

Minimum-Interference Channel Assignment in Multi-Radio Wireless Mesh Networks

Anand Prabhu Subramanian
Computer Science Department
Stony Brook University
Stony Brook, NY 11794-4400, U.S.A
Email: anandps@cs.sunysb.edu

Contents

0.1	Introduction	1
0.2	Problem Formulation	3
0.3	Related Work	6
0.4	Tabu-based Algorithm	8
0.5	Greedy Algorithm	11
0.6	Bounds on Optimal Network Interference	12
	0.6.1 Linear Programming Formulation	13
	0.6.2 Semidefinite Programming Formulation	15
0.7	Generalizations	18
0.8	Performance Evaluation	20
	0.8.1 Graph-Theoretic Performance Metric	20
	0.8.2 ns2 Simulations	23
0.9	Conclusion	26

List of Figures

1	Communication graph and corresponding conflict graph. . . .	4
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.	10
3	Fractional network interference of solutions delivered by various algorithms compared with the lower bounds in (a) Dense network: 3 channels, (b) Dense network: 12 channels (c) Sparse network: 3 channels and (d) Sparse network: 12 channels.	21
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.	24
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).	25

Abstract

Multihop wireless mesh networks (WMNs) are increasingly used to provide broadband wireless access to mobile users. The WMNs form a wireless backbone network to which mobile users get connected and get services like internet connectivity. It is well known that capacity of a multi-hop network is seriously affected by wireless interference. Since WMNs are used as backbone networks, high capacity is desired. In order to get high capacity, network interference must be minimized as much as possible. Use of multiple non-interfering channels is one way of mitigating wireless interference.

In this work, we consider static 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 at most the number of radio interfaces on that node. The above optimization problem is known to be NP-hard.

We design two efficient algorithms that are empirically shown to perform very well with respect to bounds on the optimal. To evaluate the quality of the solutions obtained by our algorithms, we develop a linear program and a semidefinite program formulation of our optimization problem to obtain lower bounds on overall network interference. Empirical evaluations on randomly generated network graphs show that our algorithms perform close to the above established lower bounds, 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.

0.1 Introduction

Wireless mesh networks [2] are multihop networks of wireless routers. There is an increasing interest in using wireless mesh networks as broadband backbone networks to provide ubiquitous network connectivity in enterprises, campuses, and in metropolitan areas. Mesh networks based on commodity 802.11 hardware are becoming more common and commercially deployed (see, for example, [7]) due to the popularity of the 802.11 standard and ubiquity of 802.11-based interfaces in mobile devices.

Mesh networks are similar to ad hoc networks in the sense as they both deal with multihop wireless links. However, there are a few major differences. Firstly, the routers in the WMNs do not have power constraints as they can be powered from power outlets or batteries that recharge automatically. Secondly, the nodes are rarely mobile and the network has very infrequent topology changes, limited node failures, etc. Finally, node addition and maintenance are quite rare events.

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 [15]. Use of multiple channels provide a simple opportunity to perform bandwidth aggregation to increase capacity [19]. The basic idea is to use orthogonal (non-interfering) channels for neighboring wireless transmissions. The current 802.11 standard indeed provides several orthogonal channels to facilitate this.

The research community has started exploring how best to utilize multiple channels in a multihop wireless network. A set of protocols have been proposed that either design new MAC protocols [33] to exploit multiple channels or extend the 802.11 standard to handle multiple channels [31] effectively. Protocols have also been proposed [4] that do not need changes the standard. However, all these protocols fundamentally rely on fast channel switching capability on the part of the interface hardware, often in the time scale of packet transmissions. This requirement makes it hard to make effective use of these strategies on commodity 802.11-based hardware, where channel switching delays are in the order of milliseconds.

An alternative approach is to use multiple radio interfaces on each router and assign different channels to these interfaces or network links [1, 22, 28, 29]. This makes easy for a router to utilize multiple channels without any requirement of fast channel switching. This strategy can easily be used with commodity hardware. However, since the number of interfaces on a router may be less than the number of available channels, effective techniques must be developed for channel assignment so that network capacity is maximized.

The channel assignment here can be static or dynamic. However, dynamic schemes are limited to changes on a long time scale so that all overheads (e.g., high channel switching delays) can be ignored. The changes in channel assignment may be prompted by significant changes in traffic or network topology. These channel assignment approaches use only “averaged” notion of network parameters such as traffic or interference.

Approach. We pursue the above multi-radio approach to exploit multiple channels and consider the problem of assigning channels to communication links to minimize the overall network interference. This is based on a given interference model describing how two given links interfere on average. Such interferences on link pairs can be estimated via measurements [25] making our approach practical and applicable in real commodity platforms. For much of our work we consider a centralized channel assignment, though we do develop a distributed algorithm in one case. Centralized algorithms are quite practical in “managed” mesh networks, where there is already a central management entity. Aggregated traffic information can be collected to this central system periodically and then channel assignment is computed based on the pre-evaluated interference model. As we mentioned before, the periodicity can be adjusted to keep the overhead low, though it is not our interest to evaluate this tradeoff here.

While channel assignment can be modeled as some variation of a graph coloring problem, it has an interesting twist in the context of mesh networks. The assignment of channels to links must obey the 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. We refer to this as *interface constraint*. Earlier research has shown that the above problem is NP-hard [22]. Thus, efficient algorithms that run reasonably fast and provide good quality solutions are of interest.

