Soft Edge Coloring

Chadi Kari* Yoo-Ah Kim* Seungjoon Lee† Alexander Russell* Min-ho Shin‡

Abstract

We consider the following channel assignment problem arising in wireless networks. We are given a graph $G = (V, E)$, and the number of wireless cards C_v for all v, which limit the number of colors that edges incident to v can use. We also have the total number of channels C_G available in the network. For a pair of edges incident to a vertex, they are said to be *conflicting* if the colors assigned to them are the same. Our goal is to color edges (assign channels) so that the number of conflicts is minimized. We first consider the homogeneous network where $C_v = k$ and $C_G \geq C_v$ for all nodes v. The problem is NP-hard by a reduction from EDGE COLORING and we present two combinatorial algorithms for this case. The first algorithm is a distributed greedy method, which gives a solution with at most $(1 - \frac{1}{k})|E|$ more conflicts than the optimal solution. We also present an algorithm yielding at most $|V|$ more conflicts than the optimal solution. The algorithm generalizes Vizing's algorithm in the sense that it gives the same result as Vizing's algorithm when $k = \Delta + 1$. Moreover, we show that this approximation result is best possible unless $P = NP$. For the case where $C_v = 1$ or k, we show that the problem is NP-hard even when $C_v = 1$ or 2, and $C_G = 2$, and present two algorithms. The first algorithm is completely combinatorial and produces a solution with at most $(2 - \frac{1}{k})OPT + (1 - \frac{1}{k})|E|$ conflicts. We also develop an SDP-based algorithm, producing a solution with at most $1.122OPT + 0.122|E|$ conflicts for $k = 2$, and $(2 - \Theta(\frac{\ln k}{k}))OPT + (1 - \Theta(\frac{\ln k}{k}))|E|$ conflicts in general.

1 Introduction

We consider a channel assignment problem arising in multi-channel wireless networks. In wireless networks nearby nodes *interfere* with each other and cannot simultaneously transmit over the same wireless channel. One way to overcome this limitation is to assign independent channels (that can be used without interference) to nearby links of the network. Consider the example shown in Figure 1. When all links use the same channel, only one pair of nodes may communicate with each other at a time due to conflicts. However, if there are three channels available and each node has two wireless interface cards (so it can use two channels), then we may assign a different channel to each link to avoid conflicts among edges in this channel assignment. Channel assignment to utilize multiple channels have recently been studied by many researchers in networking community [1, 2, 3, 4].

We informally define the SOFT EDGE COLORING problem as follows. We are given a graph $G = (V, E)$, and constraints on the number of wireless cards C_v for all v, which limit the number of colors that edges incident to v can use. In addition, we have a constraint on the total number of channels available in the network (denoted as C_G). For a pair of edges incident to a vertex, they are said to be *conflicting* if the colors assigned to them are the same. Our goal is to color edges (assign channels) so that the number of conflicts is minimized while satisfying constraints on the number of colors that can be used.

SOFT EDGE COLORING is a variant of the EDGE COLORING problem. In our problem, coloring need not be proper (two adjacent edges are allowed to use the same color)—the goal is to minimize the number of such conflicts. In addition, each node has its local color constraint, which limits the number of colors that can be used by the edges incident to the node. For example, if a node has two wireless cards ($C_v = 2$), the node can choose two colors and edges incident to the node should use only those two colors.

*Department of Computer Science and Engineering, University of Connecticut, Storrs, CT 06269 . E-mail : chadi, ykim, acr@engr.uconn.edu.
†AT&T Labs - Research, Florham Park, NJ 07932. E-mail: slee@research.att.com.
‡Department of Computer Science, University of Maryland, College Park, MD 20742. E-mail: mhshin@cs.umd.edu.

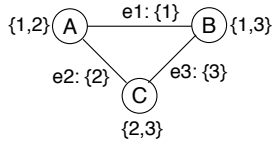


Figure 1: Each node has two wireless interface cards (thus can use two different channels) and three channels are available in total. We can assign a distinct channel to each link as shown above so that there is no conflict among edges.

Our results. We briefly summarize our results. We first consider the homogeneous network where $C_v = k$ and $C_G \geq C_v$ for all nodes v. For an arbitrary k, the problem is NP-hard by a reduction from EDGE COLORING. We present two combinatorial algorithms for this case. The first algorithm is a simple greedy method, which gives a solution with at most $(1 - \frac{1}{k})|E|$ more conflicts than the optimal solution; furthermore, it can be computed in a distributed fashion. We also present an algorithm yielding at most $|V|$ more conflicts than the optimal solution. The algorithm generalizes Vizing's algorithm in the sense that it gives the same result when $k = \Delta + 1$. In fact, our algorithm gives an optimal solution when $d_v \bmod k = k - 1$ for all vertices v. Moreover, we show that this approximation result is best possible unless $P = NP$.

In a heterogeneous network, we consider the case where each node v can have different $C_v = 1$ or k. We show that the problem is NP-hard even when $C_v = 1$ or 2, and $C_G = 2$, and present two algorithms for this case. The first algorithm is completely combinatorial and produces a solution with at most $(2 - \frac{1}{k})OPT + (1 - \frac{1}{k})|E|$ conflicts. We also develop an SDP-based algorithm, producing a solution with at most $1.122OPT + 0.122|E|$ conflicts for $k = 2$, and $(2 - \Theta(\frac{\ln k}{k}))OPT + (1 - \Theta(\frac{\ln k}{k}))|E|$ conflicts in general (slightly better than the combinatorial algorithm).

Relationship to MIN K-PARTITION and MAX K-CUT. The MIN K-PARTITION problem is the dual of MAX K-CUT problem where we color vertices with k different colors so that the total number of conflicts (monochromatic edges) is minimized. Our problem for the homogeneous network when $C_G = C_v = k$ for all v is an edge coloring version of MIN k-PARITION problem[1]. Kann *et al.* [5] showed 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 our problem when $C_v = k$ for all v, we have an approximation algorithm with additive term of $|V|$.

For the case when $C_v = 1$ or k, we use a SDP formulation similar to one used for MAX K-CUT and utilize the upperbounds obtained in [6]. To obtain a $(2 - \Theta(\frac{\ln k}{k}))$-approximation, we compare the upperbounds with two lowerbounds — one based on necessary interference at a vertex determined by its degree, and one given by the SDP relaxation.

Other Related Work. Fitzpatrick and Meertens [7] have considered a variant of graph coloring problem (called the SOFT GRAPH COLORING problem) where the objective is to develop a distributed algorithm for coloring vertices so that the number of conflicts is minimized. The algorithm repeatedly recolors vertices to quickly reduce the conflicts to an acceptable level. They have studied experimental performance for regular graphs but no theoretical analysis has been provided. Damaschke [8] presented a distributed soft coloring algorithm for special cases such as paths and grids, and provided the analysis on the number of conflicts as a function of time t. In particular, the conflict density on the path is given as $O(1/t)$ when two colors are used, where the conflict density is the number of conflicts divided by $|E|$.

In the traditional edge coloring problem, the goal is to find the minimum number of colors required to have a proper edge coloring. The problem is NP-hard even for cubic graphs [9]. For a simple graph, a solution using at most $\Delta + 1$ colors can be found by Vizing's theorem [10] where Δ is the maximum degree of a node. For multigraphs, there is an approximation algorithm which uses at most $1.1\chi' + 0.8$ colors where χ' is the optimal number of colors required [11] (the additive term was improved to 0.7 by Caprara *et al.* [12]).

