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

Anand Prabhu Subramanian, Himanshu Gupta, and Samir R. Das; SUNY, Stony Brook, NY.

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 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.

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. 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]. One common technique used to improve overall network capacity is use of multiple channels [19]. Essentially, wireless interference can be minimized 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; the overall objective of such an assignment strategy is to minimize the overall network interference.

Dynamic Channel Assignment. 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, 35, 36, 42] 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.

Static or Quasi-static Channel Assignment. Due to the difficulty of use of above dynamic approach with commodity hardware, there is need to develop techniques that assign channels statically [1, 26, 31, 32, 38]. 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 2) overheads are inconsequential. We refer to the above as *quasi-static channel assignments*. If there is only one radio interface per router, 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, 34], 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 [21, 34], or increased channel separation (as is the case in 802.11a) [32].

Problem Addressed. In our article, 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 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 [26, 31] 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 mathematical programming approaches, and develop heuristics that perform very close to the obtained bounds on the optimal.

Our Contributions. For the above described channel assignment problem, we develop a centralized and a distributed algorithm. The centralized algorithm is based on a popular heuristic search technique called Tabu search [16] that has been used in the past in graph coloring problems. The distributed approach is motivated by the greedy approximation algorithm for Max K-cut problem in graphs [10]. To evaluate their performances, we develop two mathematical programming 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.

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 multi-channel 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 [25], including channels that are supposedly orthogonal but interfere because of crosstalk or leakage [34].
- Ours is the first work that establishes good lower bounds on the optimal network interference, and demonstrates good performance of the developed heuristics by comparing them with the lower bounds.

Paper Organization. The rest of the paper is organized as follows. We start with describing the network model and the formulation of our problem in Section 2, and discuss related work in Section 3. We present our algorithms in Section 4 and Section 5 respectively. In Section 6, we obtain lower bounds on the optimal network interference using linear and semidefinite programming. Section 7 presents generalizations of our techniques. We present our simulation results in Section 8.

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 7.

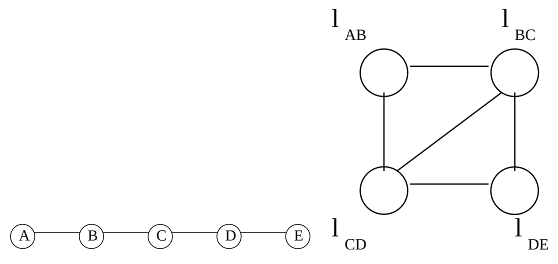
Interference Model. Due to the broadcast nature of the wireless links, transmission along a communication link (between a pair of wireless nodes) may interfere with transmissions along other communication links in the network. Two interfering links cannot engage in successful transmission at the same 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, 27]. 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 7. Moreover, in our approach of quasi-static channel assignment, 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 7.

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.



(a) Communication graph (b) Conflict graph

Figure 1. **Communication graph and corresponding conflict graph.**

We illustrate the concept of conflict graph in Figure 1. The wireless network represented in Figure 1 has five network nodes $A, B, \ldots, E$ 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 paper.

- N, the set of nodes in the network.
- R_i, the number of radio interfaces on node $i \in N$.
- $\mathcal{K} = \{1, 2, \ldots, K\}$, 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 paper, 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 paper.

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 a unique channel to each communication link[1] 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 assign channels to all links, and 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 [15].

More formally, consider a wireless mesh network over a set N of network nodes. The *channel assignment problem* is to compute a function $f : V_c \to \mathcal{K}$ to minimize the *overall network in-*

[1]Note 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.

terference $I(f)$ defined below while satisfying the below *interface constraint*.

Interface Constraint.

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

Network Interference $I(f)$.

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

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. The channel assignment problem is NP-hard since it reduces to Max K-cut (as discussed below).

Input Parameters – 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 7) can be computed using methods similar to recently reported measurement-based techniques in [29, 33]. 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 7) on each link. Such traffic measurements can be easily done using existing software tools (e.g., COMO [39]).

Relationship with Max K-cut. Given a graph G, the Max K-cut problem [10] 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.

3 Related Work

Following our discussion in Section 1, we classify the related work in two major classes.

Fast Switching of Channels. In MMAC protocol [36], 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 [42], 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. In AMCP [35], the authors uses similar notion of a control channel, but a single radio and focus on starvation mitigation. In [13] the authors use a channel assignment approach using a routing protocol and then use these channels to transmit data. For coordination, control channels are used. In [22] two radio and single radio multichannel protocols are proposed, but separate control channels are not needed.