Contributions. As a contribution, we develop two such algorithms. One is based on a popular heuristic search technique called Tabu search [16] that has been used in the past in graph coloring problems. The second approach is a greedy approach motivated by the greedy approximation algorithm for Max K-cut problem in graphs [11]. To evaluate their performances, we formulate our channel assignment problem using mathematical programming approaches. In particular, we develop two formulations, using integer linear programming (ILP) and a semidefinite programming (SDP). We obtain *bounds* on the optimal solution by relaxing the ILP and SDP formulations to run in polynomial time. Finally, detailed ns-2 simulation studies demonstrate the full performance potential of the channel assignment algorithms

in 802.11 based multi-radio mesh networks.

For clarity of presentation, for the most part, we assume uniform traffic, binary interference model (two links either interfere or do not interfere), and orthogonal channels. However, in Section 0.7, we show that our techniques can be generalized to more practical, intermediate situations (fractional interference, non-uniform traffic, and non-orthogonal channels [21]).

The rest of the report is organized as follows. We start with describing the network model and the formulation of our problem in Section 0.2, and discuss related work in Section 0.3. We present our algorithms in Section 0.4 and Section 0.5 respectively. In Section 0.6, we obtain lower bounds on the optimal network interference using linear and semidefinite programming. Section 0.7 presents the above mentioned generalizations, and Section 0.8 presents simulation studies.

0.2 Problem Formulation

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

Network Model. We consider a wireless mesh network with stationary wireless routers where each router is equipped with a certain (not necessarily same) number of radio interfaces. We model the *communication graph* of the network as a general undirected graph over the set of network nodes (routers). 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 a certain number of channels available in the network. 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 0.7.

Interference Model. Due to the broadcast nature of the wireless links, transmission along a communication link (between a pair of wireless nodes) may interfere with transmissions along other communication links in the network. Two interfering links cannot engage in successful transmission at the same time if they transmit on the same channel. 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 [15, 18, 23]. The discussion in this report is independent of the specific interference model used as long as the interference model is defined on pairs of communication

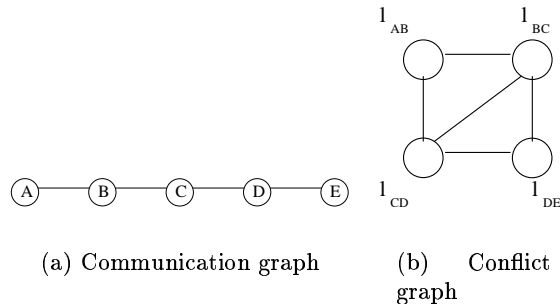


Figure 1: Communication graph and corresponding conflict graph.

links. For clarity of presentation, we assume a *binary interference model* for now (i.e., two links either interfere or do not interfere). We generalize our techniques to fractional interference model in Section 0.7.

Effect of Traffic. Since our goal is to assign channels to links on a much larger time scale than on a per-packet basis, the interference between links must be represented in a “time averaged” fashion rather than instantaneous. Since two links interfere only when they are involved in a packet transmission, the level of interference between two links depends on the actual traffic on the links. However, for clarity of presentation, we assume uniform traffic on all links for now; we extend our techniques for non-uniform traffic in Section VII.

Conflict Graph. Given an interference model, the set of pairs of communication links that interfere with each other (assuming them to be on the same channel) can be represented using a *conflict graph* [18]. 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}) in the conflict graph 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. As defined above, the conflict graph does not change with the assignment of channels to vertices in the conflict graph.

We illustrate the concept of conflict graph in Figure 1. The wireless network represented in Figure 1 has five network nodes A, B, \dots, E and four communication links as shown in the communication graph (see Figure 1(a)). The conflict graph (see Figure 1(b)) has four nodes each representing a

communication link in the network. 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. Thus, the communication link (A, B) interferes with the communication links (B, C) and (C, D) , and not with (D, E) .

Notations. Here, we introduce some notations that we use throughout this report.

- N , the set of network nodes in the network.
- R_i , the number of radio interfaces on node $i \in N$.
- $\mathcal{K} = \{1, 2, \dots, K\}$, the set of K channels.
- $V_c = \{l_{ij} \mid (i, j) \text{ is a communication link}\}$.
- $G_c(V_c, E_c)$, the conflict graph of the network.
- For $i \in N$, $E(i) = \{l_{ij} \in V_c\}$, i.e., $E(i)$ is set of vertices in V_c that represent the communication links incident on node i .

In addition, throughout this report, we use variables u, v to refer to vertices in V_c , variables i, j, a, b to refer to nodes in N , and the variable k to refer to a channel. Since assigning channel can be thought of as coloring vertices, we use the terms channel and colors interchangeably throughout our report.

Channel Assignment Problem. The problem of channel assignment in a multi-radio wireless mesh network can be informally described as follows. Given a mesh network of router nodes with multiple radio interfaces, we wish to assign channels to the communication links¹ in the network such that the number of different channels assigned to the links incident on any node is at most the number of radios on that node. Since we assume uniform traffic on all links for now, we assign channels to all links, and the total interference in the network can be defined as the number of pairs of interfering communication links. The objective of our problem is to minimize the overall network interference (as defined above).

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*.

¹Note that merely assigning channels to radios is not sufficient to measure network interference/capacity, since a link still can use one of many channels for transmission.