[1] Or it can be considered as MIN k-PARTITION problem when the given graph is a line graph where the line graph of G has a vertex corresponding to each edge of G, and there is an edge between two vertices in the line graph if the corresponding edges are incident on a common vertex in G.

1.1 Problem definition We are given a graph $G = (V, E)$ where $v \in V$ is a node in a wireless network and an edge $e = (u, v) \in E$ represents a communication link between u and v. Each node v can use C_v different channels and the total number of channels that can be used in the network is C_G. More formally, let $E(v)$ be the edges incident to v and $c(e)$ be the color assigned to e. Then $|\bigcup_{e \in E(v)} c(e)| \leq C_v$ and $|\bigcup_{e \in E} c(e)| \leq C_G$.

A pair of edges e_1 and e_2 in $E(v)$ are said to be conflicting if the two edges use the same color. Let us define the *conflict number* (CF_e) of an edge $e \in E$ to be the number of other edges that conflict with e. In other words, for an edge $e = (u, v)$, CF_e is the number of edges (other than e itself) in $E(u) \bigcup E(v)$ that use the same channel as e. Our goal is to minimize the total number of conflicts. That is,

$$CF_G = \frac{1}{2} \sum_{e \in E} CF_e. \tag{1.1}$$

In the remainder of this paper, we mean *channels* by *colors* and use edge coloring and channel assignment, interchangeably. We also use conflicts and interferences interchangeably.

2 Algorithms for Homogeneous Networks

In this section, we consider the case for a homogeneous network where for all nodes v, the number of channels that can be used is the same ($C_v = k$). For an arbitrary k, the problem is NP-hard as the edge coloring problem can be reduced to our problem by setting $k = C_G = \Delta$ where Δ is the maximum degree of nodes.

2.1 Greedy Algorithm We first present and analyze a greedy algorithm for this problem. The algorithm works as follows: We choose colors from $\{1, \ldots, k\}$ (We only use k colors even when $C_G > k$ and the approximation ratio of our algorithm remains the same regardless of the value of C_G.) For any uncolored edge $e = (u, v)$, we choose a color for edge e that introduces the smallest number of conflicts. More formally, when we assign a color to $e = (u, v)$, we count the number of edges in $E(u) \bigcup E(v)$ that are already colored with c (denoted as $n(c, e)$), and choose color c with the smallest $n(c, e)$.

THEOREM 2.1. *The total number of conflicts by the greedy algorithm in homogeneous networks is at most*

$$CF_G = \frac{1}{2} \sum_{e \in E} CF_e \leq \mathit{OPT} + (1 - \frac{1}{k})|E|. \tag{2.2}$$

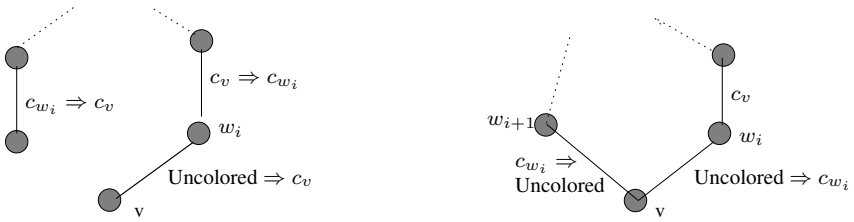
Theorem 2.1 directly follows from Lemma 2.1 and 2.2. The proofs are included in Appendix.

LEMMA 2.1. *The total number of conflicts when $C_v = k$ for all node v is at least $\frac{1}{2} \sum_v \frac{d_v^2}{k} - |E|$.*

LEMMA 2.2. *The total number of conflicts introduced by Algorithm 1 is at most $\frac{1}{2} \sum_v \frac{d_v^2}{k} - \frac{|E|}{k}$.*

Note that the algorithm can be performed in a distributed manner and each node needs only local information. We can also consider a simple randomized algorithm, in which each edge chooses its color uniformly at random from $\{1, \ldots, k\}$. The algorithm gives the same expected approximation guarantee and it can be easily derandomized using conditional expectations. The following corollary of Lemma 2.1 will be used to prove approximation factors for heterogenous networks.

COROLLARY 2.1. *Given an optimal solution, let $OPT(S)$ ($S \subseteq V$) be the number of conflicts at vertices in S and $|E(S)|$ be $\sum_{v \in S} \frac{d_v}{2}$. Then we have $OPT(S) \geq \frac{1}{2} \sum_{v \in S} \frac{d_v^2}{k} - |E(S)|$.*



(a) STEP 2 (b) STEP 3

Figure 2: The figures illustrate how recoloring is performed in BALANCEDCOLORING. The colors beside edges indicate the original color and the color after recoloring.

2.2 Improved Algorithm In this section, we give an algorithm with additive approximation factor of $|V|$. Our algorithm is a generalization of Vizing's algorithm in the sense that it gives the same result as Vizing's algorithm when $k = \Delta + 1$ where Δ is the maximum degree of nodes. We first define some notations. For each vertex v, let $m_k = \lfloor \frac{d_v}{k} \rfloor$ and $\alpha_v = d_v - m_v k$. Let $|E_i(v)|$ be the size of the color class of color i at vertex v i.e. the number of edges adjacent to v that have color i.

DEFINITION 2.1. *A color i is called* strong *on a vertex v if $|E_i(v)| = m_k + 1$. A color i is called weak on v if $|E_i(v)| = m_k$. A color i is called very weak on v if $|E_i(v)| < m_k$.*

DEFINITION 2.2. *A vertex v has a* balanced *coloring if the number of strong classes at v is at most $\min(\alpha_v + 1, k - 1)$ and no color class in $E(v)$ is larger than $m_k + 1$. A graph $G = (V, E)$ has a balanced coloring if each vertex $v \in V$ has a balanced coloring.*

In the following we present an algorithm that achieves a balanced coloring for a given graph $G = (V, E)$; we show in Theorem 2.3 that a balanced coloring implies an additive approximation factor of $|V|$ in terms of number of conflicts. In Algorithm BALANCEDCOLORING(e) described below, we color edge e so that the graph has a balanced coloring (which may require the recoloring of already colored edges to maintain the balanced coloring), assuming that it had a balanced coloring before coloring e. We perform BALANACEDCOLORING for all edges in arbitrary order. The following terms are used in the algorithm description. Let $|S_v|$ denote the number of strong color classes at vertex v.

DEFINITION 2.3. *For vertex $v \in V$ with $|S_v| < \min(\alpha_v + 1, k - 1)$ or with $|S_v| = k - 1$, i is a missing color if i is weak or very weak on v. For vertex $v \in V$ with $|S_v| = \alpha_v + 1$, i is a missing color if i is very weak on v*

DEFINITION 2.4. *An ab-*path *between vertices u and v where a and b are colors, is a path connecting u and v and has the following properties:*

- *Edges in the path have alternating colors a and b.*
- *Let $e_1 = (u, w_1)$ be the first edge on that path and suppose e_1 is colored a, then u must be missing b and not missing a.*
- *If v is reached by an edge colored b then v must be missing a but not missing b, otherwise if v is reached by an edge colored a then v must be missing b and not missing a.*

DEFINITION 2.5. *A flipping of an ab-path is a recoloring of the edges on the path such that edges previously with color a will be recolored with color b and vice versa.*

Algorithm BALANCEDCOLORING($e = (v, w)$)
Let $w_1 = w$. At i-th round ($i = 1, 2, \ldots$), we do the following.