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). But, the commodity 802.11 wireless cards incur a a channel switching delay of the order of milliseconds (based on our observations), as channel switching requires a firmware reset and execution of an associated procedure. Similar experiences were reported in [6]. In addition, the above approaches require changes to the MAC layer. Thus, the above approaches are not suitable with currently available commodity hardware.

Static/Quasi-Static Channel Assignment in Multiradio Networks. There have been many works that circumvent fast channel switching by assigning channels at a much larger time scale in a multiradio setting. In particular, [32] assume a tree-based communication pattern to ease coordination for optimizing channel assignment. Similar tree-based communication patterns have been used in [43]. The above schemes do not quantify the performance of their solutions with respect to the optimal. In addition, [38] 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.

To facilitate independent routing protocols, our work focusses on developing quasi-static channel assignment strategies that preserve the original network topology. 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 do not discuss how the channels are assigned to interfaces. In [31], 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 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 [30], 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 [26] 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.

Other Related Works. In other related works, [20] 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 [23, 27] 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, [19] derives upper bounds on capacity of wireless multihop networks with multiple channels.

4 Centralized 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. Our Tabu-based algorithm is centralized. 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 [26, 31, 38], and have also become prevalent in the industry (e.g., WLAN and mesh products from Meru Networks [24], Tropos [41], Strix Systems [37], Firetide [9]).

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 $f_0, f_1, f_2, \ldots, f_j, \ldots$, 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. 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 $I(f_i) > 0$ 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.

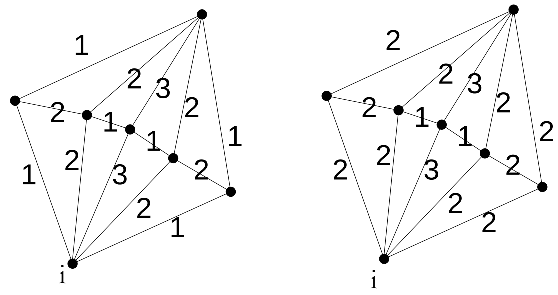


Figure 2. Merge operation of second phase. The two figures are the communication graphs of the network before and after the merge operation. Labels on the links denote the color/channel. Here, the merge operation is started at node i by changing all its 1-colored links to color 2.

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.

5 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 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 [7], 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]).

Centralized Version. The above observations motivate use of a greedy approach for our channel assignment problem. We start with presenting the 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 total time complexity of the greedy algorithm is $O(dK|V_c|^3)$.

Distributed Greedy Algorithm (DGA). 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 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 Distributed Greedy algorithm is localized, and can be made to work in dynamic topologies. Our simulation results showed that the above distributed algorithm performs almost same as the centralized version, due to the localized nature of the network interference objective function. The input network parameters of traffic and interference are measured as discussed in Section 2.

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.

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 $Y_{uk} \in \{0,1\}$, for each $u \in V_c$ and $k \in \mathcal{K}$; Y_{uk} is 1 iff $u \in V_c$ is assigned the channel k.
- Variables $X_{uv} \in \{0,1\}$, for each edge $(u,v) \in E_c$; X_{uv} is 1 iff $u,v \in V_c$ are assigned the same channel.[3]
- Variables $Z_{ik} \in \{0,1\}$, for each node $i \in N$ and channel $k \in \mathcal{K}$; Z_{ik} is 1 iff some $u \in E(i)$ is assigned a channel k.

$$\sum_{k \in \mathcal{K}} Y_{uk} = 1, \qquad \forall\, u \in V_c \tag{2}$$

$$X_{uv} \geq Y_{uk} + Y_{vk} - 1,\ \forall\,(u,v) \in E_c, \forall k \in \mathcal{K} \tag{3}$$

$$Z_{ik} \geq Y_{uk},\ \forall\, u \in E(i),\ \forall\, i \in N,\ \forall\, k \in \mathcal{K} \tag{4}$$

$$Z_{ik} \leq \sum_{u \in E(i)} Y_{uk}, \qquad \forall\, i \in N,\ \forall\, k \in \mathcal{K} \tag{5}$$

$$\sum_{f=1}^{k} Z_{if} \leq R_i \qquad \forall\, i \in N \tag{6}$$

Objective Function. Our objective function for the above ILP is to Minimize $\sum_{(u,v)\in E_c} X_{uv}$.

