

SDP-based Approach for Channel Assignment in Multi-radio Wireless Networks

Hieu Dinh
Computer Science &
Engineering Dept.
U. of Connecticut
Storrs, CT, 06269.

hdinh@engr.uconn.edu

Yoo-Ah Kim
Computer Science &
Engineering Dept.
U. of Connecticut
Storrs, CT, 06269.

ykim@engr.uconn.edu

Seungjoon Lee
AT&T Labs – Research
Florham Park, NJ 07932.
slee@research.att.com

Minho Shin
Computer Science Dept.
U. of Maryland
College Park, MD, 20740.
mhshin@cs.umd.edu

Bing Wang
Computer Science &
Engineering Dept.
U. of Connecticut
Storrs, CT, 06269.
bing@engr.uconn.edu

ABSTRACT

We consider the following channel assignment problem in multi-radio multi-channel wireless networks: Given a wireless network where k orthogonal channels are available and each node has multiple wireless interfaces, assign a channel to each link so that the total number of conflicts is minimized. We present an integer semidefinite programming formulation for the problem and show that it is equivalent to an optimal channel assignment. By relaxing integrality constraints, we can find a lowerbound on the optimal channel assignment. We develop several channel assignment algorithms based on the solution to the SDP relaxation. Our results from numerical evaluations and packet-level simulations show that our SDP-based rounding algorithms outperform other simple heuristics (up to 200% improvement in throughput). In particular, our schemes achieve even larger performance improvement when nodes have different numbers of interfaces, in which simple heuristics (e.g., greedy) do not perform well.

1. INTRODUCTION

In wireless networks, it is well known that interferences can affect network capacity significantly. For example, consider the network shown in Figure 1. If all three wireless links use the same channel, only one link can be used at a time due to the broadcast property of wireless medium. One popular way to overcome this limitation is to utilize orthogonal channels [1–8]. In Figure 1, if there are three channels available in the network and each node is equipped with two wireless interface cards, all links can be used at the same time by assigning a distinct channel to each link.

We consider multi-radio multi-channel wireless mesh networks,

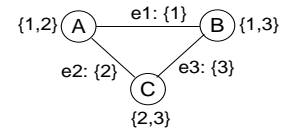


Figure 1: In this network, three channels are available, and each node has two wireless interface cards. The above channel assignment uses a distinct channel for each link, resulting in no conflict among links.

where multiple orthogonal channels are available and there are nodes equipped with multiple interfaces. Thus, those nodes can simultaneously communicate with several neighbors using different channels. An important problem here is which channels they should use for communications to maximize the network capacity. We assume that once assigned a channel, a link uses the channel for an extended period of time as in [4–7]. In this paper, we consider the following CHANNEL ASSIGNMENT problem: Given a wireless network where k orthogonal channels are available and each node v has C_v wireless interfaces, assign a channel to each link so that the total number of conflicts is minimized. We say a pair of links are *conflicting* if they are within interference range of each other and use the same channel. The problem is NP-hard even when C_v 's are the same for all v , by a reduction from the EDGE COLORING problem.

In this paper, we present an *integer semidefinite programming (ISDP) formulation*, which we prove is equivalent to an optimal channel assignment—our formulation provides a necessary and sufficient condition for an optimal channel assignment. By relaxing integrality constraints, we can obtain a lowerbound on the optimal solution in polynomial time. Subramanian *et al.* [7] recently developed a different SDP formulation. However, their formulation provides only a necessary condition to the problem even before the relaxation. They used the lowerbounds obtained from the formulation to evaluate their (non-SDP based) channel assignment algorithms. We compare these two SDP formulations and show that combining the two formulations gives a tighter lowerbound on the optimal.

We develop several *channel assignment algorithms* based on the

SDP solution. Our *numerical evaluations and packet-level simulations* show that our SDP-based rounding algorithms consistently outperform other simple heuristics (up to 200% improvement in throughput). In particular, the performance improvement is more dramatic when network environments make it more difficult to utilize channel diversity while maintaining network connectivity, for example, when nodes have different C_v 's, or C_v 's are much less than the number of channels available in the network.

The remainder of this paper is organized as follows. In Section 2 we describe the network model and formally define the problem. We review related work in Section 3. In Section 4, we present our ISDP formulation and several channel assignment algorithms. We discuss numerical and simulation results in Sections 5.

2. PROBLEM DEFINITION

We represent a network as an undirected graph $G = (V, E)$, where V is a set of nodes, and E a set of links. For each link e , $I(e)$ denotes a set of links that are within interference range of e . We also define $I(e, e')$ to be 1 if two links, e and e' , are within interference range of each other, and 0 otherwise. Let C_G be the number of orthogonal channels available in the system (*global channel constraint*) and C_v be the number of wireless interface cards at node $v \in V$ (*local interface constraints*). Without loss of generality, we assume $\forall v, C_v \leq C_G$. We call a network *homogeneous* if $C_v = l, \forall v \in V$ for some constant l . We call a network *heterogeneous* otherwise. We assume that each network interface can operate only on one channel at a given time.

Our objective is to assign channels to links to minimize the total number of conflicts. More formally, let $E(v)$ be the edges incident to node v and $c(e)$ be the channel assigned to edge e . Then, due to the local interface constraints, a channel assignment should satisfy $|\bigcup_{e \in E(v)} c(e)| \leq C_v$, and due to the global channel constraint, $|\bigcup_e c(e)| \leq C_G$. Let us define the *conflict number*, $CF_e(A)$, of an edge $e \in E$ in a channel assignment A to be the number of other edges in $I(e)$ that use the same channel as e in assignment A . We want to find a channel assignment that minimizes the total number of conflicts, $CF_G(A)$, which is defined as:

$$CF_G(A) = \frac{1}{2} \sum_{e \in E} CF_e(A). \quad (1)$$

Note that our channel assignment problem is a variant of the edge coloring problem. The objective of the traditional edge coloring problem is to minimize the number of colors used such that no two adjacent edges have the same color. In our problem, however, the maximum number of total colors (or channels) is a problem input, and the goal is to minimize the number of conflicts. In addition, we consider more general interference models. For example, in two-hop interference model, two edges may be conflicting with each other if they are incident to a common edge. In the remainder of this paper, we use colors and channels interchangeably.