STEP 1: Let $\mathcal{C}_v$ be the set of missing colors on v. If $i = 1$, $\mathcal{C}_{w_1}$ is the set of missing colors on w_1. When $i \geq 2$, $\mathcal{C}_{w_i}$ is the set of missing colors on w_i minus $c_{w_{i-1}}$. ($c_{w_{i-1}}$ is defined in STEP 2 at $(i-1)$-th round). If $\mathcal{C}_v \cap \mathcal{C}_{w_i} \neq \emptyset$, then choose color $a \in \mathcal{C}_v \cap \mathcal{C}_{w_i}$, color edge (v, w_i) with a and terminate.
STEP 2: If $\mathcal{C}_v \cap \mathcal{C}_{w_i} = \emptyset$, choose $c_v \in \mathcal{C}_v$ and $c_{w_i} \in \mathcal{C}_{w_i}$. Find a $c_v c_{w_i}$-path that starts at w_i and does not end at v. If such a path exists, flip this path, color edge (v, w_i) with c_v and terminate.
STEP 3: If all $c_v c_{w_i}$-paths that start at vertex w_i end at v, fix one path and let (v, w_{i+1}) be the last edge on that path. The edge (v, w_{i+1}) must have color c_{w_i}. Uncolor it and color edge (v, w_i) with c_{w_i}. Mark edge (v, w_i) as used and repeat the above steps with edge (v, w_{i+1}) (go to $(i+1)$-th round).

Analysis In the following, we prove that our algorithm terminates and achieves a balanced coloring. First we prove that we can always find a missing color at each round (Lemma 2.3 and 2.4) and at some round $j < d_v$, the algorithm terminates (Lemma 2.5). Due to the choice of missing colors and ab-path, we can show that our algorithm gives a balanced coloring (Lemma 2.6 and 2.7).

LEMMA 2.3. *For the given edge (v, w_1), there is a missing color at v and w_1. That is, $C_v \neq \emptyset$ and $\mathcal{C}_{w_1} \neq \emptyset$.*

Proof. When $|S_v| < \min(\alpha_v + 1, k - 1)$ or $|S_v| = k - 1$, there must be at least one weak color, which is a missing color. If $|S_v| = \alpha_v + 1$ then we can show that the remaining $k - \alpha_v - 1$ color classes cannot be all weak (i.e. having size m_v). Note that $d_v = m_v k + \alpha_v$, so if there are $\alpha_v + 1$ strong color classes and the remaining color classes have all exactly size m_v then the number of edges at v is strictly larger than d_v, which is not possible. So there must be a very weak class of which size is strictly less than m_v. □

For w_i, $i \geq 2$, we need to choose a missing color at w_i other than $c_{w_{i-1}}$. We prove in the following lemma, that there is a missing color other than $c_{w_{i-1}}$.

LEMMA 2.4. *At i-th round ($i \geq 2$), there is a missing color other than $c_{w_{i-1}}$ at w_i.*

Proof. Note first that $c_{w_{i-1}}$ was not a missing color at w_i in $(i-1)$-th round since otherwise we should have stopped at STEP 2. Consider the case that $c_{w_{i-1}}$ was a strong color in $(i-1)$-th round. As it is not possible that all k colors are strong, there must be a weak (or very weak) color c other than $c_{w_{i-1}}$ in $(i-1)$-th round. After uncoloring (u, w_i), the number of strong color classes will be reduced and we now have $|S_v| < \min(\alpha_v + 1, k - 1)$. Then c is missing at w_i in i-th round.

For the case that $c_{w_{i-1}}$ was a weak color in $(i-1)$-th round, $|S_v| = \alpha_v + 1$ in $(i-1)$-th round (otherwise, $c_{w_{i-1}}$ should have been a missing color). After uncoloring (u, w_i), S_v remains the same but $c_{w_{i-1}}$ is a very weak color. We need to show that there is a very weak color other than $c_{w_{i-1}}$. The number of edges that have weak or very weak colors is at most $(d_v - 1) - (\alpha_v + 1)(m_v + 1) = (k - \alpha_v - 1)m_v - 2 = (k - \alpha_v - 3)m_v + 2(m_v - 1)$. Therefore, there must be at least one very weak color other than $c_{w_{i-1}}$. □

LEMMA 2.5. *At some round $j < d_v$ there exists a $c_v c_{w_j}$-path starting at w_j and not ending at v.*

Proof. At some round j, if the algorithm does not terminate at Step 1 of some round $k < j$, the colors are going to run out (i.e. the missing color c_{w_j} is the same as color $c_{w_i}, i < j$ and all edges (v, w_i) with color c_{w_i} have been already used by the algorithm). We show that there is no $c_v c_{w_i}$-path connecting v and w_j (thus the algorithm has to terminate at Step 2). Suppose that there exists a $c_v c_{w_i}$-path P connecting v and w_j, v is missing color c_v and not missing color c_{w_i}, so v must be reached on an edge colored c_{w_i}. Let $e = (v, w_i)$ be the edge in P adjacent to v. Since we used in the algorithm all edges with color c_{w_i}, then edge (v, w_i) must have been already used by the algorithm. Now rewind the algorithm to the point where (v, w_i) was uncolored, w_i is missing c_{w_i} and not missing c_v, so if P exists there must be also a $c_v c_{w_i}$-path connecting w_i and w_j. This contradicts that at round $i < j$ we could not find such path. □

LEMMA 2.6. *A flipping of an ab-path in a graph with balanced coloring will not violate the balanced coloring.*

Proof. Suppose an ab-path runs from u to v. Suppose u is missing color b and not missing color a. Let $e = (u, w_0)$ be the first edge of that path, so it is colored by a. Flipping the ab-path will recolor e with color b, but since b was missing on u the color class $|E_b(v)|$ will not exceed $m_v + 1$, and also the number of strong classes will not become larger than $\min(\alpha_v + 1, k - 1)$ as we made color a missing on u. The same argument works for v but with possibly b and a interchanged in the argument. For internal vertices on the path nothing changes as the number of edges colored a and b stays the same. □

LEMMA 2.7. *Let v be a vertex that has a balanced coloring. Let $e \in E(v)$ be uncolored and let i be a missing color on v. Coloring e with i will not violate a balanced coloring at v.*

Proof. Suppose that at a vertex v, $|S_v| < \min(\alpha_v + 1, k - 1)$. Then the number of strong color classes at v is strictly less than $\alpha_v + 1$ and coloring edge e with i will not violate a balanced coloring at v as $|S_v|$ will not exceed $\min(\alpha_v + 1, k - 1)$ and $E_i(v)$ will not exceed $m_v + 1$ (i is missing on v). Suppose at a vertex v, $|S_v| = \alpha_v + 1$, then we show in Lemma 2.3 and 2.4 that there must be a very weak color class. When $|S_v| = k - 1$, the remaining color is very weak as one edge is not colored. Thus coloring e with i will not make $E_i(v)$ a strong color class and the number of strong color classes remains the same. So the balanced coloring at v will not be violated. □

