Fairness by Sharing: Split Channel Allocation for Cognitive Radio Networks

Tarun Bansal, Dong Li and Prasun Sinha
Dept. of Computer Science and Engineering, Ohio State University, Columbus, Ohio 43210
Email: {bansal, lido, prasun}@cse.ohio-state.edu

***Abstract*—**
Licensed white space channels can now be used opportunistically by unlicensed users, provided the channels are relinquished when needed by the primary users. With an estimated 85% of channels that are unused, this vast spectrum if used judiciously can revolutionize high speed wireless data services. We focus on the problem of multi-channel allocation for access-points. The objectives of our design include fairness, high throughput, low overhead, and low rate of channel reconfigurations. The resulting problem definitions are challenging due to their combinatorial nature, and channel dynamics of the primary as well as the secondary users across space and time. We allow neighboring nodes to share channels among themselves leading to a new class of problems, that to the best of our knowledge has not been studied before in networking or the graph theory literature. Our channel allocation problem translates to a particular problem of this class that we call as *Split Coloring Problem*. We also introduce the concept of effective neighbor count that helps us in estimating the expected throughput that access points will receive. Our solution comprises of a centralized algorithm and its localized version that work together to meet the objectives. The centralized approximation algorithm is shown to have tight bounds in terms of fairness while still providing high system throughput. Results from extensive realistic simulations show that our algorithm typically performs 2-3 times better than baseline algorithms.

I. Introduction

The Cognitive Radio (CR) technology has gained increased attention due to an FCC mandate that now allows unlicensed radios to operate in the unused portions of the UHF band. It is estimated that 85% of these channels, also called the white space channels, are currently unoccupied [1]. Such channels however, need to be relinquished when primary (or incumbent) users begin using it. Solutions that can opportunistically use such channels can help alleviate the congestion in the ISM bands. In one of the first such efforts, the nation's first TV white space network was deployed in Claudville, VA in late 2009 [1]. We envision a proliferation of such white space based wireless solutions that take advantage of the *newly found* vast spectrum. If used judiciously, it can revolutionize high speed wireless data services using both infrastructured (access-point based) [2] and peer-to-peer architectures. Our focus in this paper is on the former.

Advances in hardware technologies have made it possible to simultaneously use the capacity of a large number of channels that may or may not be contiguous [3] by the use of Non-Contigous OFDMA [4], [5]. Allocating multiple channels increases the aggregate throughput. Nevertheless, if the hardware restricts operation to a single channel, then multi-channel allocation can be leveraged to rapidly migrate to other assigned channels when primary users become active.

We define a channel assignment of white space channels to a collection of Access-Points (AP) to be *feasible* if a channel assigned to an AP does not interfere with any primary user. In addition, the following properties are important to achieve: (1) *Fairness* in the number of channels assigned to APs; (2) *High throughput* of the system; (3) *Low time and message overhead*; and (4) *Fewer channel reassignments*.

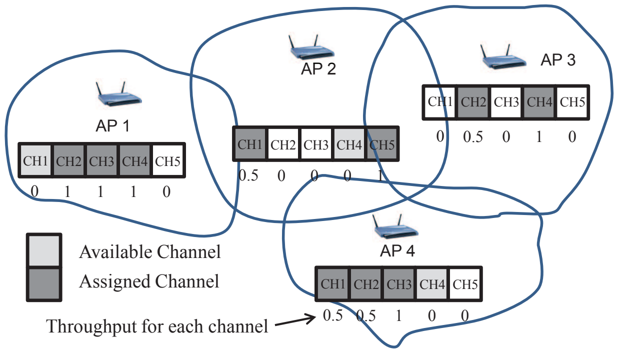


Fig. 1. A fair channel allocation using 5 channels. The bounded regions around the APs represent the interference ranges.

We seek to design a comprehensive and practical solution that addresses all the above design objectives. But, the problem is particularly challenging due to its combinatorial nature of channel selection problems, and channel dynamics of the primary as well as the secondary users across space and time. The combinatorial structure is shown through an example in Figure 1 where the solution for one specific objective, e.g., to achieve the largest lexicographic sequence of allocated channels to each AP, is shown. In our system model, we allow neighboring nodes to share the same channel. Throughput shown in the figure are under the ideal conditions of no loss due to collisions. For example, AP 2 and AP 4 share channel 1, therefore, both of them receive a throughput of 0.5 on that channel. Sharing allows every node to get at least 1.5 units of throughput (Fig. 1). On the other hand, without sharing, it is imposible to assign every node a throughput of more than 1 unit. Clearly, in realistic scenarios by sharing channels, some of the throughput of the channel will be lost due to collisions. Even in such scenarios, sharing is expected to improve fairness. Sharing also allows every node to have at least one channel for communication even when the number of available channnels are limited.

This sharing of channels among nodes gives rise to a new set of problems which we call as *split coloring*. To the best of our knowledge, this has not been studied in networking or graph theory literature. Split coloring is quite distinct from other coloring problems previously studied like list coloring, partial coloring, fractional coloring, defective coloring etc. (see Section VI for details).

Achieving fairness and high throughput in a system where neighboring nodes can share the same color is particularly challenging due to the following reasons: (1) When multiple neighbors are sharing the same channel, it is hard to approximate the amount of throughput that each of the APs would receive; (2) The transmission and interference range of APs may not follow a disk model which makes modelling the collisions difficult; (3) Ability of APs to transmit on multiple channels further adds to the complexity as APs may have different transmission and interference range on different channels; and (4) Due to continuous arrival and departure of licensed users, an AP's available throughput may suddenly decrease by a large amount. In such a case, reallocation of channels among APs may require considerable number of reconfigurations.

Our solution is comprised of a centralized algorithm, $Split$ and its localized version, $lSplit$, both of which seek to find a fair assignment of channels to the APs. To minimize the number of channel reassignments, $lSplit$ operates in between periodic executions of the $Split$ algorithm.

This paper makes the following key contributions:

- We propose a new system model in which neighboring nodes can share channels among themselves. We call the problem of fair allocation under this system model as *Split Coloring* and prove that this problem is NP-Hard.
- Using a simple MAC layer model, we introduce the concept of *Effective Neighbor Count*. This helps us in approximating the throughput that a node will receive when it shares a channel with its neighbors.
- An approximation algorithm, $Split$ with provable performance bounds is proposed. The bound on the fairness achieved by $Split$ is proven to be tight while the total throughput achieved is also shown to be *high*. Its localized version $lSplit$ is invoked in between the executions of $Split$ to reduce the number of channel reconfigurations.
- Through extensive realistic simulations that also consider sensing uncertainties, we demonstrate the performance improvement of our solution with respect to other algorithms. Our solution typically performs at least 2-3 times better than the baseline algorithms.

II. System Model

We consider N Radio Access Points (APs) $r_1, r_2, \ldots, r_N$, which are located in a 2D space. We assume that every access point in our system is capable of operating on multiple channels. Two APs are *neighbors* of each other if a client associated with one of the APs can be interfered by the other AP on some channel.