Generalizations: We can generalize our model to take account of asymmetric networks and non-uniform interferences. When links between two nodes are asymmetric, we can assume that the network is a directed graph and define $I(e)$ for each directed link accordingly. In the case where the traffic in each link is non-uniform, we can define a weight function $w(e_1, e_2)$ for each pair of links to indicate the interference level, and incorporate it in the objective function. All the algorithms presented in this paper can be easily extended to these general cases.

3. RELATED WORK

Recently, a number of papers studied various issues in multi-radio multi-channel wireless networks. Raniwala *et al.* [2, 3] and Alicherry *et al.* [4] study the joint problem of channel assignment and routing in the context of mostly static mesh networks and present channel assignment schemes assuming the knowledge of long-term traffic load. Distributed channel assignment schemes are discussed in [5, 9], and a few recent papers analyze the capacity of multi-radio networks [6, 8].

Closest to our work is the recent one by Subramanian *et al.* [7], in which they consider the same channel assignment problem and independently come up with a different SDP formulation. Their formulation only provides a necessary condition to the problem while our Integer SDP formulation is equivalent to an optimal channel assignment. We also show that combining the two formulations in SDP relaxation leads to a tighter lowerbound to the optimal channel assignment. They developed two heuristic algorithms—one using greedy strategy and the other using Tabu search—and use the SDP solution to evaluate the heuristic algorithms as it provides a lowerbound on the optimal. In our work, we design several rounding algorithms, in which the SDP solution is used to obtain valid channel assignments.

A special case of our problem where $C_G = C_v = k$ for all v is the same as MIN k -PARTITION for the conflict graph¹ of network G [10]. It has been shown that for $k > 2$ and for every $\epsilon > 0$, there exists a constant α such that the MIN k -PARTITION cannot be approximated within $\alpha|V|^{2-\epsilon}$ unless $P = NP$. In fact, our SDP formulation without interface constraints is similar to the one used for MAX k -CUT [11], which is the dual problem of MIN k -PARTITION. In the general channel assignment problem, we also have *local interface constraints*, for which we develop a formulation that is equivalent to an optimal channel assignment.

We have developed approximation algorithms for several special cases [12]. In one-hop interference model², we develop an algorithm yielding at most $|V|$ more conflicts than the optimal solution when $C_v = k$ for all v , and show that this approximation result is best possible unless $P = NP$. When $C_v = 1$ or k , we present an algorithm with $(2 - \Theta(\frac{\ln k}{k}))OPT + (1 - \Theta(\frac{\ln k}{k}))|E|$ conflicts where OPT is the optimal number of conflicts.

4. CHANNEL ASSIGNMENT ALGORITHMS

In this section, we discuss several channel assignment algorithms. We first describe two greedy heuristics to illustrate the shortcomings of simple approaches, and then show how our SDP-based algorithms overcome the limitations.

4.1 Simple Heuristics

Heuristic 1: In this heuristic, each node v first chooses the channel set S_v and then picks the best channel for each link. Specifically, v first chooses $S_v = \{1, \dots, C_v\}$. Then a link $e = (u, v)$ can use any channel from $S_u \cap S_v = \{1, \dots, \min(C_u, C_v)\}$, which ensures all links have a nonempty channel set to choose a channel from. Given S_v , we consider each link in an arbitrary order and assign a channel that creates the minimum number of conflicts. More formally, let $n(e, i)$ be the number of edges using color i in $B(e)$,

¹In a conflict graph of a network G , there is a vertex for each link in G , and an edge between two vertices if the corresponding links can interfere with each other in G .

²In the one hop interference model, two links can interfere with each other if they are incident to a common node.

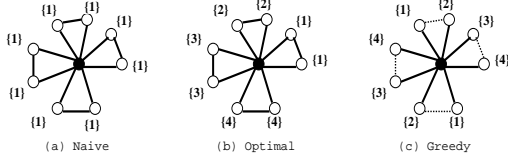


Figure 2: Channel assignments when black node has $k = 4$ interface cards and white nodes have only one interface card.

where $B(e)$ includes all edges in $I(e)$ that have chosen their colors before e . Then e chooses a channel i with the minimum value of $n(e, i)$ from $S_u \cap S_v$. We call this heuristic NAIVE ALGORITHM. (See Algorithm 1 for the pseudocode.)

Algorithm 1 NAIVE ALGORITHM

```

for each edge  $e = (u, v)$  do
   $S = \{1, \dots, \min(C_u, C_v)\}$ .
  assign color  $c \in S$  with min value of  $n(e, c)$  to  $e$ .
end for

```

A major shortcoming of this approach is that *the total number of different channels used may be much smaller than C_G* as each node chooses S_v conservatively to ensure the connectivity. For example, see the network shown in Figure 2. Suppose that the black node has k interface cards and all other nodes have only one interface. NAIVE assigns the same color for all edges whereas in the optimal solution, we can distribute edges evenly for each of those k colors.