THEOREM 2.2. *The above algorithm terminates and achieves a balanced coloring.*

Proof. In Lemma 2.5 we show that at some round j, $(1 \leq j \leq d_v)$, if we do not terminate at Step 1 of the algorithm then there will be a $c_v c_{w_j}$-path P starting at w_j and not ending at v. Now, if for some i, $(1 \leq i \leq j)$, $\mathcal{C}_v \cap \mathcal{C}_{w_i} \neq \emptyset$ then vertices v and w_i are missing the same color c_v. By Lemma 2.7, coloring edge (v, w_i) with color c_v will not violate a balanced coloring at v or at w_i, hence the algorithm terminates at Step 1 with a balanced coloring for G.
If $\forall i,\ i \leq j, \mathcal{C} \cap \mathcal{C}_{w_i} = \emptyset$ we show that the algorithm terminates at Step 2 of round j. As mentioned above, at round j there will be a $c_v c_{w_j}$-path P starting at w_j and not ending at v. On this path, the edge adjacent to w_j is colored with c_v since w_j is missing c_{w_j} and not missing c_v. Note that flipping path P will recolor this edge with color c_{w_j} making color c_v missing on w_j. Furthermore, by Lemma 2.6, flipping P will not violate the balanced coloring at any vertex in P. Thus c_v is now missing at v and w_j and as in Step 1 we can now color edge (v, w_j) with color c_v without violating a balanced coloring at v or w_j. So the algorithm terminates at Step 2 with a balanced coloring. □

THEOREM 2.3. *A balanced coloring of a graph achieves a $|V|$ additive approximation factor*

Proof. We have shown an algorithm that colors the edges of a graph $G = (V, E)$ such that the coloring is balanced at each vertex. Here we show that the algorithm introduces at each vertex $v \in V$ one more conflict than the optimal solution. At each vertex v suppose there is an ordering on the size of the color classes, 1 being a strong class and k being the weakest class. Note that at a vertex v, the number of conflicts is minimized when the number of strong classes is α_v and the remaining colors are weak. As the number of strong classes achieved by our algorithm is at most $\alpha_v + 1$, the first α_v classes introduce the same number of conflicts in both the optimal and our solution.
The $(\alpha_v + 1)^{th}$ color class in a balanced coloring which is strong, exceeds the corresponding color class in OPT (which is necessarily weak) by 1. Then the additional number of conflicts is $\frac{1}{2}(m_v + 1)m_v - \frac{1}{2}m_v(m_v - 1) = m_v$.

Now if there is an additional edge in the $(\alpha_v + 1)^{th}$ color class in a balanced coloring then there must be an additional edge in some color class i, $\alpha_v + 1 < i < k$ in OPT i.e. some color class i is weak in OPT but very weak in our balanced coloring. The number of additional conflicts of OPT in i is $\frac{1}{2}m_v(m_v - 1) - \frac{1}{2}(m_v - 1)(m_v - 2) = m_v - 1$. So, finally the additional number of conflicts introduced by the balanced algorithm is 1 at each vertex. Thus the approximation factor is $|V|$. □

COROLLARY 2.2. *When $\alpha_v = k - 1$ for all v, the algorithm gives an optimal solution.*

Proof. Note that the balanced coloring gives exactly $k-1$ strong color classes and one weak color class when $\alpha_v = k-1$, which is the optimal. □

We can show that the approximation ratio given by the algorithm is best possible unless $P = NP$. The proof is included in Appendix A.3.

THEOREM 2.4. *It is NP-hard to approximate the channel assignment problem in homogeneous networks within an additive term of $o(|V|^{1-\epsilon})$, given a constant ϵ.*

3 Networks where $C_v = 1$ or k

In this section, we present two algorithms for networks with $C_v = 1$ or k and analyze the approximation factors of the algorithms. The case where $C_v = 1$ or k is interesting since (i) it reflects a realistic setting, in which most of mobile stations are equipped with one wireless card and nodes with multiple wireless cards are placed in strategic places to increase the capacity of networks. (ii) as shown in Theorem 3.1, the problem is NP-hard even when $C_v = 1$ or 2. (The proof is in Appendix A.4.)

THEOREM 3.1. *The channel assignment problem to minimize the number of conflicts is NP-hard even when $C_v = 1$ or 2, and $C_G = 2$.*

3.1 Extended Greedy Algorithm We first present an extended greedy algorithm when $C_v = 1$ or k, and $C_G \geq C_v$. The approximation factor is $2-\frac{1}{k}$. Even though the algorithm based on SDP (semi-definite programming) gives a slightly better approximation factor (see Section 3.2), the greedy approach gives a simple combinatorial algorithm. The algorithm generalizes the idea of the greedy algorithm for homogeneous networks. In this case, an edge cannot choose its color locally since the color choice of an edge can affect colors for other edges to obey color constraints.

Before describing the algorithm, we define some notations. Let $V_i \subseteq V$ be the set of nodes v with $C_v = i$ (i.e., we have V_1 and V_k). V_1 consists of connected clusters $V_1^1, V_1^2, \ldots V_1^t$, such that nodes $u, v \in V_1$ belong to the same cluster if and only if there is a path composed of nodes in V_1 only. (See Figure 6 in Appendix for example.) Let E_1^i be a set of edges both of which endpoints are in V_1^i. We also define B_1^i to be a set of edges whose one endpoint is in V_1^i and the other is in V_k. We can think of B_1^i as a set of edges in the boundary of cluster V_1^i. Note that all edges in $E_1^i \bigcup B_1^i$ should have the same color. E_k is a set of edges both of which endpoints are in V_k. E_1 is defined to be $\bigcup_i E_1^i$.

In the greedy algorithm for homogeneous networks, each edge greedily chooses a color so that the number of interferences it creates (locally) is minimized. Similarly, when $C_v = 1$ or k, edges in the same cluster V_1^i choose a color so that the number of conflicts it creates is minimized. Formally, we choose a color c with minimum value of $\sum_{e=(u,v)\in B_1^i, v\in V_k} n_c(v)$ where $n_c(v)$ is the number of edges $e' \in E(v)$ with color c. Once edges in $E_1^i \bigcup B_1^i$ for all i choose their colors, the remaining edges (edges belonging to E_k) greedily choose their colors.

Any edges (u, v) incident to a vertex in V_1 should use the same color and therefore are conflicting with each other no matter what algorithm we use. Given an optimal solution, consider $OPT(V_1)$ and $OPT(V_k)$ where $OPT(S)$ is the number of conflicts at vertices in $S \subseteq V$. Similarly, we have $CF(V_1)$ and $CF(V_k)$ where $CF(S)$ is the number of conflicts at vertices in $S \subseteq V$ in our solution. Then we have $OPT(V_1) = CF(V_1)$. Therefore, we only need to compare $OPT(V_k)$ and $CF(V_k)$.