Linear Programming. Solving the above ILP (due to its NP-hardness) is intractable for reasonably sized problem instances. Thus, we relax the above ILP to a linear program (LP), i.e., we

[2]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.

[3]If u and v are assigned different channels, then X_{uv} is not constrained by Equation 3. However, it will be *chosen* to be 0 to minimize the objective function.

allow X_{uv}, Y_{uk}, Z_{ik} to be any real value between 0 and 1. The solution to the LP gives only a lower bound on the ILP's optimal solution. We observed that the lower bound obtained by the above LP formulation was very loose; thus, 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 [28] that the number of monochromatic edges in the complete subgraph of size $|S_u|$ when colored by K colors is at least:

$$\sigma(S_u, K) = \frac{\beta\alpha(\alpha+1) + (K-\beta)\alpha(\alpha-1)}{2}, \tag{7}$$

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

$$\sum_{v,w \in S_u} X_{vw} \geq \sigma(S_u, K) \quad \forall u \in V_c \tag{8}$$

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.

$$\sum_{(u,v) \in E(i)} X_{uv} \geq \sigma(E(i), R_i) \quad \forall i \in N \tag{9}$$

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 (12 hours vs. 1 hour on a 2.4 GHz Intel Xeon machine with 2GB RAM for a 50 node network) and memory than the LP formulation, and hence, is not feasible for very large network sizes. Note that the SDP and LP formulations are used only to demonstrate the performance of our Tabu-based and Greedy algorithms.

Semidefinite Programs. A *semidefinite program* [12] is a technique to optimize a linear function of a symmetric positive-semidefinite matrix[4] subject to linear equality constraints. Semidefinite programs can be solved in polynomial time using various techniques [14]. The reader is referred to [3, 12] for further details on semidefinite programming and its application to combinatorial optimization. The standard form of semidefinite program is as follows.

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

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 [10]. 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 [10]. We take an equilateral simplex Σ_K in $\mathbf{R}^{K-1}$ with vertices $b_1, b_2, \ldots, b_K$. Let $c_K = \frac{(b_1+b_2+\ldots+b_K)}{K}$ 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 [10].

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

$$\begin{array}{ll} \text{Maximize} & \frac{K-1}{K} \sum_{(u,v) \in E_c} (1 - y_u.y_v) \\ \text{such that} & y_u \in \{a_1, a_2, \ldots., a_K\} \end{array}$$

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

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

where $\sigma(E(i), R_i)$ is as defined as in Equation 7. Now, we add the following constraint to represent the interface constraint.

$$\sum_{u,v \in E(i)} y_u.y_v \geq \Phi_i \quad \forall i \in N \tag{10}$$

The above equation follows from the same observations as Equation 9. 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 ($E(i)$) [28], 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 10.

Note that even though Equation 10 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 10 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 relaxed SDP for the channel assignment is as follows.

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

The solution to the above 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.

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

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.) [34]. 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 [25]. 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. Measurements of these parameters was discussed in Section 2. 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 $f : V_c \rightarrow \mathcal{K}$.

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

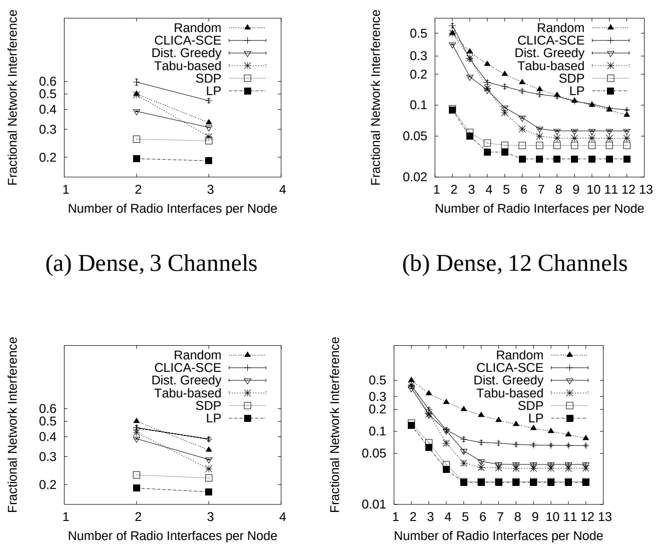
As before, Tabu-based and Greedy algorithms can use the above definition of network interference without any additional changes. However, in the LP formulation, we need to replace the Equations 3 by the following.

$$X_{uv} \geq Y_{uk_1} + Y_{vk_2} - 2 + c(k_1,k_2),\ \forall(u,v) \in E_c, \forall k_1,k_2 \in \mathcal{K}$$

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.