Heuristic 2: A reasonable approach to utilize more channel diversity is to allow each edge to choose any channel from $\{1, \dots, C_G\}$ without restricting the channel set at the beginning. Once a node uses C_v different channels, then it has to restrict the channel set and cannot add more channels to obey the local interface constraint. A link can choose its channel that minimizes the number of conflicts to be created, as in NAIVE, among the allowed channels. We call this heuristic GREEDY ALGORITHM. A more detailed description is as follows (see the pseudocode in Algorithm 2): For any node v , let S_v be the colors that any edge in $E(v)$ is already using. S_v is said to be *tight* if $|S_v| = C_v$. When neither S_u nor S_v is tight, a new color can be added to both sets and therefore, $e = (u, v)$ can use any of C_G colors. Otherwise if either S_u or S_v is tight, then e chooses any color from the tight set. When both sets are tight, e has to choose a color from $S = S_u \cap S_v$. We choose color c with the minimum $n(e, c)$ among all allowed colors. Note that an edge e may be *dropped* (i.e., node u and v cannot communicate with each other directly) in case that $S = \emptyset$.

While the greedy approach may reduce the number of conflicts by utilizing more channels, the biggest drawback of this algorithm is that *it may drop a significant number of edges*. For example in Figure 2 (c), the black node may greedily choose channels for links so that the conflicts on the node can be minimized. As a result, 33% of links cannot be used in the solution. If an edge is dropped, then the pair of nodes will have to use another route to communicate with each other, thus increasing the traffic on links along that route. Therefore, dropping too many edges is undesirable and more importantly, may lead to network partition after channel assignment. As we can see in the experiments (Section 5), GREEDY drops 10 – 25% of edges in heterogeneous cases and sometimes

the network is disconnected.

Algorithm 2 GREEDY ALGORITHM

```

for each edge  $e = (u, v)$  do
  if  $|S_v| = C_v$  and  $|S_u| = C_u$  then
     $S = S_u \cap S_v$ .
  else if  $|S_v| = C_v$  and  $|S_u| < C_u$  then
     $S = S_v$ .
  else if  $|S_v| < C_v$  and  $|S_u| = C_u$  then
     $S = S_u$ .
  else
     $S = \{1, \dots, C_G\}$ .
  end if
  if  $S = \emptyset$  then
    drop edge  $e$ .
  else
    let  $c \in S$  be the color with min  $n(e, c)$ .
    assign color  $c$  to edge  $e$ .
    add  $c$  to  $S_v$  and  $S_u$  if not included.
  end if
end for

```

In the rest of this section, we present our SDP-based channel assignment algorithms. The algorithms are essentially greedy but as channel assignments are guided by the SDP solutions, we can reduce the number of conflicts compared to NAIVE while dropping fewer edges than GREEDY. We next formulate the problem as an integer semidefinite programming (Section 4.2), present rounding algorithms to obtain a valid channel assignment (Section 4.3), and describe how to further improve the assignment via local search (Section 4.4).

4.2 Semidefinite Programming Formulation

Global Channel Constraints: We first present ISDP formulation with only global channel constraints (i.e., any node can use all C_G channels) and later extend the formulation considering local interface constraints. Without interface constraints, the problem can be formulated in a way similar to the one used in MAX- k -CUT and VERTEX COLORING [11, 13]. Let $C_G = k$. Consider a $(k - 1)$ -dimensional vector X_e for each edge e .

ISDP:

$$\min \sum_{I(e_1, e_2)=1} \frac{1}{k} ((k - 1)X_{e_1} \cdot X_{e_2} + 1) \quad (2)$$

$$|X_e| = 1 \quad \forall e \in E \quad (3)$$

$$X_{e_1} \cdot X_{e_2} \in \left\{1, \frac{-1}{k-1}\right\} \quad \text{for } I(e_1, e_2) = 1 \quad (4)$$

We can relate a solution of ISDP to a channel assignment as follows. Consider k unit length vectors in $(k - 1)$ -dimensional space such that for any pair of vectors the dot product is $-\frac{1}{k-1}$. (Only k such vectors can be chosen and these k vectors make an equilateral k -simplex in $(k - 1)$ -dimensional space [11, 13].) For example, see Figure 3(a) where $C_G = 3$. We choose three vectors such that the dot product of any pair is $-\frac{1}{2}$ and map each channel to a distinct vector. A feasible channel assignment will correspond to assigning one of these three vectors to X_e . The objective function is exactly the same as the number of conflicts in the given channel assignment since if $X_{e_1} = X_{e_2}$ (e_1 and e_2 use the same channel),

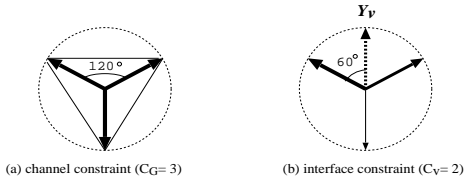


Figure 3: Vectors satisfying Constraints (4) and (5)

it contributes one to the objective function, and 0 otherwise.

Local Interface Constraints: We now consider the interface constraints that each node v has C_v interfaces. We introduce a unit vector Y_v for each node v and let $T_v = \sqrt{\frac{1}{C_v} \cdot \frac{k-C_v}{k-1}}$. We add the following constraints.

Interface Constraints 1 (IC_1)

$$\begin{aligned} Y_v \cdot X_e &\geq T_v \text{ for } \forall v, e \text{ where } C_v < k \text{ and } e \in E(v) \quad (5) \\ |Y_v| &= 1 \quad \forall v \in V \quad (6) \end{aligned}$$

Intuitively, vector Y_v will be positioned in the center of the hypercone defined by X_e 's, $e \in E(v)$. For example, see Figure 3 (b) where $k = 3$ and $C_v = 2$. Edges in $E(v)$ can use only two out of three vectors since the dot product of Y_v and X_e should be no less than $\frac{1}{2}$. In the following, we show that ISDP (2) - (6) is equivalent to an optimal channel assignment.