Interface Constraint.

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

Network Interference $I(f)$.

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

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. As shown below, the channel assignment problem is NP-hard.

Relationship with Max K -cut. Given a graph G , the Max K -cut problem [11] 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.

0.3 Related Work

The use of multiple channels to increase capacity in a multihop network has been addressed extensively. Generally, there have been two types of approaches, viz., (i) Fast switching of channels (possibly, on a per-packet basis) on a single radio, or (ii) Assigning channels to radios for an extended period of time in a multi-radio setting.

Fast Switching of Channels. In MMAC protocol [31], the authors augment the 802.11 MAC protocol such that the nodes meet at a common channel periodically to negotiate the channels to use for transmission in the next phase. In SSCH [4], the authors propose dynamic switching of channels using pseudo-random sequences. The idea is to randomly switch channels such that the neighboring nodes meet periodically at a common channel to communicate. In DCA [33], the authors use two radios - one for the control packets (RTS/CTS packets) and another for data packets. The channel to send the data packet is negotiated using the control packets and the data packets are sent in the negotiated channels.

All the above protocols require a small channel switching delay (of the order of hundred microseconds or less), since channels are switched at a fast time scale (possibly, on a per-packet basis). We have observed a channel switching delay of the order of milliseconds in commodity 802.11 wireless cards, as channel switching requires a firmware reset and execution of an associated procedure. Similar experiences were reported in [6]. Thus, the above described protocols (MMAC, SSCH, and DCA) cannot be easily implemented in commodity 802.11 platforms due to high switching latency therein.

Channel Assignment in Multiple Radio Setting. In case of multiple radio interfaces, channels can be assigned statically to interfaces or to links for an extended period of time as long as the aggregate traffic demand or topology does not change to a significant extent. The assignment can be recomputed when they change. This solution is deemed more practical as there is neither a need to modify the 802.11 protocol or need for interfaces with very low channel switching latency. Adya et al. [1] propose a channel assignment strategy where the interface constraint is ignored and the channel to use for communication is determined via a measurement-based approach. In [28, 29], Raniwala and Chiueh propose centralized and distributed load-aware channel assignment and routing algorithms. In [29], the authors assume a tree-based communication pattern to ease coordination for optimizing channel assignment. This might lead to inefficient channel assignment and routing in a more general peer-to-peer communication. They also do not quantify the performance of their solutions with respect to the optimal. In [27], a purely measurement-based approach is taken for multi-radio channel assignment. Here, one interface is always tuned to a common channel so that the topology is always preserved. This can be wasteful specially in case only a few interfaces are available.

In the most closely related work to ours, Marina and Das in [22] 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 [9], 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.

Other Related Works. In other related works, [20, 23] 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. Authors in [19] derive upper bounds on capacity of wireless multihop networks with multiple channels. In [26], the authors propose a hybrid channel assignment scheme in which one radio on each node is assigned a fixed channel to receive packets, while the other radio is dynamically assigned to different channels to transmit packets.

0.4 Tabu-based Algorithm

In this section, we describe one of our algorithms for the channel assignment problem, based on the Tabu search [16] technique for coloring vertices in graphs.

Algorithm Overview. 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/function $f : V_c \rightarrow \mathcal{K}$ 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 Tabu search based technique [16] 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. In the 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 $f_0, f_1, f_2, \dots, f_j, \dots$, 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. Given a solution f_j , we create f_{j+1} as follows. 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 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. A formal description of the first phase follows.

Input : Conflict Graph $G_c(V_c, E_c)$; Set of channels \mathcal{K} .

Output: Channel Assignment Function $f_{best} : V_c \rightarrow \mathcal{K}$.

```

Start with a random assignment function  $f_0$ ;
 $f_{best} = f_0$ ;  $I_{best} = I(f_0)$ ;  $\tau = null$ ;  $j = 0$ ;  $i = 0$ ;
while  $I(f_j) > 0$  and  $i \leq i_{max}$  do
  Generate  $r$  random neighbors of  $f_j$ ;
  Each neighbor is generated by randomly picking
  a  $u$  in  $V_c$  and  $k \in \mathcal{K}$  s.t.  $k \neq f_j(u)$  and  $(u, k) \notin \tau$ ,
  and changing  $f_j(u)$  to  $k$ 
  Let  $f_{j+1}$  be the neighbor with lowest interference.
  Add  $(u, f_j(u))$  to  $\tau$ .
  If  $\tau$  is full, delete its oldest entry;
  if  $(I(f_{j+1}) < I_{best})$ 
    then  $I_{best} = I(f_{j+1})$ ;  $f_{best} = f_{j+1}$ ;  $i = 0$ ;
    else  $i = i + 1$ ;
  endif;
   $j = j + 1$ ;
end while
RETURN  $f_{best}$ ;

```

Algorithm 1: First Phase of Tabu-based Algorithm.

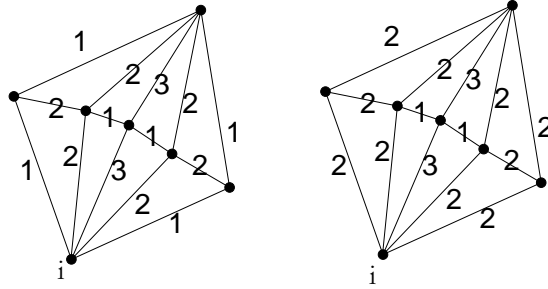


Figure 2: Merge operation of second phase. The two figures are the communication graphs of the network before and after the merge operation. Labels on the links denote the color/channel. Here, the merge operation is started at node i by changing all its 1-colored links to color 2.

