Minimum Interference Channel Assignment in Multi-Radio Wireless Mesh Networks

Anand Prabhu Subramanian, Himanshu Gupta, and Samir R. Das
{anandps, hgupta, samir}@cs.sunysb.edu
Stony Brook University, NY, USA

***Abstract*—In this paper, we consider multi-hop wireless mesh networks, where each router node is equipped with multiple radio interfaces and multiple channels are available for communication. We address the problem of assigning channels to communication links in the network with the objective of minimizing overall network interference. Since the number of radios on any node can be less than the number of available channels, the channel assignment must obey the constraint that the number of different channels assigned to the links incident on any node is atmost the number of radio interfaces on that node. The above optimization problem is known to be NP-hard.**

We design centralized and distributed algorithms for the above channel assignment problem. To evaluate the quality of the solutions obtained by our algorithms, we develop a semidefinite program formulation of our optimization problem to obtain a lower bound on overall network interference. Empirical evaluations on randomly generated network graphs show that our algorithms perform close to the above established lower bound, with the difference diminishing rapidly with increase in number of radios. Also, detailed *ns-2* simulation studies demonstrate the performance potential of our channel assignment algorithms in 802.11-based multi-radio mesh networks.

I. Introduction

Recently, there is an increasing interest in using wireless mesh networks [2] as broadband backbone networks to provide ubiquitous network connectivity in enterprises, campuses, and in metropolitan areas. An important design goal for wireless mesh networks is *capacity*. It is well-known that wireless interference severely limits network capacity in multi-hop settings [11]. One common technique used to improve overall network capacity is use of multiple channels [15]. Essentially, wireless interference can be reduced by using orthogonal (non-interfering) channels for neighboring wireless transmissions. The current IEEE 802.11 standard for WLANs (also used for mesh networks) indeed provides several orthogonal channels to facilitate the above. Presence of multiple channels requires us to address the problem of which channel to use for a particular transmission.

This work is supported in part by NSF grants CNS-0519734, OISE-0423460, CNS-0308631 and a grant from the SensorCAT center.

One of the channel assignment approaches is to frequently change the channel on the interface; for instance, for each packet transmission based on the current state of the medium. Such *dynamic channel assignment* approaches [4, 30, 31, 34] require channel switching at a very fast time scale (per packet or a handful of packets). The fast-channel switching requirement makes these approaches unsuitable for use with commodity hardware, where channel switching delays itself can be in the order of milliseconds [6] which is an order of magnitude higher than typical packet transmission times (in microseconds). Some of the dynamic channel assignment approaches also require specialized MAC protocols or extensions of 802.11 MAC layer, making them further unsuitable for use with commodity 802.11 hardware.

In order to use multiple channels with commodity hardware effectively, there is need to develop techniques that assign channels statically [1, 21, 26, 27, 32]. Such static assignments can be changed whenever there are significant changes to traffic load or network topology; however, such changes are infrequent enough that the channel-switching delay and traffic measurement (see Section II) overheads are inconsequential. We refer to the above as *quasi-static channel assignments*.

If there is only one radio per node, then the above channel assignment schemes will have to assign the *same* channel to all radios/links in the network to preserve network connectivity. Thus, such assignment schemes require use of multiple radio interfaces at each node. Due to board crosstalk or radio leakage [1, 29], commodity radios on a node may actually interfere even if they are tuned to different channels. However, this phenomena can be addressed by providing some amount of shielding or antenna separation [17, 29].

In this work, we address the problem of quasi-static assignment of channels to links in the context of networks with multi-radio nodes. The objective of the channel assignment is to minimize the *overall network interference*. Channel assignment is done as some variation of a graph coloring problem; but it has an interesting twist in the context of multi-radio mesh networks. The assignment of channels to links must obey the *interface constraint* that the number of different channels assigned to the links incident on a node is at most the number of interfaces on that node. Different variations of this problem have been shown to be NP-hard [21, 26] before. Thus, efficient algorithms that run reasonably fast and provide good quality solutions are of interest. Since computing the optimal is intractable and approximation algorithms are still an open question, we take the approach of computing a *bound on the optimal* using a mathematical programming approach and develop simple heuristics that perform very close to the obtained bound on the optimal.

Contributions: For the channel assignment problem described above, we develop simple centralized and distributed algorithms. To evaluate their performances, we develop a mathematical programming formulation using semidefinite programming (SDP) and obtain a *lower bound* on the optimal network interference by relaxing the SDP formulation to run in polynomial time. Finally, detailed ns-2 simulation studies demonstrate the full potential of our channel assignment algorithms in 802.11 based multi-radio mesh networks.

The *salient features of our work* that set us apart from the existing channel assignment approaches on multi-radio platforms are as follows.

- Our approach is "topology preserving," i.e., all links that can exist in a single channel network also exist in the multichannel network after channel assignment. Thus, our channel assignment does not have any impact on routing.
- Our approach is suitable for use with commodity 802.11-based networks without any specific systems support. We do not require fast channel switching or any form of MAC layer or scheduling support. While our algorithms indeed use interference and traffic models as input, such models can be gathered using experimental methods.
- Our work generalizes to non-orthogonal channels [20], including channels that are supposedly orthogonal but interfere because of crosstalk or leakage [29].
- Ours is the first work that establishes a good lower bound on the optimal network interference, and demonstrates good performance of the developed heuristics by comparing them with the lower bound.

The rest of the paper is organized as follows. We start with describing the network model and the formulation of our problem in Section II, and discuss related work in Section III. We present our algorithms in Section IV and Section V respectively. In Section VI, we obtain a lower bound on the optimal network interference using semidefinite programming. Section VII presents generalizations of our techniques. We present our simulation results in Section VIII and conclude our paper in section IX.