The universal channel set $\mathcal{U}$ consists of channels $c_1, c_2, \ldots, c_{|\mathcal{U}|}$. Even if a node has the capability of communicating on a particular channel, it might not be able to do so because of the presence of a primary user in the neighborhood. Every CR node is capable of operating on multiple channels which has been made possible by Non Contiguous OFDMA technology [4], [5]. The set of channels on which a node r_i can communicate without interfering with primary users is termed as its *availability set* and is denoted by $W(r_i)$. The set of channels that are assigned to node r_i is denoted by $\mathcal{A}(r_i)$. We denote the assignment of channels over the whole network by $\mathcal{A}$. We assume that APs communicate with a central entity (commonly known as *Channel Assignment Server (CAS)*) over a wired backbone, such as the Internet. *CAS* is responsible for taking decisions regarding channel assignment and propagating them back to the nodes.

TABLE I
SYMBOLS USED

Symbol	**Description**
$\mathcal{U}$	Universal channel set
N	Number of nodes in the network
Δ	Maximum neighbors of any AP among all APs
r_i	(Radio) Node i
c_i	Channel i in the universal channel set $\mathcal{U}$
$W(r_i)$	Availability set of $r_i : W(r_i) \subseteq \mathcal{U}$
$N(r_i)$	Set of neighboring APs of r_i
$\mathcal{A}(r_i)$	Set of channels assigned to r_i: $\mathcal{A}(r_i) \subseteq W(r_i)$
$\mathcal{A}$	Assignment of channels to over all the nodes
$\mathcal{T}(\mathcal{A}, r_i, c_j)$	Effective throughput available to r_i on c_j
$\mathcal{T}(\mathcal{A}, r_i)$	Total throughput available to node r_i

For the sake of simplicity, we assume that all channels provide the same amount of bandwidth. We allow neighboring nodes to share channels. If a node r_i is assigned a channel c_j, then $\mathcal{T}(\mathcal{A}, r_i, c_j)$ (Table I) denotes the throughput that node r_i receives on channel c_j. Clearly, $\mathcal{T}(\mathcal{A}, r_i, c_j)$ will depend on the assignment of c_j in the neighborhood of r_i. If a channel c_j is assigned only to r_i and not to any neighbor of r_i, then the amount of throughput that r_i can receive on c_j is 1 unit. The total throughput that r_i would get will be $\sum_{1 \le j \le |\mathcal{U}|} \mathcal{T}(\mathcal{A}, r_i, c_j)$ which is denoted by $\mathcal{T}(\mathcal{A}, r_i)$. We calculate the throughput under *saturation* conditions which means that the transmission queues of all nodes are backlogged. In practice, if this assumption is not correct, then nodes will experience higher throughput.

Our analysis assumes that the following statement is true for the underlying MAC protocol: *When the number of users contending for the channel increases, the total throughput of the channel over all the users approaches a non-zero constant*. Authors in [6] have verified through analysis and experiments that this condition is indeed true for 802.11 protocol. Various heuristical techniques can be used for estimating the throughput of a node under shared channel model. However, most of the techniques available in literature either apply only to single hop networks [6], or only show experimental analysis [7], or can be used only after the channel assignment has been completed [8], [9]. Authors in [10] have presented a throughput estimation model which is applicable to mesh networks. Estimating throughput in mesh networks is different from that in Access Point based networks. In latter, we also need to concern about the interference faced by the clients associated with the AP. To that end, we present a simple model that we use for estimating the expected throughput. In section V, using *ns-3*, we show that our formula for estimating throughput is able to capture the pattern in variation of actual throughput even with varying degree of APs.

A. Throughput Estimation Model

The throughput that a node r_i will receive depends on a multitude of factors, including but not limited to the location of all APs and their associated clients, actual transmission and interference range of nodes, and the amount of downlink and uplink traffic between all nodes. However, the following two factors are the most important: (i) Number of neighbors of r_i over channel c_j and (ii) Fraction of clients associated with r_i that lie in the interference range of neighboring APs. In order to simplify the expected throughput calculation, we introduce the concept of *effective neighbor count* that takes both these factors into account. Note that in order to keep the model simple, we only consider the downlink traffic, though it is possible to arrive at a more accurate model by considering both downlink and uplink model. By using the concept of effective neighbor count, it becomes easier to quantify the expected throughput that a node will receive after sharing the channel.

We first calculate the probability that a transmission by r_i will be unsuccessful. We categorize the clients of r_i into groups depending on how many neighboring APs they can be interfered from. Let G_{ik} be the fraction of clients of r_i that lie within the interference range of exactly k neighboring APs of r_i. The probability that a transmission by r_i will not be successful is $1-\sum_{0\le k\le deg(r_i)} G_{ik}(1-p)^k$ where $deg(r_i)$ is the number of neighboring APs of r_i and p is the probability of transmission by r_i. We assume that neighboring nodes also have the same probability of transmission. In other words, we assume that on a long term basis, all nodes have equal chance of gaining access to the common channel. This expression can be approximated as $1-\sum_{0\le k\le deg(r_i)} G_{ik}(1-kp)$ (when the number of interfering APs are large) which is same as $p\sum_{0\le k\le deg(r_i)} kG_{ik}$, since $\sum_{0\le k\le deg(r_i)} G_{ik}=1$.

We define the effective neighbor count of r_i on channel c_j as: $E(r_i,c_j)=\sum_{0\le k\le deg(r_i)} kG_{ik}$ which can be used to deduce the probability that a given transmission of r_i on channel c_j is *not successful*. We assume that there are $E(r_i,c_j)$ other neighbors of r_i which also transmit on c_j and that every client associated with r_i lies in the interference range of all these neighboring APs. The probability that a transmission of r_i on c_j will be unsuccessful will be $1-(1-p)^{E(r_i,c_j)}$. Substituting the value of $E(r_i,c_j)$, it can be verified that $1-(1-p)^{E(r_i,c_j)}$ is approximately same as $p\sum_{0\le k\le deg(r_i)} kG_{ik}$ when number of interfering APs is large. This is same as the probability of unsucessful transmission that we calculated before. Therefore, by using the concept of effective neighbor count, the probability of a transmission being unsuccessful still remains valid. In Section IV, we will describe an approach that can be used by APs for calculating $E(r_i,c_j)$.

If r_i has an effective neighbor count of $E(r_i,c_j)$ on channel c_j, then the throughput available to r_i on c_j would be $\mathcal{T}(\mathcal{A},r_i,c_j)=p(1-p)^{E(r_i,c_j)}$ where $p=\frac{1}{1+E(r_i,c_j)}$ since this value of p minimzes the number of collisions among neighbors.

III. Problem Formulation

The *primary objective* is to assign each node a subset of channels from its availability set such that the non-decreasing lexicographic sequence of the throughputs assigned to all APs is maximized. Recall that a non-decreasing lexicographic sequence is greater than the other sequence, if the first i terms of both sequences are equal and $i+1$'th term of the former is greater than the corresponding term of the latter for some $i\ge 0$.