Second Phase. Note that the solution f returned by the first phase may violate interface constraints. Thus, in the second phase, we eliminate the interface constraints by repeated application of the following “merge” procedure. Given a channel/color 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, i.e, whose number of radios is less than the number of distinct colors assigned to the incident communication links, we pick the node wherein the difference between the above two terms is the 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.

0.5 Greedy Algorithm

In this section, we describe our Greedy algorithm for the channel assignment problem. Our choice of greedy approach is motivated by the following two observations. Here, $G_{n,p}$ graphs are defined as random graph over n vertices where each edge exists with a uniform probability of p .

- In [8], the authors consider the Max K -cut problem in $G_{n,p}$ graphs, and show that the greedy heuristic that greedily decides the partition of one vertex at a time delivers a $(1 - \frac{1}{Kx})$ -approximate solution with very high probability where $x > 1$.
- We can show that the conflict graph corresponding to a random network is a $G_{n,p}$ graph, under the protocol interference model [15]).

Greedy Algorithm for Channel Assignment. The above two observations motivate use of a greedy approach for our channel assignment problem. 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 total time complexity of the Greedy algorithm is $O(dK|V_c|^3)$.

Distributing the Greedy Algorithm. The above described Greedy approach 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 be 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*² 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 $O(|V_c|K)$. The above algorithm is localized, and can be made to work in dynamic topologies.

0.6 Bounds on Optimal Network Interference

In this section, we derive lower bounds on the minimum network interference using linear and semidefinite programming approaches. These lower bounds will aid in understanding the quality of the solutions obtained from the algorithms presented in previous two sections.

²Such 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.

0.6.1 Linear Programming Formulation

Here, we formulate our channel assignment problem as an integer linear program (ILP), and use the relaxed linear programming with additional constraints to estimate the lower bound on the optimal network interference.

Integer Linear Programming. We use the following set of binary integer (taking values 0 or 1) variables and constraints in our ILP formulation.

- Variables Y_{uk} , for each $u \in V_c$ and $k \in \mathcal{K}$. The variable Y_{uk} is 1 if and only if the vertex $u \in V_c$ is assigned the channel k . Essentially, the variables Y_{uk} define the channel assignment function. Since, each vertex in V_c is given exactly one channel, we have the following constraints.

$$Y_{uk} = \{0, 1\}, \quad \forall u \in V_c, \forall k \in \mathcal{K} \quad (2)$$

$$\sum_{k \in \mathcal{K}} Y_{uk} = 1, \quad \forall u \in V_c \quad (3)$$

- Variables X_{uv} , for each edge $(u, v) \in E_c$. The variable X_{uv} is 0 only if the vertices $u, v \in V_c$ are assigned different channels.³ The following equation defines the value of X_{uv} in terms of Y variables.

$$X_{uv} = \{0, 1\}, \quad \forall (u, v) \in E_c \quad (4)$$

$$X_{uv} \geq Y_{uk} + Y_{vk} - 1, \quad \forall (u, v) \in E_c, \forall k \in \mathcal{K} \quad (5)$$

The variables X_{uv} are used to define the network interference (the objective function defined later).

- Variables Z_{ik} , for each network node $i \in N$ and channel $k \in \mathcal{K}$. The variable Z_{ik} is 1 if and only if some $u \in E(i)$ has been assigned a channel k ; note that, u represents a communication link incident on $i \in N$.

$$Z_{ik} = \{0, 1\}, \quad \forall i \in N, \forall k \in \mathcal{K} \quad (6)$$

$$Z_{ik} \geq Y_{uk}, \quad \forall u \in E(i), \forall i \in N, \forall k \in \mathcal{K} \quad (7)$$

$$Z_{ik} \leq \sum_{u \in E(i)} Y_{uk}, \quad \forall i \in N, \forall k \in \mathcal{K} \quad (8)$$

³If vertices u and v in V_c are assigned same channel, then X_{uv} can be 0 or 1. However, X_{uv} will be chosen to be 0 to minimize the objective function (see below), as there are no additional constraints involving X_{uv} . The additional constraints in Equation 11 and 12 can be looked upon as derivations of Equation 5.

The last equation above is used to enforce that Z_{ik} is 0 if there is indeed no vertex $u \in E(i)$ that has been assigned a channel k . The below equation enforces the interface constraint using Z variables.

$$\sum_{f=1}^k Z_{if} \leq R_i \quad \forall i \in N \quad (9)$$

Objective Function. Our objective function for the above ILP is to

$$\text{Minimize } \sum_{(u,v) \in E_c} X_{uv}.$$

Linear Programming. Due to NP-hardness of integer linear programming, solving the above ILP is intractable for reasonably sized problem instances. Thus, we relax the above ILP to a linear program (LP) by relaxing the integrality constraints. In particular, we replace the Equations 4, 2, and 6 by the following equation.

$$0 \leq X_{uv}, Y_{uk}, Z_{ik} \leq 1.$$

The solution to the relaxed linear program gives only a lower bound on the optimal solution to the ILP. Through simulations, we have observed that the lower bound obtained by the above LP formulation is very loose. Thus, in order to obtain a tighter lower bound, we add additional constraints as follows.