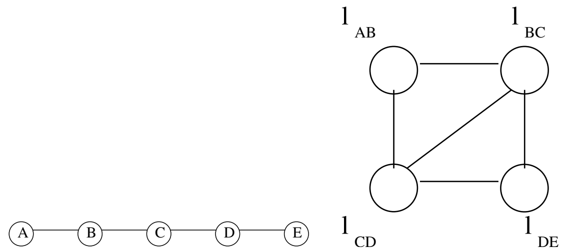
II. Problem Formulation

In this section, we first present our network model and formulate our channel assignment problem.

Network Model: We consider a wireless mesh network with stationary wireless routers. We model the *communication graph* of the network as a general undirected graph over the set of network nodes N. Each node $i \in N$ has R_i radio interfaces. An edge (i, j) in the communication graph is referred to as a *communication link* or *link*, and signifies that the nodes i and j can communicate with each other as long as both the nodes have a radio interface each with a common channel. There are K channels available in the network and let $\mathcal{K} = \{1, 2, \ldots, K\}$ denote the set of K available channels. For clarity of presentation, we assume for now that the channels are orthogonal (non-interfering), and extend our techniques for non-orthogonal channels in Section VII.

Interference Model: The *interference model* defines the set of links that can interfere with any given link in the network. There have been various interference models proposed in the literature, for example, the physical and protocol interference models [11, 14, 22]. The discussion in this paper is independent of the specific interference model used as long as the interference model is defined on pairs of communication links. For clarity of presentation, we assume a *binary interference model* for now (i.e., two links either interfere or do not interfere), and generalize our techniques to fractional interference in Section VII. The level of interference between two links actually depends on the traffic on the links. However, for clarity of presentation, we assume uniform traffic on all links for now, and generalize our techniques to non-uniform traffic in Section VII.

Given an interference model, the set of pairs of communication links that interfere with each other can



(a) Communication graph (b) Conflict graph

Fig. 1. Communication graph and corresponding conflict graph.

be represented using a *conflict graph* [14]. To define a conflict graph, we first create a set of vertices V_c corresponding to the communication links in the network. In particular,

$$V_c = \{l_{ij} \mid (i,j) \text{ is a communication link}\}.$$

Now, the conflict graph $G_c(V_c, E_c)$ is defined over the set V_c as vertices, and a *conflict edge* (l_{ij}, l_{ab}) is used to signify that the communication links (i,j) and (a,b) interfere with each other if they are on the same channel. The above concept of a conflict graph can be used to represent any interference model. We illustrate the concept of conflict graph in Figure 1. In this figure, we assume an 802.11 like interference model where the transmission range and interference range are equal. When RTS/CTS control messages are used, links within two hops interfere.

Channel Assignment Problem: Given a mesh network of router nodes with multiple radio interfaces, we wish to assign a unique channel to each communication link in the network such that the number of different channels assigned to the links incident on any node is atmost the number of radios on that node. Since we assume uniform traffic on all links for now, we define the *total network interference* as the number of pairs of communication links that are interfering (i.e., are assigned the same channel and are connected by an edge in the conflict graph). The objective of our problem is to minimize the above defined total network interference, as it results in improving overall network capacity [11].

More formally, consider a wireless mesh network over a set N of network nodes. The *channel assignment problem* is to compute a function $f : V_c \rightarrow \mathcal{K}$ to minimize the *overall network interference* $I(f)$ defined below while satisfying the below *interface constraint*.

$$\forall i \in N, \quad |\{k \mid f(e) = k \text{ for some } e \in E(i)\}| \leq R_i$$

where, $E(i) = \{l_{ij} \in V_c\}$, i.e., $E(i)$ is set of vertices in V_c that represent the communication links incident on node i and the *network interference* $I(f)$ is

$$I(f) = |\{(u,v) \in E_c \mid f(u) = f(v)\}| \qquad (1)$$

If we look at assignment of channels to vertices as coloring of vertices, then the network interference is just the number of monochromatic edges in the vertex-colored conflict graph. We use the term colors and channels interchangeably in the rest of the paper.

Measuring Interference and Traffic: Note that, under the simplying assumption of uniform traffic, the only input to our channel assignment problem is the network conflict graph. The conflict graph (along with the edge weights for fractional interference; see Section VII) can be computed using methods similar to recently reported measurement-based techniques in [24, 28]. These techniques are localized, due to the localized nature of interference, and hence, can be easily run in a distributed manner. Also, in most cases (for static network topologies), the above measurements need to be done only one-time. For the case of non-uniform traffic, we need to measure average (over the time scale of channel assignment) traffic (i.e., the function $t(.)$ of Section VII) on each link. Such traffic measurements can be easily done using existing software tools (e.g., COMO [33]).

Relationship with Max K-cut: Given a graph G, the Max K-cut problem [9] is to partition the vertices of G into K partitions in order to maximize the number of edges whose endpoints lie in *different* partitions. In our channel assignment problem, if we view vertices of the conflict graph assigned to a particular channel as belonging to one partition, then the network interference is actually the number of edges in the conflict graph that have endpoints in *same* partition. Thus, our channel assignment problem is basically the Max K-cut problem with the added interface constraint. Since Max K-cut is known to be NP-hard, our channel assignment problem is also NP-hard.

III. Related Work

Following our discussion in Section I, we classify the related work as follows.

Fast Switching of Channels: Authors in [4, 30, 31, 34] propose new MAC protcols or modifications to the 802.11 MAC protocols to support use of multiple channels on a per packet or a handful of packet basis.