THEOREM 3.2. *The number of conflicts created by the extended greedy algorithm at V_k is at most $(2-1/k)OPT + (1-1/k)|E|$.*

Proof. We will simply show that the number of conflicts created by the extended greedy algorithm at V_k is at most $(2-1/k)OPT(V_k)+(1-1/k)|E(V_k)|$ where $|E(V_k)|$ is $\sum_{v\in V_k} \frac{d_v}{2}$ as $CF(V_1) = OPT(V_1)$. For each $e \in E \setminus E_1$,

$n(e)$ be the number of conflicts at vertices in V_k which are introduced when we assign a channel to e. Then the total number of conflicts at V_k is $\sum n(e)$.

We first consider the number of conflicts created when we assign colors to edges in B_1^i (recall that B_1^i is a set of edges of which endpoints are in V_1^i and V_k). For an edge $e = (u, v)$ where $u \in V_1^i$ and $v \in V_k$, let $d_v(e)$ be the number of edges in $E(v)$ to which a color is assigned before e. Then when we choose a color for $E_1^i \bigcup B_1^i$, the number of conflicts at vertices in V_k with edges not in $E_1^i \bigcup B_1^i$, is at most

$$\frac{\sum_{v \in V_k} \sum_{e \in E(v) \bigcap B_1^i} d_v(e)}{k}$$

as we choose a color with minimum conflicts. If v has $e_i(v)$ edges in B_1^i, $\frac{1}{2}e_i(v)(e_i(v) - 1)$ additional conflicts (between edges in B_1^i) are created. For edges in E_k we use the greedy algorithm presented in Section 2, and therefore, the number of conflicts created when we assign colors in E_k is at most $\frac{\sum_{v \in V_k} \sum_{e \in E(v) \bigcap E_k} d_v(e)}{k}$.

Summing up all the conflicts,

$$\sum_{e \in E \setminus E_1} n(e) \quad \leq \quad \frac{\sum_{v \in V_k} \sum_{e \in E(v)} d_v(e)}{k} + \frac{\sum_i \sum_{v \in V_k} (e_i^2(v) - e_i(v))}{2}$$

As for each node v, $\sum_{e \in E(v)} d_v(e) \leq \frac{d_v(d_v-1)}{2} - \frac{\sum_i (e_i^2(v) - e_i(v))}{2}$ (colors for edges in B_1^i will be determined at the same time), we have

$$\begin{aligned}
\sum_{e \in E \setminus E_1} n(e) &\leq \frac{1}{k} \sum_{v \in V_k} \left(\frac{d_v(d_v - 1)}{2} - \frac{\sum_i e_i^2(v) - e_i(v)}{2}\right) + \frac{\sum_i \sum_{v \in V_k} (e_i^2(v) - e_i(v))}{2} \\
&= \frac{1}{2} \sum_{v \in V_k} \frac{d_v(d_v - 1)}{k} + (1 - \frac{1}{k}) \frac{\sum_i \sum_{v \in V_k} (e_i^2(v) - e_i(v))}{2} \\
&= \frac{1}{2} \sum_{v \in V_k} \frac{d_v^2}{k} - \frac{1}{2} \sum_{v \in V_k} d_v + (1 - \frac{1}{k}) \frac{\sum_i \sum_{v \in V_k} (e_i^2(v) - e_i(v))}{2} + \frac{1}{2}(1 - \frac{1}{k}) \sum_{v \in V_k} d_v \\
&\leq (2 - \frac{1}{k}) OPT(V_k) + (1 - \frac{1}{k}) |E(V_k)|.
\end{aligned}$$

where $OPT(V_k)$ is the optimal number of conflicts at vertices in V_k and $|E(V_k)|$ be $\sum_{v \in V_k} \frac{d_v}{2}$.

The last inequality comes from the fact that both $\frac{1}{2} \sum_{v \in V_k} \frac{d_v^2}{k} - \frac{1}{2} \sum_{v \in V_k} d_v$ (by Corollary 2.1) and $\frac{1}{2} \sum_i \sum_{v \in V_k} (e_i^2(v) - e_i(v))$ are lower bounds on the optimal solution. □

Note that as in the homogeneous case, we can obtain the same expected approximation guarantee with a randomized algorithm, i.e., choose a color uniformly at random for each cluster V_1^i. Note also that the approximation ratio remains the same for any $C_G \geq k$. In the following section, we obtain a slightly better approximation factor using SDP relaxation when $C_v = 1$ or k and $C_G = k$.

3.2 SDP-based algorithm In this subsection, we assume that k different channels are available in the network and all nodes have 1 or k wireless cards. We formulate the problem using semidefinite programming. Consider the following vector programming (VP), which we can convert to an SDP and obtain an optimal solution in polynomial time. We have an m-dimensional unit vector Y_e for each edge e ($m \leq n$).

$$\textbf{VP:} \qquad \min \sum_v \sum_{e_1, e_2 \in E(v)} \frac{1}{k}((k-1) Y_{e_1} \cdot Y_{e_2} \quad + \quad 1) \tag{3.3}$$

$$|Y_e| \quad = \quad 1 \tag{3.4}$$

$$Y_{e_1} \cdot Y_{e_2} = 1 \quad \text{if } C_v = 1,\ e_1, e_2 \in E(v) \tag{3.5}$$

$$Y_{e_1} \cdot Y_{e_2} \geq \frac{-1}{k-1} \text{ for } e_1, e_2 \in E(v) \tag{3.6}$$

We can relate a solution of VP to a channel assignment as follows. Consider k unit length vectors in m-dimensional space such that for any pair of vectors v_i and v_j, the dot product of the vectors is $-\frac{1}{k-1}$. (It has been shown that $-\frac{1}{k-1}$ is the minimum possible value of the maximum of the dot products of k vectors [6, 13].) Given an optimal channel assignment of the problem, we can map each channel to a vector v_i. Y_e takes the vector that corresponds to the channel of edge e. If C_v is one, all edges incident to v should have the same color. The objective function is exactly the same as the number of conflicts in the given channel assignment since if $Y_{e_1} = Y_{e_2}$ (e_1 and e_2 have the same color), it contributes one to the objective function, and 0 otherwise. Thus the optimal solution of the VP gives a lower bound of the optimal solution in the channel assignment problem.

The above VP can be converted to a semidefinite programming (SDP) and solved in polynomial time (within any desired precision) [14, 15, 16, 17, 18], and given a solution for the SDP, we can find a solution to the corresponding VP, using incomplete Cholesky decomposition [19].

We use the rounding technique used for MAXCUT by Goeman and Williamson [20] when $k = 2$ and show that the expected number of interferences in the solution is at most $1.122OPT + 0.122|E|$.When $k > 2$, we obtain the approximation guarantee of $(2 - \frac{1}{k} - \frac{2(1+\epsilon)\ln k}{k} + O(\frac{k}{(k-1)^2})$ with additive term of $(1 - \frac{2(1+\epsilon)\ln k}{k-1}(1 - \frac{k}{(k-1)^2}))|E|$, where $\epsilon(k) \sim \frac{\ln\ln k}{(\ln k)^{\frac{1}{2}}}$.

When $k = 2$: We select a random unit vector r, and assign channel one to all edges with $Y_e \cdot r \geq 0$ and channel two to all other edges.