Clique Constraint. For each vertex $u \in V_c$, let S_u be the set of vertices in a maximal clique containing u . It can be shown [24] that the number of monochromatic edges in the complete subgraph of size $|S_u|$ when colored by K colors is at least:

$$\sigma(S_u, K) = \frac{\beta\alpha(\alpha + 1) + (K - \beta)\alpha(\alpha - 1)}{2}, \quad (10)$$

where $\alpha = \lfloor \frac{|S_u|}{K} \rfloor$ and $\beta = |S_u| \bmod K$. The above observation yields the following additional constraint.

$$\sum_{v,w \in S_u} X_{vw} \geq \sigma(S_u, K) \quad \forall u \in V_c \quad (11)$$

Since the set of vertices $E(i)$ in V_c forms a clique in G_c and uses at most R_i colors (due to the interface constraint on node i), we also have the following constraint.

$$\sum_{(u,v) \in E(i)} X_{uv} \geq \sigma(E(i), R_i) \quad \forall i \in N \quad (12)$$

The above two additional constraints pose a lower bound on the interference on clique like subgraphs. This helps to reduce the gap between the actually integer optimum and the relaxed linear solution.

Number of Variables and Constraints. The number of variables in the above LP formulation is $|E_c| + K(|V_c| + N)$, and the total number of equations/constraints are $2(|V_c| + |N|) + K(2|V_c| + 2|N| + |E_c|)$ including the integrality constraints. We solve the linear program using GLPK [12], a public-domain MIP/LP solver.

0.6.2 Semidefinite Programming Formulation

In this section, we model our channel assignment problem in terms of a semidefinite program (SDP). In our simulations, we observed that the semidefinite programming formulation yielded a much tighter bound on the optimal network interference. However, solving the SDP formulation of channel assignment problem takes much more time and memory than the linear programming formulation, and hence, is not feasible for very large network sizes.

Semidefinite Programs. A *semidefinite program* [13] is a technique to optimize a linear function of a symmetric positive-semidefinite matrix⁴ subject to linear equality constraints. Semidefinite programs can be solved in polynomial time using various techniques [14]. The reader is referred to [3, 13] for further details on semidefinite programming and its application to combinatorial optimization. The standard form of semidefinite program is as follows.

$$\begin{aligned} & \text{Minimize} && C.X \\ & \text{such that} && A_i.X = b_i, && 1 \leq i \leq m, \text{ and} \\ & && X \succeq 0 \end{aligned}$$

where $C, A_i(\forall i)$, and X are all symmetric $n \times n$ matrices, and b_i is a scalar vector. The constraint $X \succeq 0$ implies that the variable (to be computed) matrix X must lie in the closed, convex cone of a positive semidefinite matrix. Also, $A_i.X$ refers to the standard inner product of two symmetric matrices.

Below, we start with presenting the SDP for the Max K -cut problem from [11]. We then extend it to our channel assignment problem by adding the interface constraint.

⁴A matrix is said to be *positive semidefinite* if all its eigen values are nonnegative.

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 $\{a_1, a_2, \dots, a_K\}$, where the a_i vectors are defined as follows [11]. We take an equilateral simplex Σ_K in \mathbf{R}^{K-1} with vertices b_1, b_2, \dots, b_K . Let $c_K = \frac{(b_1 + b_2 + \dots + b_K)}{K}$ 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 [11].

IP_{Max-K}:

$$\begin{aligned} \text{Maximize} \quad & \frac{K-1}{K} \sum_{(u,v) \in E_c} (1 - y_u \cdot y_v) \\ \text{such that} \quad & y_u \in \{a_1, a_2, \dots, a_K\} \end{aligned}$$

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

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

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

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

where $\sigma(E(i), R_i)$ is as defined as in Equation 10. Now, we add the following constraint to represent the interface constraint.

$$\sum_{u,v \in E(i)} y_u \cdot y_v \geq \Phi_i \quad \forall i \in N \quad (13)$$

The above equation follows from the same observations as Equation 12. In particular, 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 ($E(i)$) [24], and thus, $\left(\binom{|E(i)|}{2} - \sigma(E(i), R_i) \right)$ is an upper bound on the number of non-monochromatic edges. Since we know that $y_u \cdot y_v = 1$ for any monochromatic edge (u, v) and $y_u \cdot y_v = \frac{-1}{K-1}$ for any non-monochromatic edge, we have constraint in the above Equation 13.

Note that even though Equation 13 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 13 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 $y_u \cdot y_v$ can now take any value between 1 and -1 , we add an additional constraint to restrict $y_u \cdot y_v$ to be greater than $\frac{-1}{K-1}$. The relaxed SDP for the channel assignment is as follows.

$$\begin{aligned} & \text{Maximize} \quad \frac{K-1}{K} \sum_{(u,v) \in E_c} (1 - y_u \cdot y_v) \\ & \text{such that} \quad y_u \in R^{|V_c|} \text{ and } |y_u| = 1 \\ & y_u \cdot y_v \geq \frac{-1}{K-1}, \quad \forall u \neq v, \text{ and} \\ & \sum_{u,v \in E(i)} y_u \cdot y_v \geq \Phi_i, \quad \forall i \in N. \end{aligned}$$