All the above protocols require very small channel switching delay (of the order of hundred microseconds or less), since channels are switched at a fast time scale. But, the commodity 802.11 wireless cards incur a channel switching delay of the order of milliseconds [6]. In addition, the above approaches require changes to the

MAC layer. Thus, they are not suitable with currently available commodity hardware.

Static/Quasi-Static Channel Assignment in Multiradio Networks: In [27], the authors assume a tree-based communication pattern to ease coordination for optimizing channel assignment. [32] considers minimum-interference channel assignments that preserve k-connectivity. None of the above schemes preserve the original network topology, and hence, may lead to inefficient assignments and routing in a more general peer-to-peer communication.

Prior works on topology preserving channel assignment strategies are as follows. Adya et al. [1] propose a strategy wherein they assume a hard-coded assignment of channels to interfaces, and then determine which channel/interface to use for communication via a measurement-based approach. They only use as many channels as the number of interfaces. In [26], Raniwala et al. propose a centralized load-aware channel assignment algorithm; however, they require that source-destination pairs with associated traffic demands and routing paths to be known a priori. In [8], Das et al. present a couple of optimization models for the static channel assignment problem in a multi-radio mesh network. However, they do not present any practical (polynomial time) algorithm. In [25], a purely measurement-based approach is taken for channel assignment to radios (instead of links). Here, one radio at each node is tuned to a common channel to preserve the original topology; however, this can be wasteful when only a few interfaces are available. Moreover, assignment of channels to radios still leaves the problem of which channel to use for a transmission/link. In the most closely related work to ours, Marina and Das in [21] address the channel assignment to communication links in a network with multiple radios per node. They propose a centralized heuristic for minimizing the network interference. We compare the performance of our proposed algorithm with this heuristic, and show a significant improvement.

In other related works, [16] proposes a hybrid channel assignment strategy: some interfaces on a node have a fixed assignment, and the rest can switch channels as needed. To put things in perspective, our work presents algorithms for making these fixed assignments. Authors in [18, 22] address joint channel assignment, routing and scheduling problems. Both these papers makes an assumption of synchronized time-slotted channel model as scheduling is integrated in their methods. This makes these approaches somewhat impractical with commodity radios. Finally, [15] derives upper bounds on capacity of wireless multihop networks with multiple channels.

IV. Centralized Tabu-based Algorithm

In this section, we describe our centralized algorithm for the channel assignment problem, based on Tabu search [12] technique used for graph coloring. Centralized algorithms are quite practical in "managed" mesh networks where there is already a central entity. Moreover, they are amenable to a higher degree of optimization, easier to upgrade, and use of "thin" clients. Centralized approaches have indeed been proposed in various recent works [21, 26, 32], and have also become prevalent in the industry (e.g., WLAN and mesh products from Meru Networks [19])

Recall that our channel assignment problem is to color the vertices V_c of the conflict graph G_c using K colors while maintaining the interface constraint and minimizing the number of monochromatic edges in the conflict graph. In other words, the channel assignment problem is to find a solution $f : V_c \rightarrow \mathcal{K}$ with minimum network interference $I(f)$ such that f satisfies the interference constraint. Our Tabu-based algorithm consists of two phases. In the first phase, we use a Tabu search based technique [12] to find a good solution f without worrying about the interface constraint. In the second phase, we remove interface constraint violations to get a feasible channel assignment function f.

First Phase: We start with a random initial solution f_0 wherein each vertex in V_c is assigned to a random color in $\mathcal{K}$. Starting from such a random solution f_0, we create a sequence of solutions $f_0, f_1, f_2, \ldots, f_j, \ldots,$ in an attempt to reach a solution with minimum network interference. In the j^{th} iteration ($j \geq 0$) of this phase, we create the next solution f_{j+1} in the sequence (from f_j) as follows.

The j^{th} Iteration: First, we generate a certain number (say, r) of random neighboring solutions of f_j. A random neighboring solution of f_j is generated by picking a random vertex u and reassigning it to a random color in $(\mathcal{K} - \{f_j(u)\})$. Thus, a neighboring solution of f_j differs from f_j in the color assignment of only one vertex. Among the set of such randomly generated neighboring solutions of f_j, we pick the neighboring solution with the lowest network interference as the next solution f_{j+1}. Note that we do not require $I(f_{j+1})$ to be less than $I(f_j)$, so as to allow escaping from local minima.

Tabu List: To achieve fast convergence, we avoid reassigning the same color to a vertex more than once by

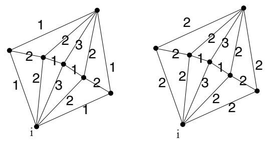


Fig. 2. Merge operation of second phase. The two figures are the communication graphs of the network before and after the merge operation. Labels on the links denote the color/channel. Here, the merge operation is started at node i by changing all its 1-colored links to color 2.

maintaining a *tabu list* τ of limited size. In particular, if f_{j+1} was created from f_j by assigning a new color to a vertex u, then we add $(u, f_j(u))$ to the tabu list τ. Now, when generating random neighboring solutions, we ignore neighboring solutions that assign the color k to u if (u, k) is in τ.

Termination: We keep track of the best (i.e., with lowest interference) solution f_{best} seen so far by the algorithm. The first phase terminates when maximum number (say, i_{max}) of allowed iterations have passed without any improvement in $I(f_{best})$. In our simulations, we set i_{max} to $|V_c|$. Since network interference $I(f)$ takes integral values and is at most $(|V_c|)^2$, the value $I(f_{best})$ is guaranteed to decrease by at least 1 in $i_{max} = |V_c|$ iterations (or else, the first phase terminates). Thus, the time complexity of the first phase is bounded by $O(rd|V_c|^3)$, since each iteration can be completed in $O(rd)$ time where r is the number of random neighboring functions generated and d is the maximum degree of a vertex in the conflict graph. Note that network interference of a neighboring solution can be computed in $O(d)$ time.

