(q)K/I)$. Since the optimal λ value is at most λ^* and since the scaled flow satisfies the sufficient condition (link schedulability constraint (11) for all channels) for it to be scheduled by our interference-free link scheduling algorithm (Section VI). Thus, the approximation bound follows. □

VIII. Evaluation

In this section, our goals are twofold: to evaluate the performance of our algorithm in realistic settings and to use our algorithm to study the performance gain of using multiple radios and multiple channels for WMNs. For the first goal ideally we should compare the performance of our algorithm against the optimal solution. However, the joint channel assignment and routing problem for WMNs quickly becomes intractable and any meaningful scenarios cannot be optimally solved in any practical setting. The other option therefore is to compare the average case performance of our algorithm versus the worst case bound we established earlier in this paper. This is what we evaluate here. For the second goal our evaluation is based on two sensitivity analysis studies that evaluate the improvement in the network throughput as the number of channels are increased and as the number of radios are increased.

We solve the three LPs in our RCL algorithm using CPLEX [1]. Our channel assignment algorithm uses the solution of LP2 as input. The channel assignment together with the total edge flow and λ^* from LP2 are the inputs to LP3. After postprocessing by LP3, we obtain ζ to get the feasible per-node throughput λ^*/ζ. We performed our evaluation on many realistic topologies and our simulation setup is as follows. The WMNs in our setting use 802.11a radios. We assume a simple wireless channel model in which link rates depend only on the distance between the links two end mesh network nodes. Adopting the values commonly advertised by 802.11a vendors, we assume that the link rate when the two end mesh nodes are within 30 m is 54 Mb/s, 48 Mb/s when within 32 m, 36 Mb/s when within 37 m, 24 Mb/s when within 45 m, 18 Mb/s when within 60 m, 12 Mb/s when within 69 m, 9 Mb/s when within 77 m, and 6 Mb/s when within 90 m. We assume the maximum transmission range R_T of 90 m and the maximum interference range of 180 m. We assume there are 12 channels available according to 802.11a specification. For simplicity, we assume that the gateway nodes have sufficient wired backhaul capacity for them not to be a bottleneck.

We generate grid and random topologies. We run our simulation with different parameter settings. We report results with the following parameters. We have a total of 60 nodes. For the grid topology, the grid size is 8×8, the distance between two adjacent grid points is $0.65R_T$, and nodes are placed in grid points

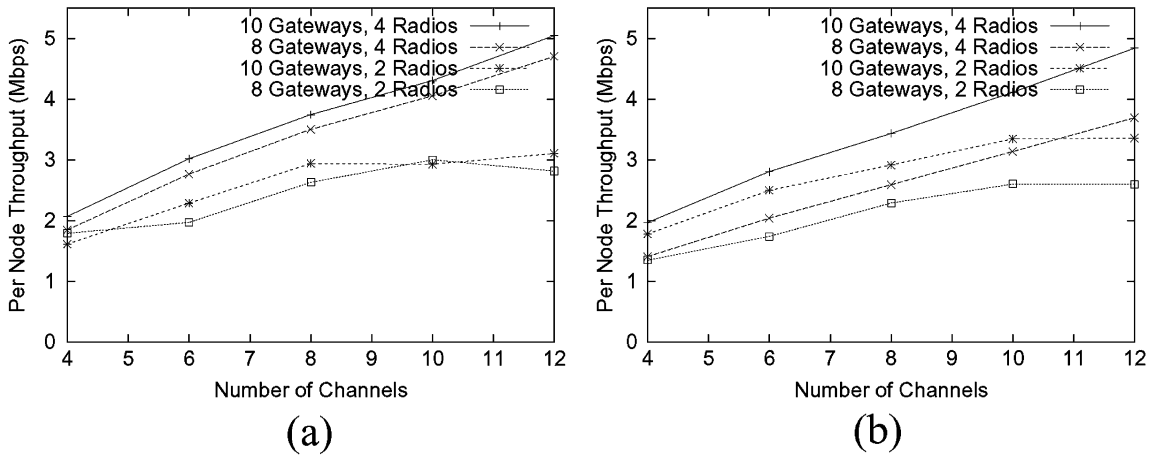

Fig. 2. Throughput improvement with increasing number of channels. (a) Grid topology. (b) Random topologies.