Standard SDP Formulation. Now, we convert the above relaxed version into the standard SDP formulation. Let W be the $|V_c| \times |V_c|$ symmetric matrix representing the adjacency matrix of the graph G_c , and let e be the $|V_c| \times 1$ vector containing all 1's. Now, let $L = d(W.e) - W$ denote the Laplacian of the W matrix, where $d(W.e)$ is the $|V_c| \times |V_c|$ matrix with $W.e$ as the main diagonal. Finally, let

$$C = -\frac{L(K-1)}{2K},$$

X be the semidefinite $|V_c| \times |V_c|$ matrix representing $y_u \cdot y_v$ for all $u, v \in V_c$. Now, the semidefinite program for the channel assignment problem in the

standard SDP form can be represented as follows.

$$\begin{aligned}
& \text{Minimize} && C.X \\
& \text{such that} && \\
& \text{diagonal}(X) = e && \\
& X_{u,v} \geq \frac{-1}{K-1}, \quad \forall u \neq v \in V_c, && \\
& A_i.X \geq 2\Phi_i, \quad \forall i \in N, \text{ and} && \\
& X \succeq 0, &&
\end{aligned}$$

where each $A_i (i \in V)$ is a $|V_c| \times |V_c|$ matrix representing $E(i)$. In particular, the $A_i[u, v] = 1$ if $(u, v) \in E_i$, and 0 otherwise. Also, the inequalities in the above constraints can be converted into equalities by subtracting linear positive variables from the left hand side.

The solution to the above semidefinite program gives an upper bounds on the number of non-monochromatic edges, and the lower bound on the optimal network interference can be obtained by subtracting it from $|E_c|$.

0.7 Generalizations

In the previous sections, for sake of clarity, we made various assumptions, viz., uniform traffic on all communication links, a binary interference model, and orthogonal channels. In this section, we generalize our techniques to relax these assumptions. These generalizations are quite 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.) [30]. While great strides have been made in RF device engineering so that radios on orthogonal channels do not interfere even when co-located [10], little public data is available about the performance of these devices. 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 [21]. 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. Note that in our network model, we assume that the traffic is known a priori. Based on the above notations, the overall network interference for a given channel assignment function $f : V_c \rightarrow \mathcal{K}$ 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 LP and SDP formulations of the channel assignment problem can be generalized by appropriately extending the objective function; no other changes are required in the list of variables and constraint equations.

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 overall network network can be further generalized as follows for a given channel assignment function $f : V_c \rightarrow \mathcal{K}$.

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

As before, Tabu-based and Greedy algorithms can use the above definition of network interference without any additional changes. However, in the LP formulation, we need to replace the Equations 4 and 5 by the following equations.

$$\begin{aligned} 0 \leq X_{uv} \leq 1, & \quad \forall (u, v) \in E_c \\ X_{uv} \geq Y_{uk_1} + Y_{vk_2} - 2 + c(k_1, k_2), & \quad \forall (u, v) \in E_c, \forall k_1, k_2 \in \mathcal{K} \end{aligned}$$

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 $a_i \cdot a_j$ is proportional to $c(i, j)$ for all channels $i, j \in \mathcal{K}$.

Measurement of Parameters. Much of the new parameters introduced in this section must be measured via some sort of monitoring function in a real testbed. The traffic on each link must be directly measured or estimated via some indirect means. The functions such as $r(\cdot)$ and $c(\cdot)$ also need to be estimated via measurements. A recent paper provides some idea how such measurements can be done [25]. The measured parameters must then be fed into the channel assignment algorithm.

0.8 Performance Evaluation

In this section, we study the performance of our designed algorithms for the channel assignment problem through extensive simulations. 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. We start with discussing various algorithms used for comparison.

Algorithms. In addition to our designed algorithms (Tabu-based and Greedy) and the lower bounds obtained from the linear and semidefinite programming techniques, we also present results for two other algorithms for comparison. In particular, we simulate a modified version of the CLICA heuristic presented in [22] for a slightly different version of the channel assignment problem.⁵ We refer to the modified algorithm of [22] as CLICA-SCE. 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.

We note here the network interference metric is actually a localized metric since a communication link interferes with only “neighboring” communication links. Thus, we observed similar performance of the centralized and distributed versions of the Greedy algorithm.

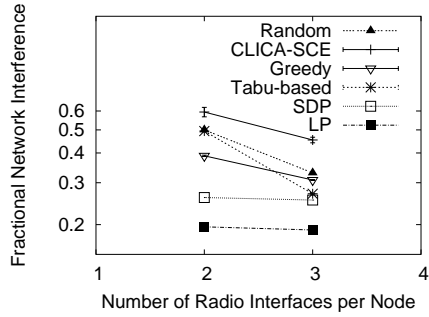
0.8.1 Graph-Theoretic Performance Metric

In this set of experiments, we generate random networks by randomly placing a number of nodes in a fixed region, and evaluate various algorithms based on a certain graph-theoretic performance metric. To solve linear programs, we used GLPK [12] which is a public-domain MIP/LP solver, while to solve semidefinite programs, we used DSDP 5.0 [5] which uses an efficient interior-point technique.