Second Phase: The solution f returned by the first phase may violate interface constraints. In the second phase, we eliminate the violations by repeated application of the following "merge" procedure. Given a channel assignment solution f, we pick a network node for the merge operation as follows. Among all the network nodes wherein the interface constraint is violated, we pick the node wherein the violation is maximum. Let i be the node picked as above for the merge operation. We reduce the number of colors incident on i by picking (as described later) two colors k_1 and k_2 incident on i, and changing the color of all k_1-colored links to k_2. In order to ensure that such a change does not create interface constraint violations at other nodes, we *iteratively* "propagate" such a change to all $k_1 - colored$ links that are "connected" to the links whose color has been just changed from k_1 to k_2. Here, two links are said to be connected if they are incident on a common node. Essentially the above propagation of color-change ensures that for any node j, either *all or none* of the k_1-colored links incident on j are changed to color k_2. See Figure 2. Completion of the above described color-change propagation marks the completion of *one* merge procedure. The above described merge procedure reduce the number of distinct colors incident on i by one, and does not increase the number of distinct colors incident on any other node (due to the all or none property). Thus, repeated application of such a merge operation is guaranteed to resolve all interface constraints. Note that a merge operation probably will result in increase in network interference. Thus, for a given node i, we pick those two color k_1 and k_2 for the merge operation that cause the least increase in the network interference due to the complete merge operation.

V. Distributed Greedy Algorithm (DGA)

In this section, we describe our Distributed Greedy Algorithm (DGA) for the channel assignment problem. Our choice of greedy approach is motivated by the observation that simple greedy heuristics have given very good solutions to Max-K-cut problem in uniform random graphs $G_{n,p}$ [7] and we can show that the conflict graph corresponding to a random network is a $G_{n,p}$ graph, under the protocol interference model [11].

Centralized Version: We start with presenting a centralized version, which yields a natural distributed implementation. In the initialization phase of our greedy approach, each vertex of V_c is colored with the color 1. Then, in each iteration of the algorithm, we try to change the color of some vertex in a greedy manner without violating the interface constraint. This strategy is different from the Tabu-based algorithm, where we resolve interface constraint violations in the second phase while not worrying about introducing them in the first phase. In each iteration of the greedy approach, we try to change the color of some vertex $u \in V_c$ to a color k. We look at all possible pairs of u and k, considering only those that do not result in the violation of any interface constraint, and pick the pair (u, k) that results in the largest decrease in network interference. The algorithm iterates over the above process, until there is no pair of u and k that decreases the network interference any further. Note that a vertex in V_c may be picked multiple times in different iterations. However, we are guaranteed to terminate because each iteration monotonically decreases the network interference. In particular, as noted in previous section, since the network interference takes integral values and is at most $(|V_c|)^2$, the number of iterations of the greedy algorithm is bounded by $(|V_c|)^2$. Since each iteration can be completed in $O(dK|V_c|)$, where K is the total number of colors and d is the maximum degree of a vertex in the conflict graph, the time complexity of the greedy algorithm is $O(dK|V_c|^3)$.

Distributed Greedy Algorithm (DGA) : The greedy approach described above can also be easily distributed by using a localized greedy strategy. The distributed implementation differs from the centralized implementation in the following aspects. Firstly, in the distributed setting, multiple link-color pairs may be picked simultaneously across the network by different nodes. Secondly, the decision of which pair is picked is based on the local information. Lastly, to guarantee termination in a distributed setting, we impose additional restriction that each pair (u, k) is picked at most once (i.e., each vertex $u \in V_c$ is assigned a particular color k at most once) in the entire duration of the algorithm.

In the distributed implementation, each vertex $u = l_{ij} \in V_c$ corresponding to the link (i, j) is *owned* by i or j, whichever has the higher node ID. This is done to ensure consistency of color information across the network. Initially, each vertex in V_c is assumed to colored 1. Let $m \geq 1$ be the parameter defining the local neighborhood of a node. Based on the information available about the colors of links in the m-hop neighborhood of i, each network node i selects (after waiting for a certain random delay) a (u, k) combination such that (i) $u = l_{ij}$ is owned by i, (ii) changing the color of u to k does not violate the interface constraint at node i or j, (iii) the pair (u, k) has not been selected before by i, and (iv) the pair (u, k) results in the largest decrease in the "local" network interference. Then, the node i sends a `ColorRequest` message to node j. The node j responds with the `ColorReply` message, if and only if changing the color of u to k still does not violate the interface constraint at node j. On responding with the `ColorReply` message, the node j *assumes*[1] that the color of u has been changed to k. On receiving the `ColorReply` message for j, the node i sends a `ColorUpdate`(u, k) message to all its m-hop neighbors. If a `ColorReply` message is not received within a certain time period, the node i abandons the choice of (u, k) for now, and starts a fresh iteration. Since each pair (u, k) is picked at most once, then the total number of iterations (over all nodes) in the above algorithm is at most $O(|V_c|K)$.

[1]Such an assumption may need to be later corrected through communication with i if the `ColorUpdate(u,k)` message is not received from i within a certain amount of time.

VI. Bound on Optimal Network Interference

In this section, we derive a lower bound on optimal network interference using semidefinite programming approaches. The lower bound will aid in understanding the quality of the solutions obtained from the heuristics presented in previous two sections.