Recall that $\mathcal{A}$ represents the channel assignment for all the nodes. If the solution space of assignments is represented by $\mathbb{A}$, then the primary objective can be written as:

$$\text{Problem 1}: \quad \max_{\mathcal{A}\in\mathbb{A}}(\text{Lexicographic sequence of throughputs}, \mathcal{T}(\mathcal{A},r_i))$$

subject to:

$$\begin{aligned}
\mathcal{T}(\mathcal{A},r_i) &= \sum_{1\le j\le|\mathcal{U}|}\mathcal{T}(\mathcal{A},r_i,c_j)\\
\mathcal{T}(\mathcal{A},r_i,c_j) &= \frac{1}{1+E(r_i,c_j)}\left(1-\frac{1}{1+E(r_i,c_j)}\right)^{E(r_i,c_j)}\\
&\quad \forall i:\ 1\le i\le N, \forall j: 1\le j\le|\mathcal{U}|\\
E(r_i,c_j) &= \begin{cases}\sum_{0\le k\le deg(r_i)} kG_{ik} & \text{if } c_j\in\mathcal{A}(r_i)\\ 0 & \text{otherwise}\end{cases}\\
\mathcal{A}(r_i) &\subseteq W(r_i)\subseteq\mathcal{U}
\end{aligned}$$

A. Reduction to Split Graph Coloring

We convert our problem statement to an equivalent *graph coloring problem*. We construct the conflict graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ such that each node in $\mathcal{V}$ corresponds to an AP in the network. An edge is drawn between two vertices of $\mathcal{V}$ if the APs corresponding to these vertices are neighbors of each other. Each channel in $\mathcal{U}$ corresponds to a color in the multi-coloring problem. A node can be colored with only those colors that correspond to the channels in its availability set. Allocation of a channel to a node is equivalent to coloring the corresponding vertex in the graph with the equivalent color. However, a color can be *split* among neighboring nodes and in that case every node gets a certain *part* of the color depending on the interference between the clients of this node and its neighbors. The total share that a node gets over all the colors is called its *effective color count*. The objective as before is to maximize the lexicographic sequence of effective color count of each node. *Splitting of colors* leads to a new class of problems that have not been discussed before in either graph theory literature or networking. We call our particular problem (Problem 1) as *split coloring problem*. In the following discussion, we will use the term "effective color count" and "throughput" interchangably.

We prove that the Split Coloring Problem is NP-Hard. To that end, next two Lemmas characterize the relationship between split coloring and *proper vertex coloring*.

Lemma 3.1: Given a graph $\mathcal{G}$ and its *proper coloring* such that every node has at least 1 color assigned to it, then it is possible to derive the split coloring of $\mathcal{G}$ in polynomial time with the same number of colors, such that every node has at least 1 unit of throughput assigned.

Proof: Since any proper coloring is also a split coloring, this lemma trivially holds true. ■

Lemma 3.2: Consider a graph $\mathcal{G}$ in which all clients associated with every node can be interfered by all neighboring APs of their associated AP. Given a split coloring of $\mathcal{G}$ such that every node has at least 1 unit of throughput assigned to it, then in polynomial time, it is possible to arrive at a proper coloring of $\mathcal{G}$ using the same number of colors.

Proof: See Appendix. ■

On the basis of the Lemmas 3.1 and 3.2, we make the final claim about the hardness of Problem 1.

Theorem 3.1: Problem 1 (or Split Coloring Problem) is NP-Hard.

Proof: Lemma 3.1 and 3.2 shows that the problem *Is it possible to assign 1 unit of throughput to every node using k colors by split coloring* is at least as hard as *k-coloring problem*. Since, the latter problem is known to be NP-Hard, therefore the former problem must be NP-Hard as well. Also, Problem 1 is a specific version of the former problem, therefore Problem 1 must be NP-Hard as well. ■

IV. Split Channel Assignment Algorithm

As discussed in Section II, all APs send their sensing data (*viz.* Channels available and effective neighbor count) to the *Channel Assignment Server (CAS)*. APs can deduce this information through various means. Channels available can be deduced from periodic sensing of the spectrum [3]. One way to figure out the effective neighbor count for each AP is to use geometrical disk model for transmission and interference range and calculate the fraction of overlaps. However, such a model is not accurate in practice. Therefore, in order to arrive at more realistic values, we run an initial measurement period. Initially, using the geographical location of APs and their approximate transmission and interference range, *CAS* builds a conservative conflict graph ensuring that any two APs that are actual neighbors of each (as per the definition given in Section II) are also neighbors in this conservative conflict graph.

In each time slot of the measuring period, two neighboring AP pairs (say r_i and r_k) of the conservative conflict graph transmit at the same time. All clients associated with r_i can accurately conclude if they are in the interference range of r_k on the basis of if they are able to receive clear transmission from r_i. Similarly, clients associated with r_k can also make this conclusion. This procedure needs to be repeated over each channel for every pair of neighboring nodes. Clearly, the time required to do this would be the same as the edge-chromatic number of the conservative conflict graph multiplied by the number of channels. Therefore, the length of this measurement period will be at most $|\mathcal{U}|(1 + \Delta_{approx})$ time slots where Δ_{approx} is the degree of the conservative conflict graph. After this measurement period is over, every client associated with r_i will know how many other APs can interfere with its reception for every channel. This measurement can be used by r_i to calculate $G_{il}\ \forall l$ and hence, effective neighbor count can be calculated using the expression discussed in Section II. Next, we describe our algorithm $Split$ for split channel assignment.

A. Algorithm Description

Phase 1 of the algorithm involves proper coloring of the graph (Line 4-10). This ensures that the number of conflicting assignments is minimized, which helps us in increasing the throughput of the system. The set of colors that are assigned to a node r_i in Phase 1 are called the *principal colors* of r_i and this set is denoted by $P(r_i)$.

Thereafter, in order to ensure fairness, $Split$ rearranges the colors in Phase 2 by splitting colors across neighbors. Again, splitting is done conservatively starting with at most 1 conflicting assignment and going up to $1 + \Delta$ conflicting assignments (Line 12-20). However, color is assigned to a node only if it does not reduce the lexicographic sequence of throughputs (Line 16).

Due to the different availability set of nodes, it is possible that the color assignment at this point is still unfair due to bad choice of color sequence in Line 4 or Line 13. Therefore, in order to correct that, in Phase 3, algorithm $Split$ tries to even out the throughputs by exploring the pair of neighboring nodes in which the principal color of the node with lower throughput is shared with the other node that has higher throughput (Line 22-23). Clearly, by removing this principal color from that neighbor's assignment set, the node with lower throughput will experience an increase in its throughput.

Note that time complexity of all the three phases of the algorithm is polynomial and is upper bounded by $O(|\mathcal{U}|N\Delta)$ since in every phase either colors are only added to the availability set of nodes or only removed. The message complexity is $O(N)$. Here Δ is the maximum degree of a node in $\mathcal{G}$.