randomly. We generate nine random connected topologies by placing nodes randomly in a 500×500 square meter area. We choose a random sample of 20 nodes to have a traffic demand of 20 Mb/s each. We vary the number of gateway nodes from 2 to 12, the number of radios from 1 to 4, and the number of available channels from 1 to 12. We assume a uniform number of radios at all nodes.

A. The Performance Impact of Multichannel

In this evaluation, we varied the number of channels in the 60-node grid topology and random topologies to study its impact on the network throughput. The results are shown in Fig. 2 for four settings with varying number of gateways and radios. Each data point for random topologies is averaged over the nine topologies. As expected, we observe the trend that, as the number of channels increase, the per-node throughput generally increases. However, we remark that, the per-node throughput our algorithm computes may not always increase when the number of channels increase. This is because the channel assignment algorithm is not necessarily optimal and its performance depends on the network flow output by the routing step. In practice, one can use the solution with the highest throughput output by the algorithm. From Fig. 2, we see that our algorithm, in general, can effectively exploit the increasing number of channels available. For example, with ten gateways and four radios, as the number of channels goes from 4 to 12, the per-node throughput goes from 2.1 to 5.0 Mb/s for the grid topology case; it goes from 2.0 to 4.8 Mb/s for the random topologies case.

B. The Performance Impact of Multiradio

In this evaluation, we varied the number of radios and gateway nodes in the 60-node grid topology and random topologies to study their impacts on the network throughput. We fix the number of channels to be 12. Each data point for random topologies is averaged over the nine topologies. As can be seen from Fig. 3, our algorithm is able to exploit the increase in the number of radios and gateways to obtain a solution with improved per-node throughput. We see that the per-node throughput increases significantly from the one radio case to two radio case, much more than the percentage increase from 2 to 3 and from 3 to 4 radio case. For example, when there are 12 gateways, for the grid topology, the throughput corresponding to 1,2,3,4 radio case is 0.53, 3.8, 5.5, and 5.9 Mb/s, respectively; for random topologies, it is 1.0, 3.8, 5.0, and 5.4 Mb/s. With one more radio, we see a 620% and 280% increase in per-node throughput for the grid topology and random topologies, respectively. This result justifies the use of a small number of radios.

C. Performance Comparison With Upper Bound and Worst Case Bound

Even to compute the worst case bound, we need the optimal value for λ, the computation of which remains intractable in our setting. Thus, we used a upper bound λ^* on this value provided by the linear program LP1 (1). Therefore, an estimate on the worst case throughput of the algorithm is $(\lambda^*)/(W)l(v)$, where $W = c(q)(K)/(I)$ (for any $v \in V$). We compare this value and λ^* with the actual throughput that our algorithm is able to achieve. The results are shown in Fig. 4. In this evaluation, we used nine different grid and random topologies, each with 60 nodes. 20 nodes have traffic demands and there are 8 gateway nodes. Nodes have three radios each. We fix the number of channels to be 12. We can see that the algorithm's average case performance is around 5.3 to 7.9 and 8.3 to 28.7 times better than the worst case estimated performance for grid and random topologies, respectively. Our algorithm is at most 4.0 and 2.4 times worse than the upper bound for grid and random topologies, respectively. Note that, the upper bound is also very loose since the integrality gap may be large.