A. Semidefinite Programming Formulation

In this section, we present a semi-definite program [10] formulation of our channel assignment problem. A *semidefinite program* is a technique to optimize a linear function of a symmetric positive-semidefinite matrix[2] subject to linear equality constraints. The reader is referred to [3, 10] for details on semidefinite programming and its application to combinatorial optimization.

As mentioned in Section II, our channel assignment problem is essentially the Max K-cut problem in the conflict graph with the additional interface constraint. Here, we start with presenting the SDP for the Max K-cut problem from [9]. We then extend it to our channel assignment problem by adding the interface constraint.

SDP for Max K-cut: Let y_u be a variable that represent the color of a vertex $u \in V_c$. Instead of allowing y_u to take 1 to K integer values, we define y_u to be a vector in $\{a_1, a_2, ..., a_K\}$, where the a_i vectors are defined as follows [9]. We take an equilateral simplex Σ_K in $\mathbf{R}^{K-1}$ with vertices $b_1, b_2, ..., b_K$. Let $c_K = \frac{(b_1+b_2+...+b_K)}{K}$ be the centroid of Σ_K, and let $a_i = b_i - c_K$ for $1 \leq i \leq K$. Also, assume $|a_i| = 1$ for $1 \leq i \leq K$. The integer quadratic program for the Max K-cut problem can now be represented as follows [9].

$\mathbf{IP_{Max-K}}$:

$$\text{Maximize} \qquad \frac{K-1}{K} \sum_{(u,v)\in E_c} (1 - y_u.y_v)$$

$$\text{such that} \qquad y_u \in \{a_1, a_2, ..., a_K\}$$

Note that since $a_i.a_j = \frac{-1}{K-1}$ for $i \neq j$, we have:

$$1 - y_u.y_v = \begin{cases} 0 & \text{if } y_u = y_v \\ \frac{K}{K-1} & \text{if } y_u \neq y_v. \end{cases}$$

Interface Constraint: We now add the interface constraint to the above SDP formulation for Max K-cut. For each $i \in N$, let

$$\Phi_i = \sigma(E(i), R_i) - (\binom{|E(i)|}{2} - \sigma(E(i), R_i))/(K-1),$$

[2]A matrix is said to be *positive semidefinite* if all its eigen values are nonnegative.

where $\sigma(E(i), R_i)$ is defined as follows:

$$\sigma(S_u, K) = \frac{\beta\alpha(\alpha+1) + (K-\beta)\alpha(\alpha-1)}{2}, \quad (2)$$

where $\alpha = \lfloor \frac{|S_u|}{K} \rfloor$ and $\beta = |S_u| \ mod \ K$. It can be shown [23] that the number of monochromatic edges in the clique of size $|S_u|$ when colored by K colors is at least $\sigma(S_u, K)$. Now, we add the following constraint to represent the interface constraint.

$$\sum_{u,v \in E(i)} y_u . y_v \geq \Phi_i \quad \forall i \in N \quad (3)$$

Recall that vertices in $E(i)$ form a clique in the conflict graph, and cannot be partitioned into more than R_i partitions to satisfy our interface constraint. Now, $\sigma(E(i), R_i))$ gives a lower bound on the number of monochromatic edges in this clique ($E(i)$) [23], and thus, $\binom{|E(i)|}{2} - \sigma(E(i), R_i))$ is an upper bound on the number of non-monochromatic edges. Since we know that $y_u . y_v = 1$ for any monochromatic edge (u, v) and $y_u . y_v = \frac{-1}{K-1}$ for any non-monochromatic edge, we have constraint in the above Equation 3.

Note that even though Equation 3 is a valid constraint, it does not necessarily restrict the number of colors assigned to vertices of $E(i)$ to R_i. Thus, the IP_{Max-K} augmented by the above Equation 3 only gives an upper bound on the number of non-monochromatic edges.

Relaxed SDP for Channel Assignment: Since we cannot solve the integer quadratic program IP_{Max-K} for problems of reasonable size, we relax it by allowing the variables y_u to take any unit vector in $R^{|V_c|}$. Since $y_u . y_v$ can now take any value between 1 and -1, we add an additional constraint to restrict $y_u . y_v$ to be greater than $\frac{-1}{K-1}$.

The solution to the relaxed SDP program gives an upper bound on the number of non-monochromatic edges, and the lower bound on the optimal network interference can be obtained by subtracting it from $|E_c|$. The above relaxed version can be easily converted into the standard SDP formulation for use by a standard SDP solver such as DSDP 5.0 [5]. We omit the details due to lack of space.

VII. Generalizations

In this section we provide some generalizations to our techniques which are very useful in practical deployments. For example, the links in the network communication graph may carry different amounts of traffic. Thus, the average interference must be weighted by traffic as interfering traffic is not the same for all interfering link pairs. Also, channels – even when they are orthogonal in theory – do interfere due to device imperfections (e.g., radio leakage, improper shielding, etc.) [29]. Thus, modeling of non-orthogonal (i.e., interfering) channels is a good idea. In addition, this also allows us to explicitly utilize non-orthogonal channels [20]. Finally, regardless of traffic and use of different channels, path loss effects can influence the degree of interference between two links – and thus, result in fractional interference between two links.