LEMMA 1. *ISDP (2) - (6) provides a necessary condition to an optimal channel assignment.*

PROOF. Given an optimal channel assignment, we can map each channel to one of k vectors for which the dot product of any pair is exactly $\frac{-1}{k-1}$ and easily check Constraints (2) - (4) are satisfied. Therefore, we only have to check Constraints (5). Since a node v uses at most C_v channels, let $S_v = \{X_1, X_2, \dots, X_t\}$ be the vectors corresponding to the channels used by edges in $E(v)$ where $t \leq C_v$. Then we can define Y_v to be $\frac{\sum_{i=1}^t X_i}{|\sum_{i=1}^t X_i|}$ (that is, Y_v is the unit vector which has the same distance to all vectors in S_v). Then

$$\begin{aligned} Y_v \cdot X_i &= X_i \cdot \frac{\sum_{i=1}^t X_i}{|\sum_{i=1}^t X_i|} = \frac{1 - \frac{t-1}{k-1}}{|\sum_{i=1}^t X_i|} \\ &= \frac{1}{|\sum_{i=1}^t X_i|} \cdot \frac{k-t}{k-1} = \sqrt{\frac{1}{t} \cdot \frac{k-t}{k-1}}. \end{aligned}$$

The last equality is because $|\sum_{i=1}^t X_i| = \sqrt{\sum_i \sum_j X_i \cdot X_j} = \sqrt{t - t(t-1)/(k-1)}$. Since $\sqrt{\frac{1}{t} \cdot \frac{k-t}{k-1}}$ is decreasing as t increases, Constraints (5) are satisfied. \square

LEMMA 2. *ISDP (2) - (6) provides a sufficient condition to an optimal channel assignment.*

PROOF. As (2)-(4) gives an optimal solution when at most k channels can be used in the network [11], we now show that Constraints (5) ensure that edges in $E(v)$ can have only C_v colors. Suppose that different vectors used by edges in $E(v)$ is X_1, X_2, \dots, X_t . We need to show that $t \leq C_v$. Since $X_i \cdot X_j = \frac{-1}{k-1}$, we have

$$\left| \sum_{i=1}^t X_i \right| = \sqrt{t \left(1 - \frac{t-1}{k-1} \right)}. \quad (7)$$

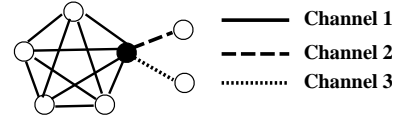


Figure 4: Black node has 2 interface cards and white nodes have only one interface card. If we only use IC_2 for interface constraints then the black node can use 3 channels even though $C_v = 2$.

We also have $X_i \cdot Y_v \geq T_v$, which means $\sum_{i=1}^t X_i \cdot Y_v \geq tT_v$ and therefore,

$$\left| \sum_{i=1}^t X_i \right| \geq Y_v \cdot \sum_{i=1}^t X_i = \sum_{i=1}^t X_i \cdot Y_v \geq tT_v. \quad (8)$$

The first inequality comes from the fact that $|x| \geq y \cdot x$ for any vector x if y is a unit vector. Comparing (7) with (8), we can obtain $t \leq C_v$. \square

By Lemma 1 and 2, we have the following theorem.

THEOREM 1. *ISDP (2)-(6) is equivalent to an optimal channel assignment.*

SDP relaxation: Solving ISDP is NP-hard and therefore, we relax Constraints (4) to

$$X_{e_1} \cdot X_{e_2} \geq \frac{-1}{k-1} \quad \text{for } I(e_1, e_2) = 1 \quad (9)$$

Then we can solve the SDP relaxation (2),(3),(5),(6),(9) in polynomial time (within any desired precision) [14-18], and the corresponding m -dimensional vectors X_e, Y_v can be obtained by Cholesky decomposition for some $m \leq |V| + |E|$ [19]. The solution gives a lowerbound on the optimal channel assignment.

Interface Constraints in [7]: Subramanian *et al.* [7] suggested a different formulation to ensure that each node uses at most C_v channels. Let d_v be the degree of vertex v , and $\alpha_v = \lfloor \frac{d_v}{k} \rfloor$ and $\beta_v = d_v \bmod k$ where $C_G = k$.

Then the number of conflicting pairs at node v when node v can use only C_v interfaces is at least:

$$\gamma(d_v, C_v) = \frac{\beta_v \alpha_v (\alpha_v + 1) + (C_v - \beta_v) \alpha_v (\alpha_v - 1)}{2}$$

and the dot product of the remaining pairs is at least $\frac{-1}{k-1}$. Let $\Phi(v)$

be $\gamma(d_v, C_v) + \left(\binom{d_v}{2} - \gamma(d_v, C_v) \right) \cdot \frac{-1}{k-1}$. Then the following constraints can be obtained.

Interface Constraints 2 (IC_2)

$$\sum_{e_1, e_2 \in E(v)} X_{e_1} \cdot X_{e_2} \geq \Phi(v) \quad \text{for each vertex } v \quad (10)$$

IC_2 provides only a necessary condition for an optimal channel assignment and does not give a valid channel assignment even with integrality constraints whereas our interface constraints IC_1 provides an optimal feasible channel assignment. Consider the instance given in Figure 4. In the figure, $k = 3$ and $C_v = 2$ for the black node and all other nodes have $C_v = 1$. Note that the clique component has to use the same channel for all edges. If

we use IC_2 , the center node can use all three channels which violate interface constraints, and no conflicts are created except in the clique. Note that in the optimal solution, only two channels (one for the clique and another for the remaining edges) can be used. Assuming the two-hop interference model, the optimal solution to the instance is 46 whereas using IC_2 gives 45, and using IC_1 gives 46.

After relaxing integrality constraints, however, we show in the following that having both IC_1 and IC_2 provides a tighter lower-bound on the optimal solution.

Comparison between IC_1 and IC_2 : We compare IC_1 and IC_2 and show that having both constraints gives a tighter bound on an optimal solution.

PROPERTY 1. IC_1 implies IC_2 when $\beta_v = 0$ for all v .