IX. Related Work

The work that is most closely related to this paper is that of [14] and [18]–[20]. Like ours, the work in [19] and [20] assumes that there is no system or hardware support to allow a radio interface to switch channels on a per-packet basis. Raniwala *et al.* propose a centralized joint channel assignment and multipath routing algorithm. The channel assignment algorithm considers high load edges first. The routing algorithm uses both shortest path routing and randomized multipath routing (a set of paths is used

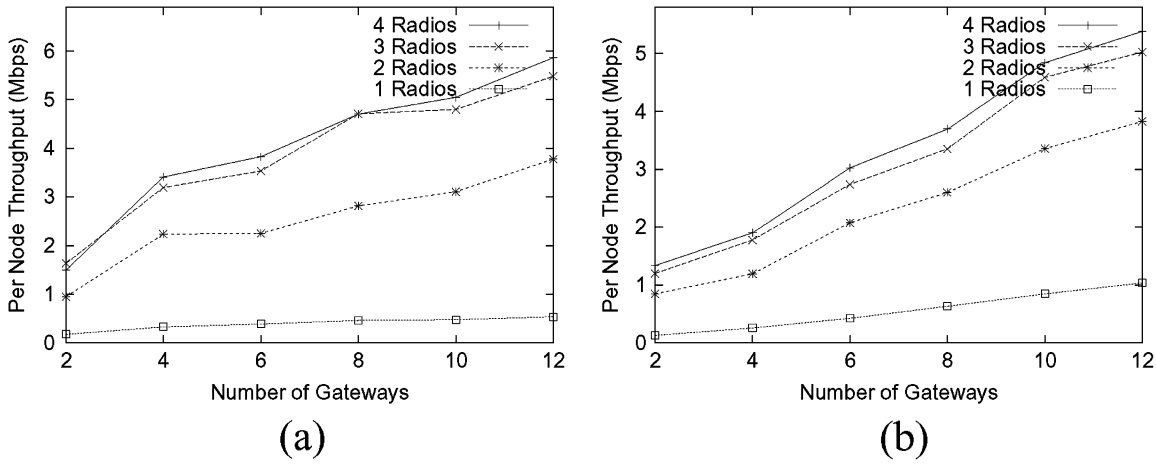

Fig. 3. Throughput improvement with increasing number of radios and gateways. (a) Grid topology. (b) Random topologies.

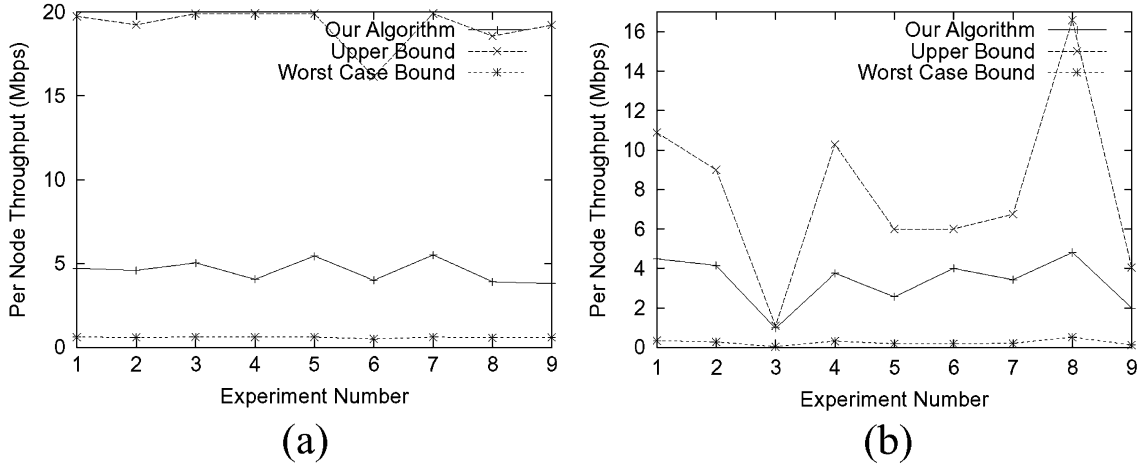

Fig. 4. Comparison with upper bound and worst case bound. (a) Grid topology. (b) Random topologies.