Non-uniform Traffic and Fractional Interference: Let u and v be two vertices in the conflict graph, $r(u, v)$ (a real number between 0 and 1) be the level of interference between two links corresponding to the vertices u and v, and $t(u)$ and $t(v)$ denote the normalized traffic on the links corresponding to the vertex u and v respectively. Now, the overall network interference for a given channel assignment function $f : V_c \to \mathcal{K}$ can be defined as follows. Let $M = \{(u, v)|u, v \in V_c \text{ and } f(u) = f(v)\}$. Then,

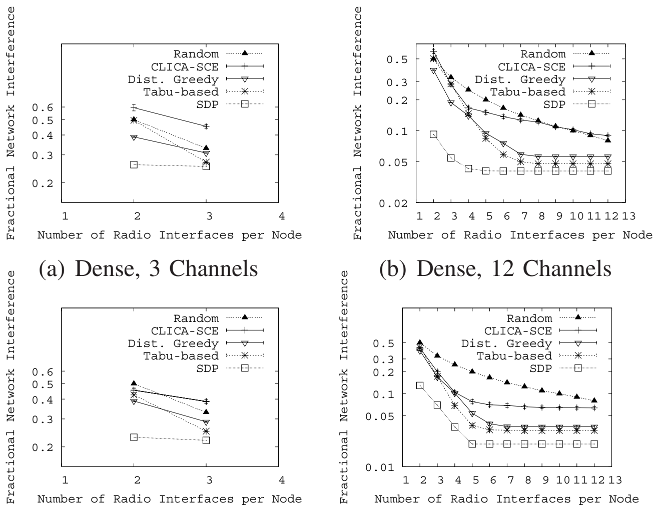
$$I(f) = \sum_{(u,v) \in M} t(u)t(v)r(u, v).$$

For the generalized interference and traffic model, the Tabu-based and Greedy algorithms use the above definition of network interference; no additional changes are required. Similarly, the SDP formulation can be generalized by appropriately extending the objective function.

Non-orthogonal Channels: Let $c(k_1, k_2)$, a value between 0 and 1, denote the level of interference between two channels k_1 and k_2. For non-orthogonal channels, the objective function can be further generalized as follows for a given channel assignment function $f : V_c \to \mathcal{K}$.

$$I(f) = \sum_{(u,v) \in M} t(u)t(v)r(u, v)c(f(u), f(v)).$$

As before, Tabu-based and Greedy algorithms can use the above definition of network interference without any additional changes. Unfortunately, the SDP formulation cannot be generalized easily for non-orthogonal channels. The problem arises from the difficulty in choosing appropriate vectors a_i such that $a_i.a_j$ is proportional to $c(i, j)$ for all channels $i, j \in \mathcal{K}$. The values $c(i, j)$ are characteristics of the channel spectrum, and can be measured independently [13].



(a) Dense, 3 Channels (b) Dense, 12 Channels

(c) Sparse, 3 Channels (d) Sparse, 12 Channels

Fig. 3. Fractional network interference of solutions delivered by various algorithms compared with the lower bound in dense or sparse networks for 3 or 12 channels.

VIII. Performance Evaluation

We present our performance results for two different settings. First, we evaluate a graph-theoretic performance metric, and then, evaluate throughput improvement using ns2 simulations.

In addition to our designed algorithms (Tabu-based and Distributed Greedy) and the lower bound obtained from the semidefinite programming technique, we also present results for two other algorithms for comparison. In particular, we simulate CLICA-SCE, a modified version of the centralized CLICA heuristic presented in [21] for a slightly different version of the channel assignment problem.[3] We also simulate a *random* algorithm which uses only a limited number of channels (equal to the number of radio interfaces), assigns a different channel to each radio interface, and then, selects a random interface (and hence, channel) for transmitting a packet. See Section III for a discussion on other related works.

Graph-Theoretic Performance: We consider two sets of random network, viz., dense and sparse networks, generated by randomly placing 50 nodes in 500×500 and 800×800 square meters of area respectively.[4] In dense networks, the average node degree is around 10, while in sparse networks the average node degree is around 5. Each node has the same number of radio interfaces, and has a uniform transmission and interference range of 150 meters. The protocol interference model [11] is used to compute the conflict graph. To solve semidefinite programs, we used DSDP 5.0 [5] which uses an efficient interior-point technique. We assume orthogonal channels and uniform traffic on all links.

We evaluate the performance of our algorithms in random networks using the metric "fractional network interference". Given a channel assignment function f computed by an algorithm, the *fractional network interference* is defined as the ratio of network interference ($I(f)$) and the total number of edges in the conflict graph. This represents the number of conflicts that remain even after channel assignment relative to the number of conflicts in the single-channel network.

In Figure 3, we plot the fractional network interference for varying number of radio interfaces/node, in dense and sparse networks using 3 and 12 channels. In general, both our algorithms perform extremely well compared to the CLICA-SCE and random algorithms. The Tabu-based algorithm almost always performs better than the Distributed Greedy algorithm, except when the number of radios is very small.

The performance of our algorithms compared to the lower bound obtained from the SDP formulation shows that our algorithms deliver very good solutions, particularly for larger number of radios. Note that the vertical axis of the plots is presented in log-scale for ease of viewing. The performance difference between the Tabu-based algorithm and the SDP lower bound is about 1% to 4% when the number of radios is large.

The comparison of plots for dense and sparse networks bring out interesting features. The fractional interference reduces with increase in number of radios per node; however, this trend saturates beyond a certain number of radios. This saturation point is reached with smaller number of radios for sparse networks than for dense networks, for the same number of channels. This is because the denser networks can potentially support more concurrent transmissions than the sparse networks.

ns2 Simulations: In this set of experiments, we study the impact of channel assignment in improving throughput in an 802.11-based mesh network. We compare the performance of various algorithms by measuring the *saturation throughput* using ns2 simulations over randomly generated networks. We consider networks of 50 nodes randomly placed in a 1000×1000 square meters area. The transmit power, receive and carrier sense thresholds

[3] In CLICA [21], a communication link may multiplex between multiple channels, but in our network model each communication link uses exactly one channel for transmission. We modify CLICA to use our network model.