Algorithm 1: Algorithm $Split$

```
1  Input: Graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ and the availability set $W(r_i) \forall i : 1 \le i \le N$
2  Output: A fair color assignment
3  //Phase 1: Assign proper colors
4  foreach color $j = 1$ to $|\mathcal{U}|$ do
5      while true do
6          $S_1 \leftarrow \{r_m : c_j \in W(r_m)$ and $c_j \notin \mathcal{A}(r_m)$ and $c_j$ is not
           assigned to any neighbor of $r_m\}$
7          if $S_1 = \Phi$ then
8              break
9          $i \leftarrow \operatorname{argmin}_{m:r_m \in S_1} \mathcal{T}(\mathcal{A}, r_m)$
10         $\mathcal{A}(r_i) \leftarrow \mathcal{A}(r_i) \cup \{c_j\}, P(r_i) \leftarrow P(r_i) \cup \{c_j\}$
11 //Phase 2: Share colors to improve fairness
12 foreach max number of conflicts $k = 1$ to $1 + \Delta$ do
13     foreach color $j = 1$ to $|\mathcal{U}|$ do
14         while true do
15             $S_2 \leftarrow \{r_m : c_j \in W(r_m)$ and $c_j \notin \mathcal{A}(r_m)$ and
               assigning $c_j$ to $r_m$ will not lead to any node sharing its
               color with more than $k$ neighbors$\}$
16             $S_3 \leftarrow \{r_m : r_m \in S_2$ and assigning $c_j$ to $r_m$ does not
               make the new throughput of any neighbor of $r_m$ less
               than current throughput of $r_m\}$
17             if $S_3 = \Phi$ then
18                 break
19             $i \leftarrow \operatorname{argmin}_{m:r_m \in S_3} \mathcal{T}(\mathcal{A}, r_m)$
20             $\mathcal{A}(r_i) \leftarrow \mathcal{A}(r_i) \cup \{c_j\}$
21 //Phase 3: Give preference to proper color assignment
22 while there exists neighbors $r_i$ and $r_k$ such that
   $\mathcal{T}(\mathcal{A}, r_i) \le \mathcal{T}(\mathcal{A}, r_k)$, $c_j \in P(r_i)$ and $c_j \in \mathcal{A}(r_k)$ do
23     $\mathcal{A}(r_k) \leftarrow A(r_k) \backslash \{c_j\}$
```

B. Performance Analysis

Though our algorithm maximizes the non-decreasing lexicographic sequence of assigned throughput, our analysis is restricted to only the first term of the sequence (otherwise known as the *max-min term*). For that purpose, we next compare the performance of $Split$ with the exponential time optimal algorithm, OPT. Here, the OPT is defined as the algorithm that maximizes the lexicographic sequence of problem 1. Let $\Upsilon_{Split} = \min_{1 \le i \le N} \mathcal{T}(\mathcal{A}_{Split}, r_i)$; and $\Upsilon_{OPT} = \min_{1 \le i \le N} \mathcal{T}(\mathcal{A}_{OPT}, r_i)$.

Let r_k be a neighbor of r_i. Let x_{ki} be the total throughput of r_k over all colors that satisfy the following two conditions: (i) The color belongs to the availability set of r_i; and (ii) Among all nodes in the neighborhood of r_i, the color was first assigned by $Split$ to r_k. The proof for the following related lemmas and the final theorem can be found in Appendix.

Lemma 4.1: At the end of Phase 1, $\mathcal{T}(\mathcal{A}_{Split}, r_i) \ge \frac{|W(r_i)|-\Delta}{1+\Delta} \; \forall r_i$.

Lemma 4.2: Consider a node r_i whose throughput is less than that of all its neighbors. Then at the end of Phase 3, $\mathcal{T}(\mathcal{A}_{Split}, r_i) \ge \frac{|W(r_i)|-\Delta}{1+\Delta}$.

Theorem 4.1: For a given graph and a set of colors,

$$\Upsilon_{Split} \ge \frac{\Upsilon_{OPT} - \Delta}{1+\Delta}$$

Next, we prove that there exists no polynomial time algorithm that can guarantee bounds on the first term of the lexicographic throughput sequence that are tighter than those of $Split$. By contradiction, we assume that there exists a polynomial time algorithm (say $POLY$) that guarantees $\Upsilon_{POLY} \ge \frac{\Upsilon_{OPT} - f(\Delta)}{g(\Delta)}$. We will now prove that for any such algorithm, it is required that $f(\Delta) = \Omega(\Delta)$ and $g(\Delta) = \Omega(\Delta)$. Before that, we show that no polynomial time algorithm can approximate the chromatic number of a graph with an error factor of less than Δ. The proof of the following lemmas can be found in the appendix.

Lemma 4.3: For an arbitrary graph $\mathcal{G}$, if a polynomial time algorithm approximates the chromatic number of $\mathcal{G}$ $(\chi(\mathcal{G}))$ with an approximation of $\frac{y-d}{1+z(\Delta)} \le \chi(\mathcal{G}) \le y$ where d and y are constants, then the following must hold true: $z(\Delta) = \Omega(\Delta)$.

Using the above lemma, we show a lower bound on both the functions *viz.* $f(\Delta)$ and $g(\Delta)$.

Lemma 4.4: $f(\Delta) = \Omega(\Delta)$

Lemma 4.5: $g(\Delta) = \Omega(\Delta)$

Theorem 4.2: $Split$ provides tight approximate bounds on the first term of the lexicographic non decreasing sequence of assigned throughputs.

Proof: Directly follows from Theorem 4.1, Lemma 4.4 and Lemma 4.5. ■

We have shown that $Split$ provides a tight bound on the first term of the throughput sequence. Another important measure of a channel allocation algorithm is the throughput provided. The throughput of the system can be calculated as $\sum_{1 \le i \le N} \mathcal{T}(\mathcal{A}, r_i)$. In the appendix, we show that $Split$ guarantees high system throughput by showing that total throughput assigned over the whole network is with in a bounded factor of the throughput assigned by the exponential time optimal algorithm, OPT. This proves that the throughput of the network colored by algorithm $Split$ is high.

Theorem 4.3: The system throughput guaranteed by $Split$ is at most orderwise Δ times smaller than the system throughput given by OPT.

Localized Algorithm: We now present $lSplit$ which is a *localized* version of $Split$. $lSplit$ will be invoked when the throughput of a node decreases considerably as compared to its neighbors. Algorithm $Split$ guarantees better fairness than $lSplit$, however it may end up reassigning channels for all the APs resulting in a lot of overhead. To avoid that, we run $lSplit$ in the locality of the node that has less throughput. $lSplit$ helps us to achieve *local fairness* in a reactive fashion while $Split$ is invoked proactively at regular intervals and ensures fairness across the whole network. As before, APs communicate to the CAS over the backbone network. Execution of $lSplit$ at CAS involves executing a limited hop version of $Split$ in the locality of node with low throughput. However, if the number of hops are not sufficient enough such that the throughput of some nodes is still considerably less than that of their neighbors, then the number of hops are doubled until all APs in that hop-restricted network have comparable throughput as compared to their neighbors. This helps us in reducing the number of APs that need to be reconfigured. The criterion for comparing the throughputs depends on the particular implementation. In the next section, we discuss the results obtained through various simulations.