between any pair of communicating node pair). The joint channel assignment and multipath routing algorithm proceeds in an iterative fashion. However, their algorithm is based on heuristics and a worst performance bound on its performance is not known. In addition, in their scheme no guarantees on fair allocation of bandwidth is provided. In [19], Raniwala and Chiueh propose a distributed heuristic algorithm. The algorithm also is not known to have any worst case performance bound. Unlike ours, the work in [14] and [18] assume a radio interface is capable of switching channels rapidly and is supported by system software. In [14], Kodialam and Nandagopal presents channel assignment and routing algorithms to characterize the capacity regions between a given set of source and destination pairs. In [18], Kyasanur and Vaidya study how the capacity of multichannel wireless networks scale with respect to the number of radio interfaces and the number of channels as the number of nodes grow.

Algorithms aspects of wireless networks has been an active area of research. Jain *et al.* [12] consider throughput optimization using a general interference model. Their algorithm can be computationally intensive to achieve close to optimal performance. In addition, their algorithm does not exploit the properties of interference using 802.11 MAC for better performance. Kumar *et al.* [16] consider the throughput capacity of wireless networks between given source destination pairs for various interference models. However, they do not take channel allocation into account as they consider a single-channel network. Kodialam and Nandagopal [13] investigate the same problem using a simple interference model, where a node cannot send and receive at the same time. Objectives other than throughput have also been considered, e.g., power optimization [7].

There have also been approaches that consider routing and channel assignment separately. In [10], Draves *et al.* propose a routing metric that exploits multichannel diversity. In particular, paths with more channel diversity and fewer hops are preferred. In [4], Bahl *et al.* present a MAC protocol that exploits the availability of multiple channels. However, they change channel assignment in a fast time scale on a per-packet basis which may not work with existing commodity hardware.

In this paper, we assume the network is given. The problem of how to design a multihop mesh network has been studied in [5] and [9]. Their goal is to place a minimal set of gateways to meet certain performance requirements. They do not consider multiradio and multichannel mesh networks and their algorithms do not apply in our setting.

Finally, fair bandwidth allocation and load balancing has been considered in single-radio wireless LAN context [6]. Channel assignment has been extensively studied in cellular networks. However, there is no multihop routing in that context.

X. Conclusion and Future Work

IWMNs are increasingly deployed for commercial use and law enforcement. These deployment settings place stringent requirements on the performance of the underlying IWMNs. Bandwidth guarantee is one of the most important requirements of applications in these settings. For these IWMNs, topology change is infrequent and the variability of aggregate traffic demand from each mesh router (client traffic aggregation point) is small. These characteristics admit periodic optimization of the network which may be done by a system management software based on traffic demand estimation. In this paper, we rigorously formulate the joint channel assignment and routing problem in IWMNs. Our goal is to maximize the bandwidth allocated to each traffic aggregation point subject to fairness constraint. We propose a constant approximation algorithm for this NP-hard problem. Our algorithm takes interference constraint into account and is based on flow transformation. Our evaluation shows that the algorithm performs much better that the worst case bounds.

For future work, we would like to investigate the problem when routing solutions can be enforced by changing link weights of a distributed routing protocol such as OSPF. We would also like to improve the worst case bounds of our algorithms.

References

[1] *ILOG CPLEX Mathematical Programming Optimizers*, [Online]. Available: http://www.ilog.com/products/cplex/

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

[3] I. F. Akyildiz, X. Wang, and W. Wang, "Computer networks journal," in *Wireless Mesh Networks: A Survey*. New York: Elsevier, 2005.

[4] 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*, 2004, pp. 216–230.

[5] Y. Bejerano, "Efficient integration of multi-hop wireless and wired networks with QoS constraints," in *Proc. ACM MobiCom*, 2001, pp. 215–226.

[6] Y. Bejerano, S.-J. Han, and L. E. Li, "Fairness and load balancing in wireless LANs using association control," in *Proc. MobiCom*, 2004, pp. 315–329.