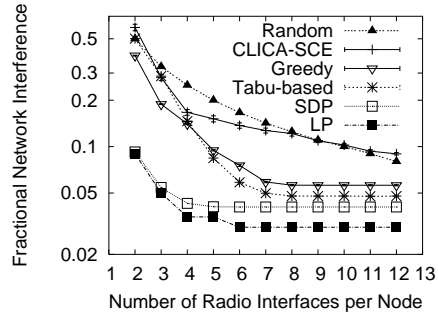
Graph Parameters. We consider two sets of random network, viz., dense and sparse networks, generated by randomly placing 50 nodes in 500×500 and 800×800 square meters of area respectively.⁶ In dense networks, the

⁵In CLICA [22], 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.

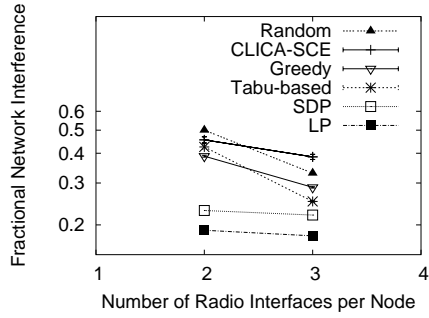
⁶We evaluated networks of size up to 750 nodes and varying densities, with similar performance results for all algorithms. However, the LP and SDP formulations for networks of size larger than 50 nodes took unreasonably long computation time.



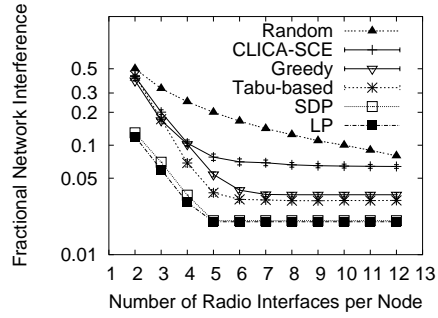
(a) Dense Network : 3 Channels



(b) Dense Network: 12 Channels



(c) Sparse Network : 3 Channels



(d) Sparse Network: 12 Channels

Figure 3: Fractional network interference of solutions delivered by various algorithms compared with the lower bounds in (a) Dense network: 3 channels, (b) Dense network: 12 channels (c) Sparse network: 3 channels and (d) Sparse network: 12 channels.

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. Two nodes are connected by a communication link if they lie within each other’s *transmission range*. Also, two communication links (i, j) and (g, h) interfere with each other if and only if either g or h lies within the *interference range* of i or j ; this is based on the protocol interference model [15]. We assume orthogonal channels and uniform traffic on all links.

Performance Metric. 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. The fractional network interference for the random algorithm is given by $\frac{1}{R}$, where R is the number of radios on each node. Note that the above performance metric is purely graph-theoretic and hence, we do not use any network simulator for these experiments.

Results. 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 Greedy algorithm, except when the number of radios is very small. When the number of radios is very small, the second phase of Tabu-based algorithm is forced to perform many inefficient merge operations which leads to performance degradation.

The performance of our algorithms compared to the lower bounds obtained from the LP and SDP formulations 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. We can also see that the SDP formulation delivers a much better lower bound than the LP formulation, for all parameter values. However, as we noted before, running SDP is significantly more computationally expensive (in terms of time and memory) than LP.

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. Similar trends were observed in [22].

0.8.2 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 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 same default radio parameters as in ns2 [32], 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.

Performance for Various Traffic Models. 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. The traffic information is used in the channel assignment algorithms as suggested in Section 0.7.

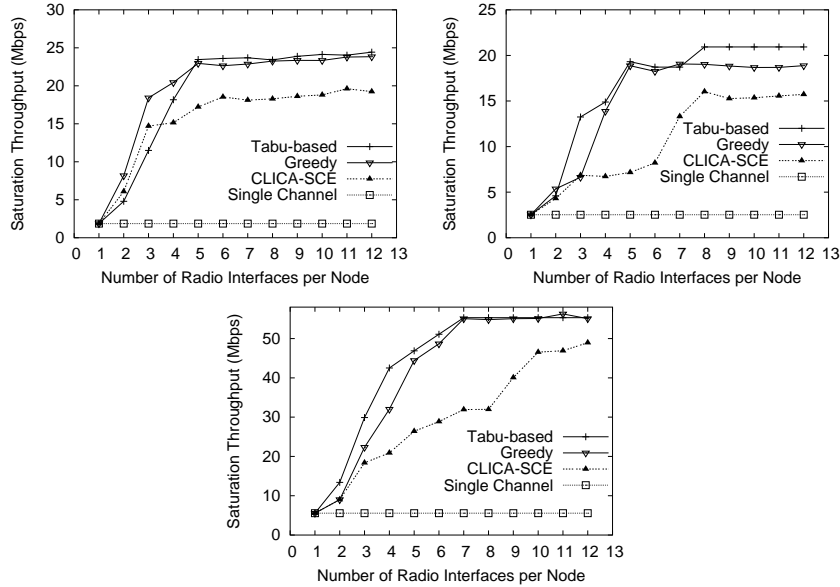


Figure 4: Saturation throughput in ns2 simulations for 12 channels and various traffic models, viz., (a) Single hop, (b) Multi-hop Peer-to-Peer, (c) Multi-hop Gateway.

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.