LEMMA 3.1. *[20] For $-1 \leq t \leq 1$, $\frac{\arccos t}{\pi} \geq \frac{\alpha}{2}(1-t)$, where $\alpha > .87856$.*

THEOREM 3.3. *The expected number of total conflicts by our algorithm is at most $1.122OPT + 0.122|E|$.*

Proof. See Appendix A.5. □

When $k > 2$: We use the rounding algorithm for MAX k-CUT when $k > 2$ [6]. Given an optimal solution for VP, we obtain a coloring as follows. We first select k random vectors, denoted as $R = \{r_1, r_2, \cdots, r_k\}$. Each random vector $r_i = (r_{i,1}, r_{i,2}, \ldots r_{i,n})$ is selected by choosing each component $r_{i,j}$ independently at random from a standard normal distribution $N(0, 1)$. For each edge e, assign e to vector r_i if r_i is the closest vector to Y_e (i.e., the vector with the maximum value of $Y_e \cdot r_i$). Ties are broken arbitrarily.

Let $\beta_{ij} = Y_{e_i} \cdot Y_{e_j}$. Let P include all pairs of edges in $E(v)$ for any $v \in V_k$. For a pair $(i, j) \in P$, (i, j) is included in pP (positive pairs)$\subseteq P$ if $\beta_{ij} \geq 0$ and (i, j) is included in nP (negative pairs) $\subseteq P$ if $\beta_{ij} < 0$. We utilized the following two lemmas from [6].

LEMMA 3.2. *[6] For $(i, j) \in nP$, $E[X_{ij}] = \frac{1}{k} + 2(1+\epsilon)\frac{\ln k}{k}\beta_{ij} + O(\beta_{ij}^2)$ where $\epsilon(k) \sim \frac{\ln\ln k}{(\ln k)^{\frac{1}{2}}}$ [21]*

LEMMA 3.3. *For $(i, j) \in pP$, $E[X_{ij}] \leq \frac{1}{k}((k-1)\beta ij + 1)$*

Proof. As $E[X_{ij}] = \frac{1}{k}((k-1)\beta_{ij} + 1)$ when $\beta_{ij} = 0$ and 1, and $E[X_{ij}]$ is a convex function in $[0, 1]$ [6], we have the lemma. □

Note the if we simply compare the lowerbound obtained by SDP and the upperbound given in Lemma 3.2 and 3.3, we cannot obtain a constant factor approximation. However, by carefully combining the lowerbound in Corollary 2.1, we can obtain a slightly better approximation factor than the greedy algorithm. We define $Val_1(S)$ to be $\frac{1}{k}|S|$ for any set $S \subseteq P$. In addition, let $Val_2(S)$ be $\frac{1}{k}\sum_{(i,j)\in S}((k-1)\beta_{ij}+1)$. That is, $Val_2(S)$ is a lowerbound obtained by SDP relaxation and $Val_1(S)$ is a lower bound based on the fact that all edges incident to a vertex can interfere with each other. As shown in Figure 3, simply combining two lowerbounds gives a 2-approximation. To prove a better bound than greedy, we first prove the following lemmas.

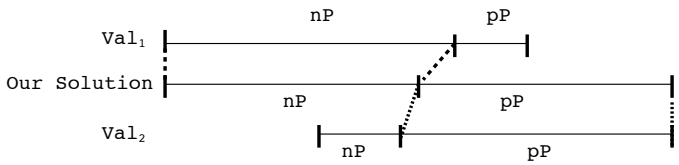


Figure 3: The expected number of conflicts in nP is bounded by $Val_1(nP)$ and conflicts in pP is bounded by $Val_2(pP)$.

LEMMA 3.4. *The number of conflicts by the algorithm is at most* $Val_2(P) + \delta(\frac{k-1}{2(1+\epsilon)\ln k} - 1) + \sum_{(i,j)\in nP} O(\beta_{ij}^2)$ *where* $\delta = -\frac{2(1+\epsilon)\ln k}{k}\sum_{(i,j)\in nP}\beta_{ij}$.

Proof.

$$\begin{aligned}
\sum_{(i,j)\in P} E[X_{ij}] &= \sum_{(i,j)\in pP} E[X_{ij}] + \sum_{(i,j)\in nP} E[X_{ij}] \\
&\leq \sum_{(i,j)\in pP} \frac{1}{k}((k-1)\beta ij + 1) + \sum_{(i,j)\in nP}(\frac{1}{k} + 2(1+\epsilon)\frac{\ln k}{k}\beta_{ij} + O(\beta_{ij}^2)) \\
&= Val_2(P) + \sum_{(i,j)\in nP}(\frac{1}{k} + 2(1+\epsilon)\frac{\ln k}{k}\beta_{ij}) - \sum_{(i,j)\in nP}\frac{1}{k}((k-1)\beta ij + 1) + \sum_{(i,j)\in nP} O(\beta_{ij}^2) \\
&\leq Val_2(P) + \delta(\frac{k-1}{2(1+\epsilon)\ln k} - 1) + \sum_{(i,j)\in nP} O(\beta_{ij}^2). \qquad (3.7)
\end{aligned}$$

□

LEMMA 3.5. *The number of conflicts by the algorithm is at most* $Val_1(P) - \delta + Val_2(P)(1-\frac{1}{k}) + \sum_{(i,j)\in nP} O(\beta_{ij}^2)$ *where* $\delta = -\frac{2(1+\epsilon)\ln k}{k}\sum_{(i,j)\in nP}\beta_{ij}$.

Proof. By Lemma 3.2 the number of conflicts of pair of edges in nP is at most $Val_1(nP) - \delta + \sum_{(i,j)\in nP} O(\beta_{ij}^2)$ and by Lemma 3.3 the number of conflicts of pair of edges in pP is at most $Val_1(pP) + \frac{k-1}{k}\sum_{(i,j)\in pP}\beta_{ij}$ so we have:

$$\begin{aligned}
\sum_{(i,j)\in P} E[X_{ij}] &\leq Val_1(nP) - \delta + Val_1(pP) + \frac{k-1}{k}\sum_{(i,j)\in pP}\beta ij + \sum_{(i,j)\in nP} O(\beta_{ij}^2) \\
&= Val_1(P) - \delta + \frac{k-1}{k}\sum_{(i,j)\in pP}\beta ij + \sum_{(i,j)\in nP} O(\beta_{ij}^2) \\
&\leq Val_1(P) - \delta + Val_2(P)(1-\frac{1}{k}) + \sum_{(i,j)\in nP} O(\beta_{ij}^2) \qquad (3.8)
\end{aligned}$$

□

THEOREM 3.4. *The number of conflicts created by the algorithm is at most* $((2 - \frac{1}{k} - \frac{2(1+\epsilon)\ln k}{k} + O(\frac{k}{(k-1)^2}))OPT + (1 - \frac{2(1+\epsilon)\ln k}{k-1}(1 - \frac{k}{(k-1)^2}))|E|$, *where* $\epsilon(k) \sim \frac{\ln\ln k}{(\ln k)^{\frac{1}{2}}}$.