[7] R. Bhatia and M. Kodialam, "On power efficient communication over multi-hop wireless networks: Joint routing, scheduling and power control," in *Proc. IEEE INFOCOM*, 2004, pp. 1457–1466.

[8] T. X. Brown, H. N. Gabow, and Q. Zhang, "Maximum flow-life curve for a wireless ad hoc network," in *Proc. ACM MobiCom*, 2002, pp. 128–136.

[9] R. Chandra, L. Qiu, K. Jain, and M. Mahdian, "Optimizing the placement of internet TAPs in wireless neighborhood networks," in *Proc. IEEE ICNP*, 2004, pp. 271–282.

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

[11] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. IT-46, no. 2, pp. 388–404, Mar. 2000.

[12] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," in *Proc. ACM MobiCom*, 2003, pp. 66–80.

[13] M. Kodialam and T. Nandagopal, "Characterizing achievable rates in multi-hop wireless networks: The joint routing and scheduling problem," in *Proc. ACM MobiCom*, 2003, pp. 42–54.

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

[15] S. Kravitz, "Packing cylinders into cylindrical containers," in *Math. Mag.*, 1967, vol. 40, pp. 65–70.

[16] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, "Algorithmic aspects of capacity in wireless networks," in *Proc. ACM SIGMETRICS*, 2005, pp. 133–144.

[17] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, "End-to-end packet-scheduling in wireless ad-hoc networks," in *Proc. ACM-SIAM Symp. Discrete Algorithms*, 2004, pp. 1021–1030.

[18] P. Kyasanur and N. Vaidya, "Capacity of multi-channel wireless networks: Impact of number of channels and interfaces," in *Proc. ACM MobiCom*, 2005, pp. 43–57.

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

[20] A. Raniwala, K. Gopalan, and T.-C. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *ACM Mobile Comput. Commun. Rev. (MC2R)*, vol. 8, no. 2, pp. 50–65, 2004.

Mansoor Alicherry received the B.Tech. degree from Regional Engineering College, Calicut, India, in 1997 and the M.E. degree from the Indian Institute of Science, Bangalore, in 2000, both in computer science.

He is currently a Member of Technical Staff at the Network Software Research Department, Lucent Technologies, Bell Laboratories, Murray Hill, NJ. His interests are primarily in routing and design algorithms for optical networks.

Randeep Bhatia received the B.Tech. degree in computer science and engineering from the Indian Institute of Technology, Delhi, the M.S. degree in mathematics and computer science from the University of Illinois, Chicago, and the Ph.D. degree in computer science from University of Maryland, College Park.

He is currently with the High-Speed Networks Research Department, Bell Laboratories, Lucent Technologies, Murray Hill, NJ, working on network design, traffic engineering, and scheduling algorithms. His current research interests are in the area of QoS for emerging multimedia services in next-generation networks.

Li Erran Li received the B.E. degree in automatic control from Beijing Polytechnic University, Beijing, China, in 1993, the M.E. degree in pattern recognition from the Institute of Automation, Chinese Academy of Sciences, Beijing, in 1996, and the Ph.D. degree in computer science from Cornell University, Ithaca, NY, in 2001, where J. Y. Halpern was his Advisor.

During his graduate study at Cornell University, he worked at Microsoft Research and Bell Laboratories, Lucent Technologies, as an intern, and at AT&T Research Center at ICSI Berkeley as a visiting student. He is presently a member of the Networking Research Center, Bell Laboratories, Lucent Technologies, Murray Hill, NJ. He has published over 30 papers. His research interests are in networking with a focus on wireless networking and mobile computing.

Dr. Li has served as a program committee member for several conferences including ACM MobiCom, ACM MobiHoc, IEEE INFOCOM, and IEEE ICNP. He is a Guest Editor for the JOURNAL ON SELECTED AREAS IN COMMUNICATION (Special Issue on Noncooperative Behavior in Networking).