[4] We evaluated networks of size up to 750 nodes and varying densities, with similar performance results for all algorithms. However, SDP formulation for networks of size larger than 50 nodes took unreasonably long computation time.

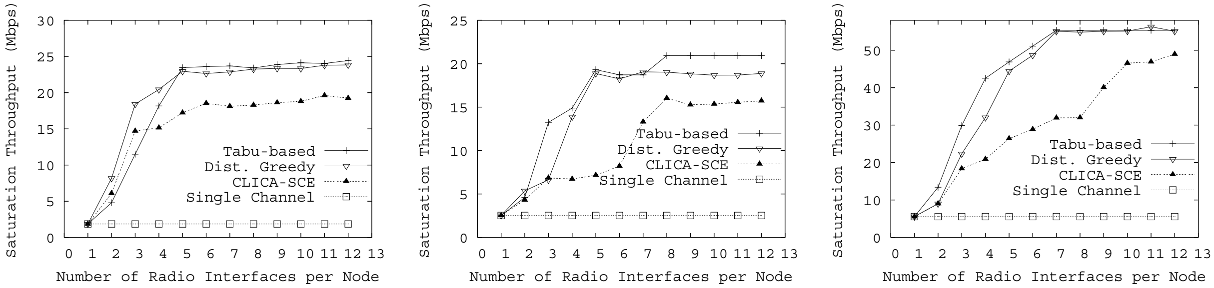


Fig. 4. Saturation throughput in ns2 simulations for 12 channels and various traffic models, viz., (a) Single hop, (b) Multi-hop Peer-to-Peer, (c) Multi-hop Gateway.

in the default setting of ns2 are such that the transmission range is 250 meters and the interference range is 550 meters. We used the default radio parameters as in ns2, except that we set the channel data rate to 24Mbps. All transmissions are unicast transmissions following the 802.11 MAC protocol with RTS/CTS, and the packet size is fixed to 1000 bytes. We use three different traffic models.

Single-hop traffic model: This model consists of identical poisson traffic for each communication link. The single-hop traffic model is useful to evaluate the performance in the case when all links in the network carry the same load.

Multi-hop peer-to-peer traffic model: In this model, 25 randomly selected source-destination pairs communicate using multihop routes. The routes are computed statically using the shortest number of hops as the metric, and do not change for the lifetime of the simulation.

Multi-hop gateway traffic model: In this model, 4 random nodes are selected as gateways, and 25 source nodes send traffic to their nearest (in terms of hops) gateway. Routes are determined as in the previous traffic model. Such a traffic model will be common when the mesh network is used for Internet gateway connectivity. Note that in the last two traffic models the traffic on the links is non-uniform.

Figure 4 plots *saturation throughput* against number of radio interfaces per node for the three traffic models and 12 channels (as we are experimenting with an 802.11a like system). We obtain the saturation throughputs as follows. For a particular number of radios and channels, we run a series of simulations, increasing the offered load each time, starting from a low value. We stop when the throughput does not increase any further with increase in the offered load.

We note that in all the three traffic models, our algorithms perform very well. We also see that the observations we made from the earlier graph-theoretic evaluations translate well into the ns2 results. The saturation throughput remain same after a certain number of radios, as inferred in the graph-theoretic simulations. Also, the relative performance of the algorithms in the ns2 simulations is the same as observed in the graph-theoretic simulations. This indirectly establishes the merit of the chosen interference model, optimization objective, and use of graph-theoretic measures as a method of performance evaluation.

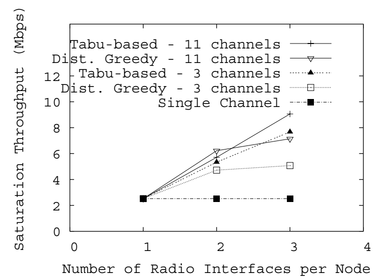


Fig. 5. Saturation throughput in ns2 simulations when using non-orthogonal channels with 802.11b-like multi-channel model (11 channels with varying degrees of interference; 3 channels are mutually orthogonal).

Modeling Non-Orthogonal Channels: So far, we have used only perfectly orthogonal channels. This however is a limitation in systems such as 802.11b where few orthogonal channels are available. Here, we assume an 802.11b like system where there are 11 channels, with only 3 of them being mutually orthogonal. For modeling the interference between non-orthogonal channels, we follow the technique outlined in Section VII. We use the data from [13] to model the "weighted" nature of conflicts. This data is obtained based on a simple analysis of the amount of overlapped spectrum between every pair of channels in 802.11b.

We use the peer-to-peer multihop traffic model (as defined before) to show the performance of our algorithms with non-orthogonal channels. See Figure 5. We observe that both our algorithms perform better when using all available 11 channels than when using only the 3 mutually orthogonal channels. The factor of improvement is less in the Tabu-based algorithm compared to the Distributed Greedy algorithm due to the inefficiency of the merge operations. Overall, use of non-orthogonal channels is a better choice than restricting channel assignments to only orthogonal channels.

IX. Conclusion

In this paper, we have formulated and addressed the channel assignment problem in multichannel wireless mesh networks where each node may be equipped with multiple radios. We have presented centralized and distributed algorithms that assign channels to communication links in the network with the objective of minimizing network interference. Using a semidefinite programming formulation of our optimization problem, we obtain a tight lower bound on the optimal network interference, and empirically demonstrate the goodness of the quality of solutions delivered by our algorithms. Using simulations on *ns2*, we observe the effectiveness of our approaches in improving the network throughput. One of the future directions is to consider assignment of multiple channels to each link.