V. Simulations

A. Throughput Estimation Model

In Section II, we gave a formula for estimating the throughput of an AP which was based on the effective neighbor count of that AP. In order to verify the proposed throughput model, we used *ns-3* to simulate a network of varying size by increasing the number of APs from 15 to 100. Every AP had some clients associated with it, whose count varied from 30-60 clients per AP. The rest of the simulation parameters are shown in Table II. We ran multiple scenarios, each simulation involving transmission of 1000 packets to every associated client by each AP. Simulations show that our formula for estimating the throughput is able to capture the pattern in variation of actual throughput due to varying degree of APs (Fig. 2). The difference in the actual throughput and the estimated throughput is almost proportional to the actual throughput which implies that estimated throughput is almost linearly proportional to actual throughput. This is sufficient for $Split$ to work correctly as it only compares the throughput of nodes, rather than using the absolute values.

As is clear from Fig. 2, our throughput model is very close to reality when the degree of the AP is high as compared to when the degree is low. We suspect that this is because the approximation used in deriving the effective neighbor count is true only when the clients are interfered by a large number of neighboring APs (see Section II). Also, at low degree, the losses due to networking overheads (headers etc.) are much more pronounced than the losses due to collisions. The former losses are not taken into account by our model, which explains its bad performance when the degree of AP is small.

B. Comparison with baseline algorithms

In order to evaluate the performance of our algorithm, we used a C++ based simulator. The various parameters were set up as shown in Table II. Even though the size of the universal channel set $\mathcal{U}$ is 80, still due to hardware differences, every node had access to an average of 40 randomly picked channels from the set $\mathcal{U}$. Every simulation was repeated 20 times and the mean along with its 95% confidence interval was plotted for

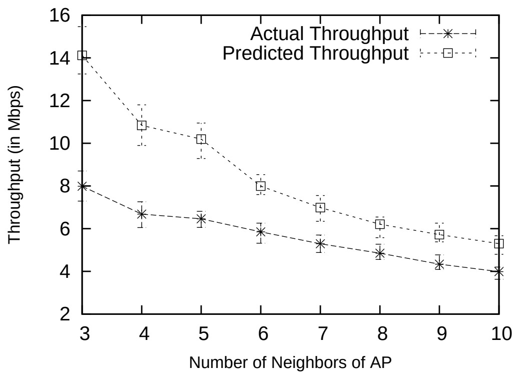


Fig. 2. Plot of throughput of various APs with varying degree

TABLE II
DEFAULT SIMULATION PARAMETERS

Variable	**Value**
Size of field	500m X 500m
Number of primary users	10
Number of APs	300
Interference range of primary users	210m
Interference range of APs	70m
Size of universal channel set	80
Average size of $\mathcal{A}_i$	40

every measurement. In order to better analyze the performance of our algorithm, we also designed 4 other algorithms:

1) *Proper*: This algorithm only does a *proper coloring* of the graph. This means that no two neighboring nodes are assigned the same channel.
2) *AssignAll*: For all nodes, this algorithm assigns all the colors in that node's availability set to its assignment set.
3) *BF (4min)*: This Brute-Force algorithm sequentially generates a random assignment of colors. The assignment which generates the maximum value of minimum bandwidth among all nodes is saved. The algorithm is terminated after 4 minutes of execution. The probability that a color would be assigned to a node varies from $0.1 - 1.0$ with different executions.
4) *Naive*: This algorithm first does a proper coloring of the graph. After that, it searches for node pairs in which bandwidth of one node (say r_i) is lesser than that of other (say r_j). For such pairs, it finds a color that is not assigned to r_i but assigned to r_j and assigns that color to r_i as well.

Parameters: In order to study the performance of our algorithm, we studied different performance parameters under diverse conditions: *(1)* Minimum throughput assigned over all nodes *(2)* Total throughput of the system *(3)* Execution time *(4)* Number of reconfigurations. Note that this list covers all the requirements that we had discussed in Section I.

1) Increasing density of APs: In the first scenario, we increased the number of APs in the field. As we can see from Figure 3 (next page) that with increase in number of APs, in terms of minimum throughput, our algorithm performs 2-3 times better than the baseline algorithms. We also observe that with increase in number of APs, *Proper* was not able to assign at least 1 color to every node. This demonstrates the advantages of shared coloring as $Split$ is able to ensure better fairness compared to *Proper*. However, *Proper* was able to assign more throughput to the system compared to

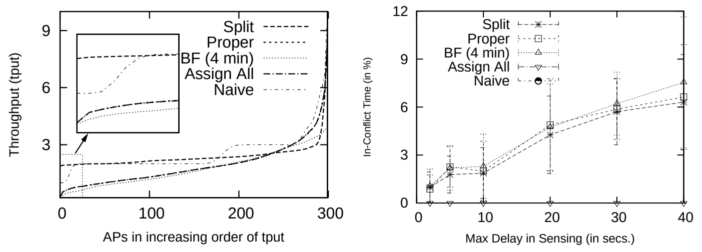


Fig. 4. Lexicographic ordering and In-conflict time

our algorithm. This is expected as *Proper* does not assign same color to neighboring nodes and this keeps the system throughput high. We can also see that the system throughput achieved by $Split$ starts decreasing after number of APs increase beyond 600. This is because at that point, the number of available colors are not sufficient enough and $Split$ employs extensive channel sharing in order to increase the fairness and this results in decreased throughput. However, note that even though the system throughput is low, but still $Split$ is able to perform well in terms of minimum throughput assigned over all nodes.

A detailed analysis of lexicographic ordering (Fig. 4) of nodes with respect to their color count verified that the other algorithms assign colors to only a few nodes without giving much preference to other nodes with lower color count. We also observed that *Proper* allocates minimum number of colors, followed by $Split$ (Fig. 3). Further, it was observed during the simulations that all the algorithms terminate within 5 seconds while *BF (4 min)* took 240 seconds to complete. It is also remarkable that results from *AssignAll* and *BF (4min)* were quite close to each other in all the simulations. We observed that *BF (4min)* maximized the minimum throughput when the probability of channel assignment was quite high which explains its closeness with *AssignAll*.

2) Increasing number of channels: Here, we varied the number of channels in the Universal Channel Set (Fig. 5). As expected, the minimum bandwidth increases with an increase in number of channels. $Split$ performs better than other algorithms in terms of fairness, while as before *Proper* performs better than $Split$ in terms of throughput.