Proof of Property (1): Let $\{X_1, X_2, \dots, X_{d_v}\}$ be the set of vectors used by edges in $E(v)$ in a solution to SDP relaxation with IC_1 . Let $X = \sum_{i=1}^{d_v} X_i$. Then due to IC_1 we have $Y_v \cdot X \geq d_v T_v$ and therefore,

$$|X|^2 \geq |Y_v|^2 |X|^2 \geq |Y_v \cdot X|^2 \geq \frac{d_v^2}{C_v} \frac{k - C_v}{k - 1}.$$

On the other hand,

$$|X|^2 = \sum_i |X_i|^2 + 2 \sum_{i \neq j} X_i \cdot X_j = d_v + 2 \sum_{i \neq j} X_i \cdot X_j.$$

Therefore,

$$\begin{aligned} \sum_{i \neq j} X_i \cdot X_j &\geq \frac{1}{2} \left(\frac{d_v^2}{C_v} \frac{k - C_v}{k - 1} - d_v \right) \\ &= \frac{d_v}{2} \left(\alpha_v \frac{k - C_v}{k - 1} - 1 \right) \end{aligned}$$

which is exactly $\Phi(v)$ when $\beta_v = 0$.

PROPERTY 2. Including both IC_1 and IC_2 in the SDP relaxation provides a tighter lowerbound on the optimal solution. In general, IC_2 is likely to have a tighter bound in homogeneous cases whereas IC_1 performs better in heterogeneous cases.

We have already seen an example in Figure 4 in which IC_1 gives a tighter bound than IC_2 . On the other hand, the following gives a concrete example where IC_2 is tighter than IC_1 . Consider a star graph where a center node v has three edges to nodes u_1, u_2 , and u_3 . Let $C_G = C_v = 2$. In an optimal solution, there should be one conflict as at least two out of three edges should use the same channel, which can be achieved with IC_2 . If we use IC_1 then the solution to the relaxed SDP is $\frac{3}{4}$ by assigning vectors so that $X_i \cdot X_j = -\frac{1}{2}$ for every pair.

In Figure 5, we compare the solution with IC_2 to the one with IC_1 in various networks (See Section 5.1 for the details of network graphs). In our experiments, IC_2 gives tighter bounds in homogeneous cases and IC_1 gives tighter bounds in heterogeneous cases (random and proportional distributions). We include both constraints to obtain tighter lowerbounds, and the lowerbounds are used to evaluate the performance of various channel assignment algorithms in Section 5.2.

4.3 Rounding Algorithms

We now present our rounding algorithms. Given an optimal solution to the SDP relaxation, we round the solution to obtain a feasible channel assignment by satisfying both channel and interface

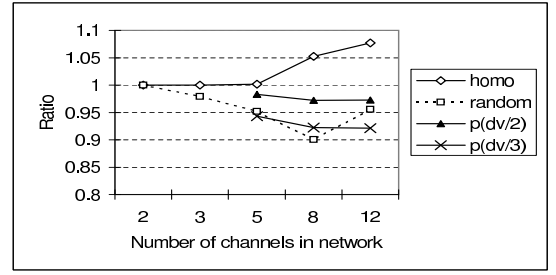


Figure 5: The ratio of the solution with IC_2 to IC_1 .

Algorithm 3 SDP-COLORSET ALGORITHM

```

solve SDP and obtain  $\{X_e, Y_v\}$ .
randomly select  $k$  vectors (Let  $R = \{r_1, r_2, \dots, r_k\}$  be the
chosen vectors).
for each vertex  $v$  do
    include  $C_v$  vectors with maximum  $r_i \cdot Y_v$  in  $S_v$ .
end for
for each edge  $e$  do
     $S = S_u \cap S_v$ .
    if  $S = \emptyset$  then
        drop edge  $e$ .
    else
        choose the vector with maximum value of  $r_i \cdot X_e$  from  $S$ .
        assign color  $r_i$  to edge  $e$ .
    end if
end for

```

constraints.

To round the solution, we first select C_G random vectors, denoted as $R = \{r_1, r_2, \dots, r_{C_G}\}$, each of which represents one of C_G channels. Each random vector $r_i = (r_{i,1}, r_{i,2}, \dots, r_{i,m})$ is selected by choosing each component $r_{i,j}$ independently at random from a standard normal distribution $N(0, 1)$, as in [11, 13]. We consider three rounding algorithms. The first two algorithms (SDP-COLORSET and SDP-GREEDY) do not guarantee the connectivity of the network although they preserve connectivity in almost all our experiments in Section 5. The third algorithm (SDP-SKELETON) finds a spanning tree (called *skeleton*) and maintains all the edges in the skeleton to guarantee the network connectivity.

SDP-COLORSET (Algorithm 3): In this algorithm, we first choose C_v vectors (based on Y_v) as color set $S_v \subseteq R$ for each node v and then choose a vector for an edge $e = (u, v)$ from $S_u \cap S_v$. In more detail, for each node v , S_v is chosen to be C_v vectors closest to Y_v . That is, $|S_v| = C_v$ and $r_i \cdot Y_v \geq r_j \cdot Y_v$ for any $r_i \in S_v$, $r_j \notin S_v$. Let $S = S_u \cap S_v$. We assign a vector $r_i \in S$ to an edge $e = (u, v)$ so that $r_i \cdot X_e \geq r_j \cdot X_e$ for any $r_j \in S$ (i.e., r_i is the closest vector to X_e in S). If S is empty, then e is dropped.

SDP-GREEDY (Algorithm 4): We greedily choose vectors for each edge from R instead of choosing color sets for nodes first. The algorithm is similar to GREEDY ALGORITHM but we choose a vector closest to X_e instead of choosing the one with the minimum conflicts. That is, we find the closest vector r_i to X_e among the vectors that can be used for e . As in GREEDY ALGORITHM, S is defined to be R if $|S_u| < C_u$ and $|S_v| < C_v$, and otherwise if either set is tight, we choose the tight set as S . If both are tight, $S = S_u \cap S_v$. The edge is dropped if $S = \emptyset$.