Proof. By Lemma 3.4 and 3.5 the number of conflicts is upper bounded by $\min(Val_2(P) + \delta(\frac{k-1}{2(1+\epsilon)\ln k} - 1), Val_1(P) - \delta + Val_2(P)(1 - \frac{1}{k})) + \sum_{(i,j)\in nP} O(\beta_{ij}^2)$, which is maximized when $\delta = \frac{2(1+\epsilon)\ln k}{k-1}(Val_1(P) - \frac{1}{k}Val_2(P))$. Let $f(k) = \frac{2(1+\epsilon)\ln k}{k-1}$. Then the maximum number of conflicts is

$$(1 - f(k))Val_1(P) + (1 - \frac{1}{k} + \frac{f(k)}{k})Val_2(P) + \sum_{nP} O(\beta_{ij}^2).$$

Note that $Val_1(P) \leq OPT(V_k) + |E(V_k)|$ and $Val_2(P) \leq OPT(V_k)$. Therefore, the total conflict at V_k is at most

$$(2 - \frac{1}{k} - \frac{(k-1)f(k)}{k})OPT(V_k) + (1 - f(k)|E(V_k)| + \sum_{nP} O(\beta_{ij}^2).$$

Since $\beta_{ij}^2 \leq \frac{1}{(k-1)^2}$, we have the theorem. □

4 Discussion

Note that in all of our algorithms the total number of different colors used in the network is only $\max C_v$ rather than C_G. Although the number of conflicts may be reduced using more colors than $\max C_v$ (see for example Figure 1), it is not easy to make sure that each edge has at least one channel which are available at both endpoints in that case. In fact, it may be possible that the size of the set of common channels is small, which may result in creating more conflicts. One possible solution is to further improve the solution by recoloring edges with additional colors after obtaining the solution by the algorithm. It will be an interesting future work to analyze how much we can improve the performance by such recoloring.

Acknowledgements. The second author would like to thank Nikhil Bansal for useful discussions.

References

[1] A. Raniwala and T. Chiueh, "Architecture and algorithms for an IEEE 802.11-based multi-ch annel wireless mesh network," in *Proceedings of Infocom*, March 2005.

[2] P. Kyasanur and N. H. Vaidya, "Routing and interface assignment in multi-channel multi-interfa ce wireless networks," in *Proceedings of WCNC*, March 2005.

[3] Mansoor Alicherry, Randeep Bhatia, and Li (Erran) Li, "Joint channel assignment and routing for throughput optimization in mu lti-radio wireless mesh networks," in *In Proceedings of ACM MobiCom*, 2005.

[4] Min ho Shin, Seungjoon Lee, and Yoo-Ah Kim, "Distributed channel assignment for multi-radio wireless networks," in *The Third IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2006.

[5] V. Kann, S. Khanna, J. Lagergren, and A. Panconesi, "On the hardness of max k-cut and its dual," in *Proc. 5th Israel Symposium on Theory and Computing Systems (ISTCS)*, 1996, pp. 61–67.

[6] 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.

[7] Stephen Fitzpatrick and Lambert Meertens, "An experimental assessment of a stochastic, anytime, decentralized, soft colourer for sparse graphs," in *1st Symposium on Stochastic Algorithms, Foundations and Applications (SAGA)*, 2001.

[8] Peter Damaschke, "Distributed soft path coloring," in *STACS '03: Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science*, London, UK, 2003, pp. 523–534, Springer-Verlag.

[9] Ian Holyer, "The np-completeness of edge-coloring," *SIAM J. Computing*, vol. 10, no. 4, pp. 718–720, 1981.

[10] V. G. Vizing, "On an estimate of the chromatic class of a p-graph (russian)," *Diskret. Analiz.*, vol. 3, pp. 25–30, 1964.

[11] T. Nishizeki and K. Kashiwagi, "On the 1.1 edge-coloring of multigraphs," *SIAM J. Disc. Math.*, vol. 3, no. 3, pp. 391–410, August 1990.

[12] Alberto Caprara and Romeo Rizzi, "Improving a family of approximation algorithms to edge color multigraphs," *Information Processing Letters*, vol. 68, no. 1, pp. 11–15, 1998.

[13] D. Karger, R. Motwani, and M. Sudan, "Approximate graph coloring by semidefinite programming," in *Proc. 35th IEEE Symposium on Foundations of Computer Science*, 1994, pp. 2–13.
[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," *SIAM*, 1994.
[19] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1983.
[20] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *Journal of the ACM*, vol. 42, pp. 1115–1145, 1995.
[21] J. Galambos, "The asymptotic theory of extreme order statistics," Wiley, New York, 1978.

A Proofs

A.1 Proof of Lemma 2.1 For a node v, let $E(v)$ be the edges incident to v. $E(v)$ will be partitioned into k sets according to colors assigned to edges. Let $E_i(v)$ be a set of edges with color i. $E(v) = \bigcup_i E_i(v)$. Let $c(e)$ be the color assigned to e. Then

$$\begin{aligned}
\frac{1}{2}\sum_e CF_e &= \frac{1}{2}\sum_{e=(u,v)} (|E_{c(e)}(v)| + |E_{c(e)}(u)| - 2) \\
&= \frac{1}{2}\sum_v \sum_{e\in E(v)} (|E_{c(e)}(v)| - 1) \\
&= \frac{1}{2}\sum_v \sum_i |E_i(v)|(|E_i(v)| - 1) \\
&= \frac{1}{2}\sum_v \sum_i |E_i(v)|^2 - |E|
\end{aligned}$$

Note that for each node v, $\sum_i |E_i(v)|^2$ is minimized when the size of $E_i(v)$ is the same for all colors i. Therefore, we have

$$\frac{1}{2}\sum_e CF_e \geq \frac{1}{2}\sum_v (\frac{d_v}{k})^2 \cdot k - |E| = \frac{1}{2}\sum_v \frac{d_v^2}{k} - |E|.$$

A.2 Proof of Lemma 2.2 To upperbound the number of conflicts by the greedy algorithm, consider an edge $e = (u, v)$. Let $n(e)$ be the number of conflicts that e introduces when it gets colored. The total number of conflicts in the final coloring is $\sum_e n(e)$. Since we choose a color for e such that it introduces the smallest conflicts in node u and v, $n(e)$ is at most $\lfloor \frac{d_v(e)+d_u(e)}{k} \rfloor$ where $d_v(e)$ and $d_u(e)$ are the number of edges that get colored *before* e in $E(v)$ and $E(u)$, respectively. Therefore, the total number of conflicts by the greedy algorithm is:

$$\begin{aligned}
CF_G = \sum_e n(e) &\leq \sum_{e=(u,v)} \lfloor \frac{d_v(e) + d_u(e)}{k} \rfloor \\
&\leq \sum_v \sum_{e\in E(v)} \frac{d_v(e)}{k} = \frac{1}{k}\sum_v \sum_{i=0}^{d_v-1} i \\
&= \frac{1}{k}\sum_v \frac{d_v(d_v-1)}{2} = \frac{1}{k}\sum_v \frac{d_v^2}{2} - \frac{1}{k}\sum_v \frac{d_v}{2}
\end{aligned}$$

$$= \frac{1}{2}\sum_v \frac{d_v^2}{k} - \frac{|E|}{k}.$$

A.3 Proof of Theorem 2.4 Suppose that we have a simple graph $G = (V, E)$. It is known that finding the edge chromatic number $\chi'(G)$ of G is NP-hard (the edge chromatic number is the minimum number of colors for edge-coloring G) [9]. By the Vizing's theorem [10], the chromatic index of a simple graph G is Δ or $\Delta + 1$ where Δ is the maximum degree of any vertex $v \in V$.