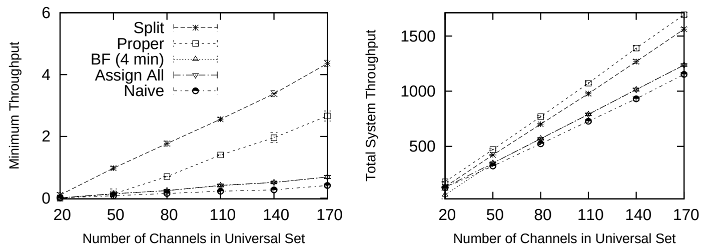
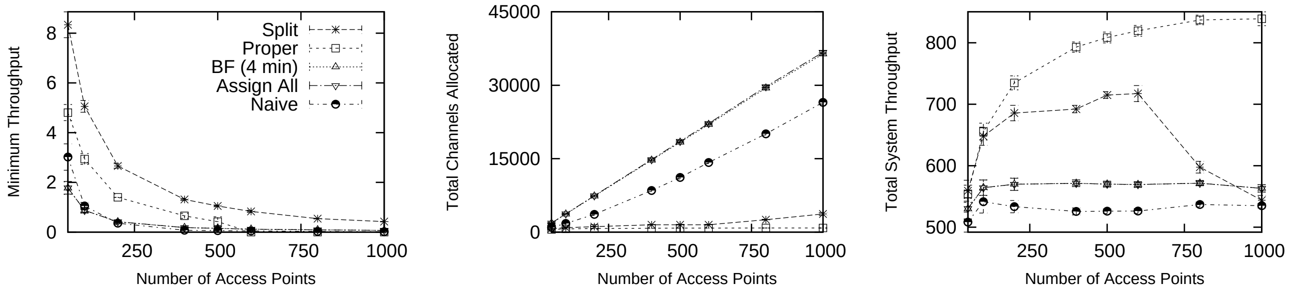


Fig. 5. With increasing number of channels

3) Behavior with time: In order to study the interaction between $Split$ and $lSplit$, we simulated the arrival and departure of primary users with time. While keeping the churn rate of primary users fixed at 0.3 per second, we studied how the two algorithms together behave (see Fig. 6). As we can see from the figure, the miniumum throughput assigned by $lSplit$ decreases with time though it still stays above *Naive* and *BF (4*



(a) Throughput (b) Total Channels Assigned (c) Total System Throughput

Fig. 3. Simulation results with varying number of APs

min). The invocation of $Split$ globally at every 300 seconds *restores* the system to a fair state. The advantage of the localized algorithm is that it requires far less reconfigurations at APs as compared to centralized algorithms. The figure shows that the reconfigurations made by our algorithm were far less than the reconfigurations required by other baseline algorithms. We define the total number of reconfigurations for an algorithm as $\sum_{1\leq i\leq N}(|\mathcal{A}_{r_i(old)} \setminus \mathcal{A}_{r_i(new)}| + |\mathcal{A}_{r_i(new)} \setminus \mathcal{A}_{r_i(old)}|)$ where $\mathcal{A}_{r_i(old)}$ and $\mathcal{A}_{r_i(new)}$ are respectively the assignment sets of the node r_{r_i} before and after the execution of channel assignment algorithm. The number of reconfigurations done by $Split$ was less than 5,000, while baseline algorithms exceeded 300,000.

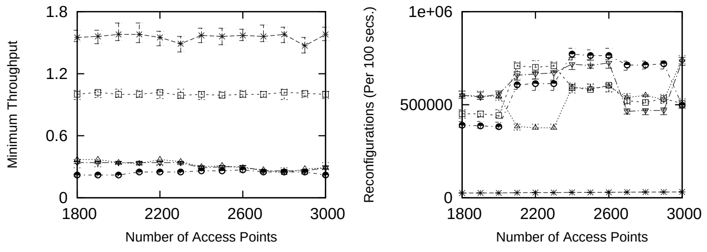


Fig. 6. Variation with Time

4) Sensing Inaccuracies: In practical systems, the spectrum sensing may not be *accurate* and *instant*. In order to measure the effect of real life inacuracies in sensing, we induced a uniformly distributed random delay between the insertion/removal of a primary user and its sensing by the APs for all the algorithms. The delay introduced was randomly chosen between [0, *MaxDelay*]. Figure 4 shows the behavior of the system with increasing values of *MaxDelay*. Assignments for all the algorithms were observed to have a similar average percentage of time for which APs are in conflict with Primary user.

VI. Related Work

The problem of channel assignment has been extensively studied before in various domains *viz.* cognitive radios, cellular networks, multi-radio multi-channel networks, wireless networks and graph theory.

Cognitive Radio: Various protocols have been proposed in literature for channel assignment [11], [12]. In [13], authors have studied channel assignment in multi-radio wireless mesh networks. In their algorithm, every node calculates the rank of channels available to it on the basis of interference and channel utilization. These rankings are then forwarded to the Channel Assignment Server that assigns channels to nodes in a breadth-first fashion. However, none of these algorithms provide any worst case bounds on performance.

The distributed algorithm proposed by Gan *et al.* [14] involved APs bidding for channels in an auction. In [2], each node calculates a weight for each channel that is dependent on the usage of that channel and the number of APs operating on that channel. Finally, the AP selects those channels that have higher weight. Again, there are no bounds on the performance of their algorithm.

Cellular Networks: The problem of multicoloring a graph with fixed number of colors has been extensively studied in cellular networks [15]. In that domain, this problem has been heuristically solved by using a *central pool of channels* and *channel borrowing* [16] schemes. Unlike cognitive radio networks, in cellular networks the same set of channels are available at all nodes. Moreover, in that domain, the problem has been mostly studied under the assumption that the base stations are located in a hexagonal pattern [15].

Multi-radio multi-channel networks: Recently, a lot of research has focused on channel assignment problem for Multi-Radio Multi-Channel Multi-hop wireless networks. [17] proposes a greedy algorithm that tries to satisfy a given traffic demand for different source-destination pairs. Similarly, [18], [19] aim at assigning the channels in a way that maximizes the throughput of the network or satisfies a required flow. Clearly, all these approaches would not work in our problem scenario where the requirement is to assign channels to the neighboring nodes while ensuring *fairness*. Chandra *et al.* [20] proposed using variable channel width for fair channel allocation. However, if the number of available channels is small, then it becomes imperative to assign the same channel to neighboring nodes which cannot be handled by their approach. Similarly, [21], [22] also do not allow splitting of channels among neighbors.

Wireless Networks: In wireless networks, a related problem of assigning overlapping 802.11 channels while minimizing the interference has been studied. For example, [23], [24] present greedy algorithms for overlapping channel assignments for an AP-like setting. However, the existing work in overlapping channel assignment for wireless LANs is distinct from our work in various ways. Firstly, in wireless LANs, only one channel is assigned to one access point. Also, it is not trivial to extend their system model to multi-channel assignment scenario. Second, most of the previous works do not present any bounds on the proposed algorithms [10], [23], [24].

Graph theory: In the domain of graph theory, the problem of *multiple channel allocation* has been termed as problem of *multi-coloring* [16] where the objective is to assign multiple colors to all the nodes while ensuring that no two neighbors get the same color. *Partial coloring* has been studied before in graph theory where the requirement is to color only a subset of nodes. *Fractional coloring* [25] allows assignment of only a fraction of a color to a node. However, two neighbors can still not be assigned the same color. In the problem of *List coloring* [25], a node can be colored only from a list of colors applicable to that node. List coloring also does not allow same color to be assigned to neighbors. *Defective Coloring* [26] allows the same color to be assigned to a maximum of d neighboring nodes all of which get a color count of 1 from that color. On the other hand, upon sharing, split coloring assigns fractional color count to each neighbor. Thus, defective coloring does not exhibit the concept of *splitting* (or sharing) the color among neighbors which is the essential idea behind split coloring.

Link Scheduling in Wireless Networks: Link scheduling has been widely studied as part of the literature in wireless networks. However, the link scheduling problem involves finding the schedule of minimum duration while assigning a fixed number of time slots to each link. On the other hand, the channel allocation problem involves maximizing the number of channels assigned to each node (from a fixed set of channels) [27].