Modeling Non-Orthogonal Channels. So far, we have used only perfectly orthogonal channels. This however is a limitation in systems such as

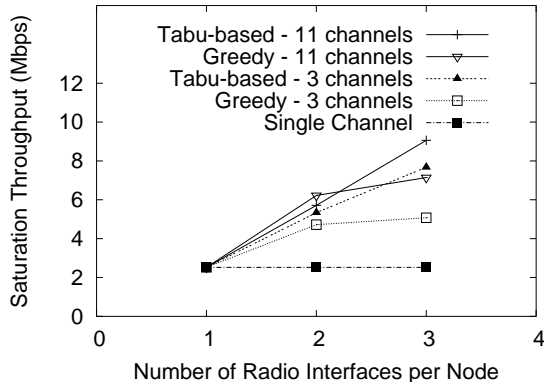


Figure 5: Saturation throughput in ns2 simulations when using non-orthogonal channels with 802.11b-like multi-channel model (11 channels with varying degrees of interference; 3 channels are mutually orthogonal).

802.11b where few orthogonal channels are available. Since our techniques are general enough to handle non-orthogonal channels (Section VII), we now model a non-orthogonal channel situation.

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 [17] 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 also did direct measurements on an 802.11b testbed to estimate interference between non-orthogonal channels and the values we obtained are similar to those quoted in [17]. Since such measurements can be very much hardware and environment specific, we stick to the data in [17].

In the ns2 simulator, we model inter-channel interference as follows. Physical layer frames transmitted on channel k_1 arriving at a radio interface tuned to channel k_2 are reduced in power depending on the degree of non-interference. For example, if a k_1 -frame arrives at a k_1 -interface, the frame does not undergo any power reduction. On the other hand, if a k_1 -frame arrives at a k_2 -interface, where k_1 and k_2 are perfectly orthogonal, then the k_1 -frame is completely silenced. Power reduction between 0% and 100% occur for other intermediate cases. In the simulator, the interference (e.g., carrier-sense or collisions) is calculated only after such power reduction.

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 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.

0.9 Conclusion

In this work, 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 two algorithms that assign channels to communication links in the network with the objective of minimizing network interference. Using linear programming and semidefinite programming formulations of our optimization problem, we obtain tight lower bounds 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.

Bibliography

- [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, September 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] Chaska Wireless Solutions. <http://www.chaska.net>.
- [8] A. Coja-Oghlan, C. Moore, and V. Sanwalani. MAX k-CUT and Approximating the Chromatic Number of Random Graphs. In *ICALP*, 2003.
- [9] 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.
- [10] Engim. www.engim.com.
- [11] A. Frieze and M. Jerrum. Improved approximation algorithms for MAX k-CUT and MAX BISECTION. *Algorithmica*, 18, 1997.
- [12] GLPK: GNU Linear Programming Kit. <http://www.gnu.org/software/glpk/glpk.html>.
- [13] 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.
- [14] M. Grotschel, L. Lovasz, and A. Schrijver. Geometric Algorithms and Combinatorial Optimization. *Springer-Verlag*, 1987.
- [15] P. Gupta and P. R. Kumar. The Capacity of Wireless Networks. *IEEE Transactions on Information Theory*, 46(2), 2000.

- [16] A. Hertz and D. de Werra. Using tabu search techniques for graph coloring. *Computing*, 39(4), 1987.
- [17] Cirond Technologies Inc. Channel Overlap Calculations for 802.11b Networks. http://www.cirond.com/White_Papers/FourPoint.pdf, 2002. White Paper.
- [18] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of Interference on Multi-hop Wireless Network Performance. In *MOBICOM*, 2003.
- [19] P. Kyasanur and N.H.Vaidya. Capacity of Multi-Channel Wireless Networks: Impact of Number of Channels and Interfaces. In *MOBICOM*, 2005.
- [20] M.Alichery, R.Bhatia, and L.Li. Joint Channel Assignment and Routing for throughput optimization in Multi-Radio wireless mesh networks. In *MOBICOM*, 2005.
- [21] 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.
- [22] M.K.Marina and S.Das. A Topology Control Approach to Channel Assignment in Multi-Radio Wireless Mesh Networks. In *Broadnets*, 2005.
- [23] M.Kodialam and T.Nandagopal. Characterizing the Capacity Region in Multi-Radio, Multi-Channel Wireless Mesh Networks. In *MOBICOM*, 2005.
- [24] R. Montemanni, D.H. Smith, and S.M. Allen. Lower bounds for fixed spectrum frequency assignment. *Annals of Operations Research*, 107, October 2001.
- [25] 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.
- [26] P.Kyasanur and N.H. Vaidya. Routing and Interface Assignment in Multichannel Multi-Interface Wireless Networks. In *WCNC*, 2005.
- [27] K. Ramachandran, E. Belding, K. Almeroth, and M. Buddhikot. Interference-aware channel assignment in multi-radio wireless mesh networks. In *(Infocom)*, 2006.
- [28] A. Raniwala, K. Gopalan, and T. Chiueh. Centralized Channel Assignment and Routing Algorithms for Multi-Channel Wireless Mesh Networks. *ACM SIGMOBILE MC2R*, 8(2):50–65, 2004.
- [29] R. Raniwala and T. Chiueh. Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network. In *INFOCOM*, 2005.
- [30] J. Robinson, K. Papagiannaki, C. Diot, X. Guo, and L. Krishnamurthy. Experimenting with a multi-radio mesh networking testbed. In *(WinMee Workshop)*, 2005.
- [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] The Network Simulator ns-2. <http://www.isi.edu/nsnam/ns/>.
- [33] 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.