We now describe how we can guarantee network connectivity us-

Algorithm 4 SDP-GREEDY ALGORITHM

solve SDP and obtain $\{X_e, Y_v\}$.
randomly select k vectors (Let $R = \{r_1, r_2, \dots, r_k\}$ be the chosen vectors).
for each edge $e = (v, u)$ **do**
 if $|S_v| = C_v$ and $|S_u| = C_u$ **then**
 $S = S_u \cap S_v$.
 else if $|S_v| = C_v$ and $|S_u| < C_u$ **then**
 $S = S_v$.
 else if $|S_v| < C_v$ and $|S_u| = C_u$ **then**
 $S = S_u$.
 else
 $S = R$.
 end if
 if $S = \emptyset$ **then**
 drop edge e .
 else
 choose the vector with maximum value of $r_i \cdot X_e$ from S .
 assign color r_i to edge e .
 add r_i to S_v and S_u if not included.
 end if
end for

ing a tree property. Suppose a given network is a rooted tree, and we assign channels to links as in SDP-GREEDY, but in the breadth first search (BFS) order starting from the root. Then, no tree edge is dropped because links are considered in a top-down fashion, and when we assign a channel to a link $e = (u, v)$, one of the endpoints has an empty color set (thus at least one color set is not tight). SDP-SKELETON utilizes this property to obtain a connectivity-guaranteed channel assignment.

SDP-SKELETON (Algorithm 5): We first find a spanning tree T in the network. We choose an arbitrary node as the root of T and assign channels to links in T in BFS order. As in SDP-GREEDY, the closest vector satisfying the interface constraints will be chosen for each link. After finishing all links in T , we assign channels for the remaining links as in SDP-GREEDY.

4.4 Improvements via Local Search

Given a channel assignment, we further improve it by checking if we can make local changes to reduce the number of conflicts. For each edge e we check if there is any color c' other than the current color $c(e)$ so that setting $c(e)$ to c' does not violate interface constraints and reduce the number of conflicts. We repeat this until no such changes can be made.

We adopt this improvement to all the channel assignments obtained by the algorithms in the previous sections, and improve the channel assignments. Subramanian *et al.* [7] used the improvement algorithm, in which they start with assigning channel 1 to all nodes. We compare their algorithm (S-GREEDY) with our schemes in the following section.

5. PERFORMANCE EVALUATION

We now compare our SDP-based algorithms to alternative algorithms using numerical studies and packet-level simulations. More specifically, we compare the following algorithms: SDP-COLORSET, SDP-GREEDY, SDP-SKELETON, GREEDY, NAIVE and S-GREEDY. For SDP-based algorithms, we solve the semidefinite programming using DSDP 5.0 [20]. Then for each rounding algorithm we find a channel assignment by choosing the best result among 10 random

Algorithm 5 SDP-SKELETON ALGORITHM

solve SDP and obtain $\{X_e, Y_v\}$.
randomly select k vectors (Let $R = \{r_1, r_2, \dots, r_k\}$ be the chosen vectors).
compute a spanning tree T and its root
sort edges in BFS order
for each edge $e = (v, u) \in T$ (in sorted order) **do**
 suppose $|S_u| = 0$.
 if $|S_v| = C_v$ **then**
 $S = S_v$.
 else
 $S = R$.
 end if
 choose the vector with maximum value of $r_i \cdot X_e$ from S .
 assign color r_i to edge e .
 add r_i to S_v and S_u if not included.
end for
for each edge $e = (v, u) \notin T$ **do**
 if $|S_v| = C_v$ and $|S_u| = C_u$ **then**
 $S = S_u \cap S_v$.
 else if $|S_v| = C_v$ and $|S_u| < C_u$ **then**
 $S = S_v$.
 else if $|S_v| < C_v$ and $|S_u| = C_u$ **then**
 $S = S_u$.
 else
 $S = R$.
 end if
 if $S = \emptyset$ **then**
 drop edge e .
 else
 choose the vector with maximum value of $r_i \cdot X_e$ from S .
 assign color r_i to edge e .
 add r_i to S_v and S_u if not included.
 end if
end for

sets of vectors. That is, we select the result with the minimum number of conflicts among the ones with the least number of edge drops. (We consider the number of edge drops first as dropping edges may lead to a network partition.)

5.1 Settings

Network graphs: We place 50 nodes in a 1000m by 1000m square area uniformly at random, and create an edge between two nodes if they are within transmission range. We use 200m and 300m for the transmission range, which create sparse and dense graphs, respectively. In the rest of the paper, we use 5 sparse and dense graphs thus generated. Our algorithm can be used in any interference model but in experiments we use the two-hop interference model³ [21].

Number of Channels and Interface Cards: We vary C_G , from 2 to 8, and C_v is no more than 3 (considering realistic scenarios). Furthermore, we consider the following three different types of distributions for C_v :

- *Homogeneous distribution:* Every node v has the same number of interface cards. We represent this distribution as $h(l)$ where $C_v = l$ for all v .

³In the two-hop interference model, two edges can interfere with each other if they are within two-hop distance.