References

[1] A. Adya, P. Bahl, J. Padhye, A. Wolman, and L. Zhou. A Multi-Radio Unification Protocol for IEEE 802.11 Wireless Networks. In *Broadnets*, 2004.

[2] Ian F. Akyildiz, Xudong Wang, and Weilin Wang. Wireless mesh networks: a survey. *Comput. Netw. ISDN Syst.*, 47(4), 2005.

[3] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal of Optimization*, 5, 1995.

[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 *MOBICOM*, 2004.

[5] Steven J. Benson and Yinyu Ye. DSDP5: Software for semidefinite programming. Technical report, 2005.

[6] R. Chandra, P. Bahl, and P. Bahl. MultiNet: Connecting to multiple IEEE 802.11 networks using a single wireless card. In *INFOCOM*, 2004.

[7] A. Coja-Oghlan, C. Moore, and V. Sanwalani. MAX k-CUT and Approximating the Chromatic Number of Random Graphs. In *ICALP*, 2003.

[8] A. Das, H. Alazemi, R. Vijayakumar, and S.Roy. Optimization Models for Fixed Channel Assignment in Wireless Mesh Networks with Multiple Radios. In *SECON*, 2005.

[9] A. Frieze and M. Jerrum. Improved approximation algorithms for MAX k-CUT and MAX BISECTION. *Algoritmica*, 18, 1997.

[10] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of ACM*, 42(6), 1995.

[11] P. Gupta and P. R. Kumar. The Capacity of Wireless Networks. *IEEE Transactions on Information Theory*, 46(2), 2000.

[12] A. Hertz and D. de Werra. Using tabu search techniques for graph coloring. *Computing*, 39(4), 1987.

[13] Cirond Technologies Inc. Channel Overlap Calculations for 802.11b Networks. http://www.cirond.com/White_Papers/FourPoint.pdf, 2002. White Paper.

[14] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of Interference on Multi-hop Wireless Network Performance. In *MOBICOM*, 2003.

[15] P. Kyasanur and N.H.Vaidya. Capacity of Multi-Channel Wireless Networks: Impact of Number of Channels and Interfaces. In *MOBICOM*, 2005.

[16] P. Kyasanur and N.H. Vaidya. Routing and Link-layer Protocols for Multi-Channel Multi-Interface Ad Hoc Wireless Networks. *ACM SIGMOBILE MC2R*, 10(1):31–43, 2006.

[17] S. Liese, D. Wu, and P. Mohapatra. Experimental Characterization of an 802.11b Wireless Mesh Network. Technical report, University of California, Davis, 2005.

[18] M.Alichery, R.Bhatia, and L.Li. Joint Channel Assignment and Routing for throughput optimization in Multi-Radio wireless mesh networks. In *MOBICOM*, 2005.

[19] Meru Networks. http://www.merunetworks.com/index.shtml.

[20] A. Mishra, E. Rozner, S. Banerjee, and W. Arbaugh. Exploiting partially overlapping channels in wireless networks: Turning a peril into an advantage. In *(IMC)*, 2005.

[21] M.K.Marina and S.Das. A Topology Control Approach to Channel Assignment in Multi-Radio Wireless Mesh Networks. In *Broadnets*, 2005.

[22] M.Kodialam and T.Nandagopal. Characterizing the Capacity Region in Multi-Radio, Multi-Channel Wireless Mesh Networks. In *MOBICOM*, 2005.

[23] R. Montemanni, D.H. Smith, and S.M. Allen. Lower bounds for fixed spectrum frequency assignment. *Annals of Operations Research*, 107, October 2001.

[24] J. Padhye, S. Agarwal, V.N. Padmanaban, L. Qiu, A. Rao, and B. Zill. Estimation of link interference in static multi-hop wireless networks. In *(IMC)*, 2005.

[25] K. Ramachandran, E. Belding, K. Almeroth, and M. Buddhikot. Interference-aware channel assignment in multi-radio wireless mesh networks. In *INFOCOM*, 2006.

[26] A. Raniwala, K. Gopalan, and T. Chiueh. Centralized Channel Assignment and Routing Algorithms for Multi-Channel Wireless Mesh Networks. *ACM SIGMOBILE MC2R*, 8(2), 2004.

[27] R. Raniwala and T. Chiueh. Architechture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network. In *INFOCOM*, 2005.

[28] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Measurement-based models of delivery and interference in static wireless networks. In *SIGCOMM*, 2006.

[29] J. Robinson, K. Papagiannaki, C. Diot, X. Guo, and L. Krishnamurthy. Experimenting with a multi-radio mesh networking testbed. In *(WiNMee Workshop)*, 2005.

[30] J. Shi, T. Salonidis, and E.W Knightly. Starvation Mitigation Through Multi-Channel Coordination in CSMA Multi-hop Wireless Networks. In *MOBIHOC*, 2006.

[31] J. So and N.H. Vaidya. Multi-Channel MAC for Ad Hoc Networks: Handling Multi-Channel Hidden Terminals Using A Single Transceiver. In *MOBIHOC*, 2004.

[32] J. Tang, G. Xue, and W. Zhang. Interference-Aware Topology Control and QoS Routing in Multi-Channel Wireless Mesh Networks. In *MOBIHOC*, 2005.

[33] The CoMo project. http://como.intel-research.net/.

[34] S.-L. Wu, C.-Y Lin, Y.-C. Tseng, and J.-P. Sheu. A new multi-channel MAC protocol with on-demand channel assignment for multi-hop mobile ad hoc networks. In *(ISPAN)*, 2000.