8 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.



(a) Dense, 3 Channels

(b) Dense, 12 Channels

(c) Sparse, 3 Channels

(d) Sparse, 12 Channels

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

We start with discussing various algorithms used for comparison.

Algorithms. In addition to our designed algorithms (Tabu-based and Distributed 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 centralized CLICA heuristic presented in [26] for a slightly different version of the channel assignment problem.[5] We refer to the modified algorithm of [26] 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. See Section 3 for a discussion on other related works.

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 that the centralized version of the greedy algorithm performed almost exactly the same as the Distributed Greedy algorithm.

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 [11] 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 average node degree is

[5]In CLICA [26], 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.

[6]We 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.

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 that than the Distributed 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 [26].

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 [40], 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 7.

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 802.11b where few orthogonal channels are available. Since our techniques are general enough to handle non-orthogonal channels (Section 7), 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 7. 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

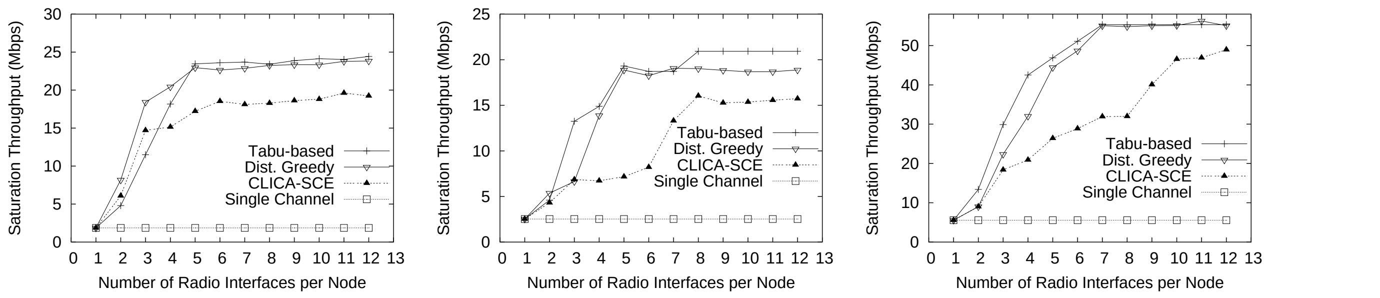


Figure 4. Saturation throughput in ns2 simulations for 12 channels and various traffic models, viz., (a) Single hop, (b) Multi-hop Peer-to-Peer, (c) Multi-hop Gateway.

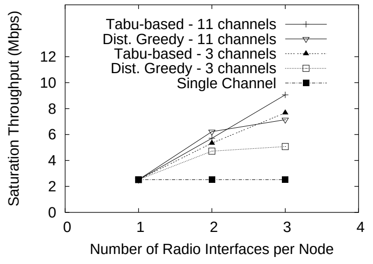


Figure 5. Saturation throughput in ns2 simulations when using non-orthogonal channels with 802.11b-like multi-channel model (11 channels with varying degrees of interference; 3 channels are mutually orthogonal).

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 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.

9 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 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. One of the future directions is to consider assignment of multiple channels to each link.

10 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] Firetide Mesh Networks. http://www.firetide.com/.

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

[11] GLPK: GNU Linear Programming Kit. http://www.gnu.org/software/glpk/glpk.html.

[12] 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.

[13] M.X. Gong, S.F. Midkiff, and S. Mao. A Combined Proactive Routing and Multi-Channel MAC Protocol for Wireless Ad Hoc Networks. In *Broadnets*, 2005.

[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] 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), 2006.

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

[22] R. Maheshwari, H. Gupta, and S. R. Das. Mutichannel MAC Protocols for Wireless Networks. In *SECON*, 2006.

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

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

[25] 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.

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

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

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

[29] 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.

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

[31] 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.

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

[33] 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.

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

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

[36] 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.

[37] Strix Systems. http://www.strixsystems.com/.

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

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

[40] The Network Simulator ns-2. http://www.isi.edu/nsnam/ns/.

[41] Tropos Networks. http://www.tropos.com.

[42] 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.

[43] Q. Xue and A. Ganz. Temporal Topology Control in Multi-channel Multihop Wireless Access Networks. In *Broadnets*, 2005.