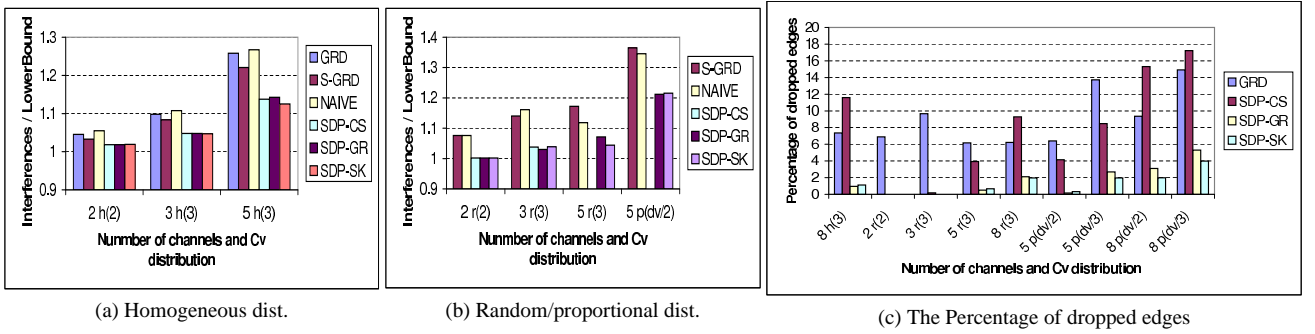


Figure 6: The ratio of the number of conflicts to the lowerbound. (α, β) in x-axis represents that $C_G = \alpha$ and the distribution of C_v follows β . Only cases with no edge drops are plotted.

- *Random distribution:* We assign C_v uniformly at random from $\{1, \dots, l\}$ for each node v . We represent this distribution as $r(l)$.
- *Proportional distribution:* C_v is proportional to the degree of v . That is, the network administrator may assign more wireless cards to nodes with high degrees (hub nodes) to handle the network traffic. We consider $C_v = \lceil d_v/2 \rceil$ and $\lceil d_v/3 \rceil$ where d_v is the degree of node v . We represent this distribution as $p(l)$ where l is either $d_v/2$ or $d_v/3$.

5.2 Numerical Results

For the various algorithms, we first compare the number of conflicts, and then investigate the number of dropped edges and the connectivity of a network. We present the results for sparse graphs; the results for dense graphs are similar.

Number of conflicts: When comparing the number of conflicts, we only consider the cases where no edge is dropped after channel assignment. This is because when different sets of edges are dropped in channel assignments, we cannot directly compare the number of conflicts as more edge drops typically decreases the number of conflicts. Note that when the number of conflicts is reduced by dropping edges, it does not necessarily lead to network performance improvement. We compare the performance for these cases using packet-level simulations in Section 5.3.

We first present the results for homogeneous distributions. Figure 6(a) compares the ratio of the number of conflicts to the lowerbound that we obtained from the SDP relaxation with both interface constraints. (The results are averaged over 5 graph instances). SDP-based algorithms outperform all other algorithms, with more significant performance improvements for larger number of channels. Moreover, SDP-based algorithms give solutions close to the optimal solution.

The results for random or proportional distributions are shown in Figure 6(b). The results for GREEDY are not included in the figure since it drops edges in all the cases plotted. For SDP-COLORSET, only the results when there is no edge drop are plotted (i.e., when $C_G < 5$). We again observe that SDP-based algorithms outperform other algorithms and obtain solutions close to the optimal solution.

Number of Dropped Edges and Network Connectivity: The channel assignments by SDP-based algorithms and GREEDY may lead to edge drops. Fig. 6 (c) shows the percentage of dropped edges by these algorithms. In general, SDP-GREEDY and SDP-SKELETON drops the least number of edges among these algorithms. As C_G increases, SDP-COLORSET drops more edges because in SDP-COLORSET, each node independently chooses its

color set without considering the information in the neighborhood (even though it is guided by the solution to SDP). Therefore, as C_G increases, the chances that the two endpoints of an edge fall into disjoint sets increase. (However, as we will see below SDP-COLORSET maintains network connectivity in most instances even when it drops more edges than GREEDY.)

Dropping edges may result in a disconnected network. We observe that GREEDY leads to several cases of network partition when $C_G \geq 3$ and the nodes have different number of interface cards. In particular, when C_v follows a random or proportional distribution, 32 out of 50 sparse instances (64%) were disconnected. For SDP-based algorithms, all solutions are connected except one instance in SDP-COLORSET.

5.3 Packet-level Simulation Study

We now evaluate the performance of the various channel assignment algorithms using simulation through the *ns-2* simulator. For this purpose, we have modified *ns-2* to support multiple radios in the network and multiple interfaces for each node. Our goal of this simulation-based study is twofold: First, we compare the performance of various algorithms in realistic environments, especially for the cases where some edges are dropped. Second, we investigate whether minimizing conflicts is a good indicator of performance observed by applications.

Simulation Settings: Traffic inside the network (graph) is generated as follows. We first separate the 50 nodes uniformly into two groups, each with 25 nodes, representing the source and destination group respectively. We then generate a random one-to-one mapping between the source and destination groups to obtain 25 source-destination pairs. We generate 10 such traffic instances and apply them to each graph. The path from a source to a destination is obtained using the shortest path routing (in terms of hop counts). The maximum data transmission rate is fixed at 1 Mbps for all the nodes.

The performance metric we use is *sustainable aggregate throughput* obtained as follows. All sources generate a constant-bit-rate flow at the same bitrate (a flow consists of 1024-byte UDP packets). The aggregate throughput is the sum of the bitrate of all the flows. We say that the aggregate throughput of the flows is *sustainable* when the average success delivery ratio of all the flows is at least 98%. To obtain the sustainable aggregate throughput, all flows start with a low bit rate (at 2 Kbps) and increase the bitrate with an increment of 2 Kbps until the aggregate bitrate cannot be sustained. If the initial bitrate is not sustainable, the sustainable aggregate throughput is set to 0.

Simulation Results: We now present the results for sparse graphs.