Given a constant ϵ, let $G' = (V', E')$ be the graph which has $|V|^{\frac{1}{\epsilon}-1}$ copies of G. Note that $|V'| = |V|^{\frac{1}{\epsilon}}$. We set $C_G = C_v = \Delta$ If $\chi'(G) = \Delta$ then the optimal solution of the channel assignment problem is 0. Otherwise if $\chi'(G) = \Delta + 1$, then each of component of G' has at least one conflict and therefore, the optimal solution has at least $|V|^{\frac{1}{\epsilon}-1}$ conflicts, which is the same as $|V'|^{1-\epsilon}$. Thus if we have an approximation algorithm with additive term of $o(|V'|^{1-\epsilon})$ for a graph $G' = (V, E')$, we can decide the chromatic index of G, which is NP-hard. Contradiction.

A.4 Proof of Theorem 3.1 A node v is defined to have a balanced assignment if for each color i used by any edge in $E(v)$, the number of edges assigned to color i is exactly $\frac{d_v}{C_v}$. A network has the minimum number of conflicts if every node has a balanced assignment. Given an instance C of the problem 3SAT, we construct a graph G, in which each node has a balanced assignment if and only if C is satisfiable.

We need three types of components — inverting components, variable setting components, and satisfaction testing components. Figures 4 and 5 in Appendix show the components we need. In each component, a black or gray node has $C_v = 1$, and a white node has $C_v = 2$. In inverting components, if the input pair of edges use the same channel, the output pair should use different channels (and vice versa) for a white node to have a balanced assignment. In a component, a pair of input or output edges are said to be $true$ if the same channel is assigned to the pair, and $false$ if different channels are assigned to them. (To assign *true* to a pair of edges we may choose either channel 1 or 2.) The inverting component can be used to obtain the invert of a variable.

Using the variable setting components, we can set pairs of edges to be either true or false. We need to have as many pairs as there are appearances of variable v_i or $\neg v_i$ in C. Note that the specific channel assigned to each edge can be chosen as we want when we assign true or false to a pair of edges. For example, we can either use channel one or two for true assignments.

For each clause c_j in C, we have one satisfaction testing component (see Figure 5). In a satisfaction testing component, a white node has a balanced assignment if and only if at least one of three pairs is true. That is, if all three pairs are false, then we have exactly three edges with channel one and three edges with channel two for the three pairs, which prevents the white node from having a balanced assignment. On the other hand, if at least one is true, we can find an assignment of either $(5, 1)$ or $(4, 2)$ for the three pairs ((i, j) means that i edges have channel one and j edges have channel two), and there are balanced assignments for both cases.

A.5 Proof of Theorem 3.3 Let X_{ij} be 1 if e_i and e_j have the same color for $e_i, e_j \in E(v)$. For any vertex v with $C_v = 1$, all edges in $E(v)$ should have the same color in any solution. Therefore, we only consider edges $e_i, e_j \in E(v)$ for vertices with $C_v = 2$. Let $c(e)$ be the color assigned to edge e.

$$\begin{aligned}
E[X_{ij}] &= Pr(c(e_i) = c(e_j)) \\
&= 1 - Pr(c(e_i) \neq c(e_j)) \\
&= 1 - 2Pr(Y_{e_i} \cdot r \geq 0,\ Y_{e_j} \cdot r < 0) \\
&= 1 - \frac{\arccos(Y_{e_i} \cdot Y_{e_j})}{\pi} \\
&\leq 1 - \frac{\alpha}{2}(1 - Y_{e_i} \cdot Y_{e_j}) \\
&= (1 - \alpha) + \frac{\alpha}{2}(Y_{e_i} \cdot Y_{e_j} + 1)
\end{aligned}$$

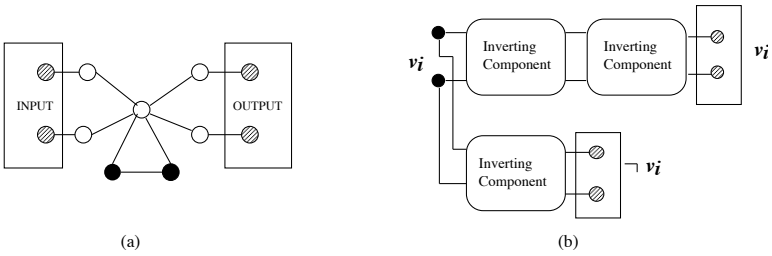


Figure 4: Black and gray nodes have $C_v = 1$ and white nodes have $C_v = 2$. $C_G = 2$. (a) Inverting Components (b) Variable setting components.

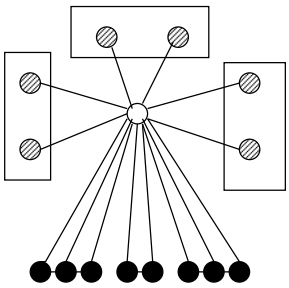


Figure 5: Satisfaction Testing Component

The total number of such conflicts is $X = \sum X_{ij}$. Let $OPT(S)$ ($S \subseteq V$) be the optimal number of conflicts at vertices in S and $|E(S)|$ be $\sum_{v \in S} \frac{d_v}{2}$.

$$\begin{aligned}
\sum_{v \in V_k} \sum_{e_i, e_j \in E(v)} E[X_{ij}] &= \sum_{v \in V_k} \sum_{e_i, e_j \in E(v)} ((1-\alpha) + \frac{\alpha}{2}(Y_{e_i} \cdot Y_{e_j} + 1)) \\
&\leq \sum_{v \in V_k} \sum_{e_i, e_j \in E(v)} (1-\alpha) + \alpha OPT(V_k) \\
&\leq (1-\alpha) \sum_{v \in V_k} \frac{d_v^2 - d_v}{2} + \alpha OPT(V_k) \\
&= (1-\alpha) \sum_{v \in V_k} \frac{d_v^2}{2} - (1-\alpha)|E(V_k)| + \alpha OPT(V_k) \\
&= 2(1-\alpha)(\sum_{v \in V_k} \frac{d_v^2}{4} - |E(V_k)|) + (1-\alpha)|E(V_k)| + \alpha OPT(V_k) \\
&\leq 2(1-\alpha) OPT(V_k) + \alpha OPT(V_k) + (1-\alpha)|E(V_k)| \\
&\leq (2-\alpha) OPT(V_k) + (1-\alpha)|E(V_k)|.
\end{aligned}$$

By Lemma 3.1 and the fact that $CF(V_1) = OPT(V_1)$, we have the theorem.

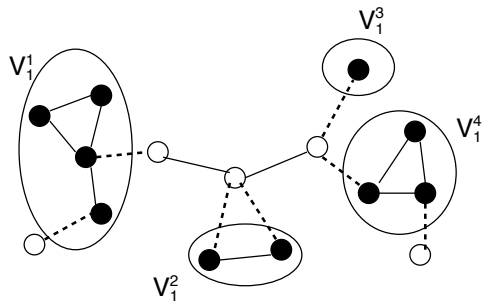


Figure 6: The figure show an example of clusters V_1^i when $C_v = 1$ or k. Black nodes have only one wireless card and white nodes have k wireless cards. Dotted lines belong to B_1^i.