VII. Conclusion & Future Directions

Channel assignment in Cognitive Radio Networks under the sharing model leads to a new class of problems. We showed that the problem of fair channel assignment under this class is NP-Hard. We presented a heuristic algorithm for solving this problem. We also proposed a simple method for estimating the expected throughput of APs under the channel sharing model. The proposed algorithm has provable bounds in terms of both fairness and throughput. In order to reduce the number of reconfigurations in the network, we also proposed its localized version. Combining the centralized algorithm and its localized version, our complete solution is able to achieve high fairness, and high network throughput with few channel reconfigurations. Using simulations, we showed that sharing of color indeed helps in increasing the fairness of the system. Simulations also showed that the proposed throughput estimation model is able to capture the pattern in variation of throughput of APs with varying degree. Currently, we are implementing our algorithms on real hardware. We plan to do a detailed analysis of its performance in practical deployments. We would also like to see how the frequency of invocation of the centralized algorithm affects the overall quality of channel assignment.

References

[1] S. Bridge, "High Speed Broadband to Rural Areas," http://www.spectrumbridge.com/web/images/pdfs/whitespaces_casestudy-spectrumbridge.pdf.

[2] P. Bahl, R. Chandra, T. Moscibroda, R. Murty, and M. Welsh, "White space networking with wi-fi like connectivity," in *SIGCOMM*. ACM, 2009, pp. 27–38.

[3] I. Akyildiz, W. Lee, M. Vuran, and S. Mohanty, "NeXt Generation/Dynamic Spectrum Access/Cognitive Radio Wireless Networks: A Survey," *Computer Networks*, vol. 50, no. 13, pp. 2127–2159, 2006.

[4] R. Rajbanshi, Q. Chen, A. Wyglinski, G. Minden, and J. Evans, "Quantitative comparison of agile modulation techniques for cognitive radio transceivers," in *Proc. IEEE Consumer Commun. and Networking Conf.-Wksp. on Cognitive Radio Networks*.

[5] J. Poston and W. Horne, "Discontiguous OFDM considerations for dynamic spectrum access in idle TV channels."

[6] G. Bianchi *et al.*, "Performance analysis of the IEEE 802. 11 distributed coordination function," *IEEE JSAC*, vol. 18, no. 3, pp. 535–547, 2000.

[7] S. Xu and T. Saadawi, "Does the IEEE 802. 11 MAC protocol work well in multihop wireless ad hoc networks?" *IEEE communications Magazine*, vol. 39, no. 6, pp. 130–137, 2001.

[8] S. Shah, K. Chen, K. Nahrstedt *et al.*, "Available bandwidth estimation in IEEE 802.11-based wireless networks," in *The 1st Bandwidth Estimation Workshop (BEst 2003)*, 2003.

[9] R. Prasad, C. Dovrolis, M. Murray, and K. Claffy, "Bandwidth estimation: metrics, measurement techniques, and tools," *IEEE network*, vol. 17, no. 6, pp. 27–35, 2003.

[10] E. Villegas, R. Ferré, and J. Paradells, "Frequency assignments in IEEE 802.11 WLANs with efficient spectrum sharing," *Wireless Communications and Mobile Computing*, vol. 9, no. 8, pp. 1125–1140, 2009.

[11] C. Raman, R. Yates, and N. Mandayam, "Scheduling Variable Rate Links via a Spectrum Server," in *IEEE DySPAN*, 2005, pp. 110–118.

[12] S. Zekavat and X. Li, "User-Central Wireless System: Ultimate Dynamic Channel Allocation," in *DYSPAN*, 2005, pp. 82–87.

[13] K. N. Ramachandran, E. M. Belding-Royer, K. C. Almeroth, and M. M. Buddhikot, "Interference-aware channel assignment in multi-radio wireless mesh networks," in *INFOCOM*, 2006.

[14] S. Gandhi, C. Buragohain, L. Cao, H. Zheng, and S. Suri, "A General Framework for Wireless Spectrum Auctions," in *IEEE DySPAN*, 2007.

[15] M. Iridon, D. W. Matula, and C. Yang, "A graph theoretic approach for channel assignment in cellular networks," *Wireless Networks*, vol. 7, no. 6, pp. 567–574, 2001.

[16] L. Narayanan and S. M. Shende, "Static frequency assignment in cellular networks," *Algorithmica*, vol. 29, no. 3, pp. 396–409, 2001.

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

[18] A. Raniwala and T. Chiueh, "Architecture and Algorithms for an IEEE 802.11-Based Multi-Channel Wireless Mesh Network," in *INFOCOM*, Mar. 2005.

[19] M. Alicherry, R. Bhatia, and E. L. Li, "Joint channel assignment and routing for throughput optimization in multiradio wireless mesh networks," *IEEE JSAC*, vol. 24, no. 11, pp. 1960–1971, 2006.

[20] R. Chandra, R. Mahajan, T. Moscibroda, R. Raghavendra, and P. Bahl, "A case for adapting channel width in wireless networks," in *SIGCOMM*, 2008, pp. 135–146.

[21] S. Deb, V. Srinivasan, and R. Maheshwari, "Dynamic spectrum access in dtv whitespaces: design rules, architecture and algorithms," in *MOBICOM*, 2009, pp. 1–12.

[22] T. Moscibroada, R. Chandra, Y. Wu, S. Sengupta, P. Bahl, and Y. Yuan, "Load-Aware Spectrum Distribution in Wireless LANs," in *Proceedings of the 16th International Conference on Network Protocols*, 2008.

[23] A. Mishra, S. Banerjee, and W. Arbaugh, "Weighted coloring based channel assignment for WLANs," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, no. 3, p. 31, 2005.

[24] A. Mishra, V. Shrivastava, S. Banerjee, and W. Arbaugh, "Partially overlapped channels not considered harmful," in *Proceedings of the joint international conference on Measurement and modeling of computer systems*. ACM, 2006, p. 74.

[25] M. Kubale, *Graph Colorings*. American Mathematical Society, 2004.

[26] L. Cowen, W. Goddard, and C. Jesurum, "Defective coloring revisited," *J. Graph Theory*, vol. 24, pp. 205–219, 1995.

[27] Y. Cao and V. O. K. Li, "Scheduling Algorithms in Broad-band Wireless Networks," *PROCEEDINGS-IEEE*, vol. 89, no. 1, pp. 76–87, 2001.

Appendix

Proof of Lemma 3.2: We show the polynomial time reduction that transforms the split coloring to a proper coloring. The reduction satisfies the following invariants at all times:

1) If a node r_i is assigned only one color, then no neighbor of r_i is assigned that color.
2) If a color c_j is assigned to only r_i and to none of the neighbors of r_i, then c_j is the only color assigned to r_i.
3) The total aggregate throughput of all nodes is at least N.

Observe that initially every node has throughput of at least 1 unit. Therefore, initially if a node is assigned only one color, then the color must not be assigned to any of its neighbors as that will make its throughput less than 1. Hence, the first and third invariants must be true initially. Also, the second invariant can be trivially satisified initially by removing extra colors from the assignment set of nodes which have been exclusively assigned some color. This operation will still keep first and third invariants as true.