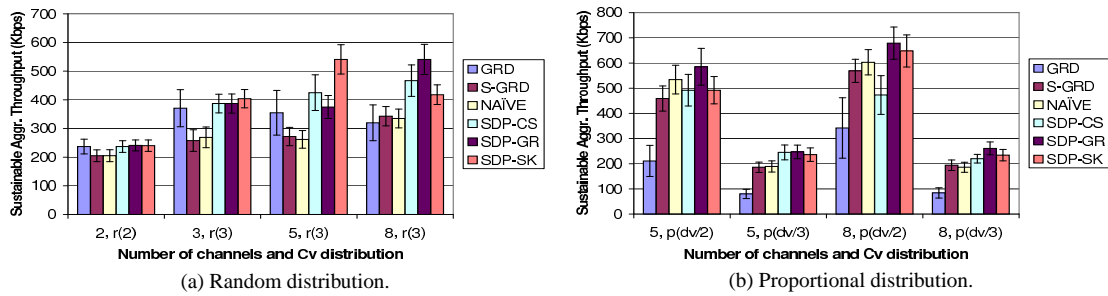


Figure 7: ns-2 simulation results for sparse graphs. (α, β) in x-axis represents that $C_G = \alpha$ and the distribution of C_v follows β .

(The results for dense graphs are similar.) In homogeneous networks, performances under the various algorithms are similar, while SDP-based algorithms slightly outperform other algorithms when there are more than 3 channels (figure omitted). However, in heterogeneous networks (random or proportional distributions) we observe a significant gain from using SDP-based algorithms, as shown in Figure 7. The 95% confidence intervals are obtained from 50 simulation runs (over five random graphs, each with 10 traffic instances). When C_v follows a random distribution (Figure 7 (a)), SDP-based algorithms consistently outperform other algorithms in all cases. When C_v follows a proportional distribution (Figure 7 (b)), SDP-GREEDY outperforms other algorithms; SDP-COLORSET may lead to disconnected networks (one disconnected instance is when $C_G = 8$ and $C_v = \lceil d_v/2 \rceil$) and does not perform as well as SDP-GREEDY. The channel assignment under GREEDY leads to many dropped edges and several disconnected networks when C_v follows a proportional distribution, and results in much lower throughput compared to other algorithms.

6. CONCLUSIONS

In this paper, we have considered the channel assignment problem in multi-radio wireless networks where nodes have multiple wireless interface cards. We have presented various algorithms to minimize the number of conflicts among nearby edges. We have formulated the problem as an integer semidefinite programming and developed several rounding algorithms based on the relaxed SDP solution. Our experimental results indicate that the SDP-based algorithms outperform other heuristics in various scenarios.

7. REFERENCES

- [1] A. Adya, P. Bahl, J. Padhye, A. Wolman, and L. Zhou, "A multi-radio unification protocol for IEEE 802.11 wireless networks," Microsoft Technical Report, MSR-TR-2003-41, June 2003.
- [2] A. Raniwala, K. Gopalan, and T. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *SIGMOBILE Mobile Computing and Communications Review*, vol. 8, no. 2, pp. 50–65, 2004.
- [3] A. Raniwala and T. Chiueh, "Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network," in *Proceedings of Infocom*, March 2005.
- [4] M. Alicherry, R. Bhatia, and L. Li, "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *MobiCom*, 2005.
- [5] M. Shin, S. Lee, and Y. Kim, "Distributed channel assignment for multi-radio wireless networks," in *IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2006.
- [6] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *ACM MobiCom*, New York, NY, USA, 2005.
- [7] Anand Prabhu Subramaniam, Himanshu Gupta, and Samir R. Das, "Minimum-interference channel assignment in multi-radio wireless mesh networks," in *SECON*, 2006.
- [8] P. Kyasanur and N. H. Vaidya, "Capacity of multi-channel wireless networks: impact of number of channels and interfaces," in *ACM MobiCom*, 2005.
- [9] B. Ko, V. Misra, J. Padhye, and D. Rubenstein, "Distributed channel assignment in multi-radio 802.11 mesh networks," in *WCNC*, 2007.
- [10] V. Kann, S. Khanna, J. Lagergren, and A. Panconesi, "On the hardness of max k-cut and its dual," in *Israel Symposium on Theory and Computing Systems (ISTCS)*, 1996.
- [11] A. Frieze and M. Jerrum, "Improved approximation algorithms for MAX k-CUT and MAX BISECTION," in *Integer Programming and Combinatorial Optimization*, Egon Balas and Jens Clausen, Eds., vol. 920, pp. 1–13. Springer, 1995.
- [12] C. Kari, Y. Kim, S. Lee, A. Russell, and M. Shin, "Soft edge coloring," in *APPROX*, 2007.
- [13] D. Karger, R. Motwani, and M. Sudan, "Approximate graph coloring by semidefinite programming," *Journal of the ACM*, vol. 45, pp. 264–265, 1998.
- [14] F. Alizadeh, "Interior point methods in semidefinite programming with applications to combinatorial optimization," *SIAM Journal on Optimization*, vol. 5, no. 1, pp. 13–51, 1995.
- [15] M. Grotschel, L. Lovasz, and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization," *Combinatorica*, vol. 1, pp. 169–197, 1981.
- [16] M. Grotschel, L. Lovasz, and A. Schrijver, "Geometric algorithms and combinatorial optimization," *Springer-Verlag*, 1987.
- [17] V. Nesterov and A. Nemirovskii, "Self-concordant functions and polynomial time methods in convex programming," *Central Economical and Mathematical Institute, U.S.S.R. Academy of Science, Moscow*, 1990.
- [18] V. Nesterov and A. Nemirovskii, "Interior-point polynomial algorithms in convex programming," *Society for Industrial and Applied Mathematics (SIAM)*, 1994.
- [19] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1983.
- [20] S. J. Benson and Y. Ye, "DSDP5: Software for semidefinite programming," 2005.
- [21] V. S. Anil Kumar, Madhav V. Marathe, S. Parthasarathy, and A. Srinivasan, "End-to-end packet-scheduling in wireless ad-hoc networks," in *Proc. of SODA*, 2004, pp. 1021–1030.