Algorithm 2 reduces the split coloring to proper coloring. It picks the node (say r_i) with minimum degree that is sharing some colors with its neighbors (line 4-6). The shared color is then removed from the assignment set of its neighbors (Line 7). This decreases the available throughput of its neighbors but at the same time it increases the throughput of r_i. Note that since all clients associated with an AP (say r_i) can hear interference from all neighbors of r_i, therefore if r_i is sharing a color with x neighbors, then the effective throughput of r_i over this color would be at most $\frac{1}{1+x}$.

Algorithm 2: Algorithm for reducing split coloring to proper coloring

```
1 Input: Graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ and its split coloring
2 Output: Proper coloring of $\mathcal{G}$
3 while there exists at least one color assigned to two neighbors do
4     Select a color $c_j$ that is still assigned to two neighboring nodes
5     Construct subgraph $\mathcal{G}_j$ that has only those nodes of $\mathcal{G}$ that are
      assigned $c_j$ and at least one other color. Only those edges from $\mathcal{G}$
      are included which are between these nodes.
6     Select a node (say $r_i$) that has minimum degree in $\mathcal{G}_j$ (say
      $deg(r_i)$).
7     $\mathcal{A}(r_k) \leftarrow \mathcal{A}(r_k)\backslash\{c_j\}\forall r_k \in N(r_i)$, $\mathcal{A}(r_i) \leftarrow \{c_j\}$
```

We show that invariants are maintained after every iteration of the *while* loop. Clearly, the first and second invariants are maintained (Line 7). We now show that the third variant is also maintained. Observe that as a color is deleted from the assignment set of neighbors of r_i, the throughput of r_i increases by at least $1 - \frac{1}{deg(r_i)+1}$. This is because initially r_i was sharing c_j with $deg(r_i)$ other neighbors and later r_i is not sharing that color with any neighbor. Also, this operation decreases the throughput of each of its neighbors by at most $\frac{1}{deg(r_i)+1}$. Again, this is because they were sharing c_j with at least $deg(r_i)$ neighbors. Therefore, the total loss in throughput over all the neighbors of r_i is at most $\frac{deg(r_i)}{deg(r_i)+1}$. Clearly, the decrease in throughput of the system is not more than than the increase. Therefore, the new throughput is at least as much as the throughput of the system before this rearrangement.

We repeat this procedure while there exists no color which is assigned to two neighboring nodes. We now show that when algorithm 2 terminates, then all nodes have exactly one color assigned to them. By contradiction, assume that when it terminates, there are x nodes with no color left in their assignment set. Then, all other $N-x$ nodes must have exactly one color assigned to them since there are no nodes left in the system that have more than one color assigned (due to invariant 2). This implies that system throughput at that moment must be $N-x$. Recall that the system throughput at the termination of the algorithm must be at least N as per the third invariant. Therefore, $x = 0$. Hence, when the procedure stops, all N nodes of the system will be assigned exactly one color that would form the desired proper coloring of the graph. ■

Proof of Lemma 4.1: First, we claim that during the execution of Phase 1, the following invariant is always true: $x_{ki} \leq (1 + \mathcal{T}(\mathcal{A}_{Split}, r_i))$. We prove this by contradiction. Clearly, the invariant is initially true. Assume that the invariant was first violated when $Split$ assigned c_j to a neighboring AP of r_i (say r_k). Before r_k was assigned color c_j, its available throughput was at least $x_{ki}-1$ and was therefore, greater than the throughput of r_i at that moment. Also, from the definition of x_{ki}, we know that r_k is the first neighbor of r_i that was assigned c_j. Therefore, in that case, c_j would not have been assigned to r_k since its neighbor r_i had lesser throughput (Line 10). Hence, the invariant must hold at all the times.

Next, we claim that at the end of Phase 1, the following is true: $\mathcal{T}(\mathcal{A}_{Split}, r_i) + \sum_{k \in N(r_i)} x_{ki} \geq |W(r_i)|$. This is true because every color in $W(r_i)$ must have been assigned to either r_i or a neighbor of r_i (Line 5-7). Therefore,

$$\begin{aligned} \mathcal{T}(\mathcal{A}_{Split}, r_i) + \Delta x_{ki} &\geq \mathcal{T}(\mathcal{A}_{Split}, r_i) + \sum_{k \in N(r_i)} x_{ki} \\ \mathcal{T}(\mathcal{A}_{Split}, r_i) + \Delta x_{ki} &\geq |W(r_i)| \end{aligned}$$

Also, $x_{ki} \leq (1 + \mathcal{T}(\mathcal{A}_{Split}, r_i))$. Substituting that in the above result, we get:

$$\begin{aligned} \mathcal{T}(\mathcal{A}_{Split}, r_i) + \Delta(1 + \mathcal{T}(\mathcal{A}_{Split}, r_i)) &\geq |W(r_i)| \\ \mathcal{T}(\mathcal{A}_{Split}, r_i) &\geq \frac{|W(r_i)| - \Delta}{1+\Delta} \end{aligned}$$

■

Proof of Lemma 4.2: If a node r_i has lesser throughput than all its neighbors, then all its principal colors would be still assigned to it at the end of Phase 3 (Line 22-23). Further, it would not be sharing any of its principal colors with its neighbors (Line 22-23). Therefore, its throughput would be at least as much as its throughput was at the end of Phase 1. From Lemma 4.1, we know that $\mathcal{T}(\mathcal{A}_{Split}, r_i) \geq \frac{|W(r_i)|-\Delta}{1+\Delta}$. ■

Proof of Theorem 4.1: Let r_i be the node that was assigned minimum throughput by $Split$. Since, r_i has the minimum throughput among all nodes in $Split$, therefore, we know from Lemma 4.2 that $\mathcal{T}(\mathcal{A}_{Split}, r_i) \geq \frac{|W(r_i)|-\Delta}{1+\Delta}$. Therefore, $\Upsilon_{Split} \geq \frac{|W(r_i)|-\Delta}{1+\Delta}$. Also, the size of its availability set is $|W(r_i)|$, therefore its maximum possible throughput is $|W(r_i)|$. Hence, $\Upsilon_{OPT} \leq \mathcal{T}(\mathcal{A}_{OPT}, r_i) \leq |W(r_i)|$ or $|W(r_i)| \geq \Upsilon_{OPT}$. We now know that $\Upsilon_{Split} \geq \frac{|W(r_i)|-\Delta}{1+\Delta}$ and $|W(r_i)| \geq \Upsilon_{OPT}$. Combining the two results, we can see that $\Upsilon_{Split} \geq \frac{\Upsilon_{OPT}-\Delta}{1+\Delta}$. ■

Proof of Lemma 4.3: There are two possibilities. First possibility is that $y > d + 3(1 + z(\Delta))$. In this case, it would be possible to conclude that $\chi(\mathcal{G}) > 3$. However, we know that a polynomial time algorithm $POLY$ cannot solve 3-coloring problem. Therefore, this case is not possible. The other case is when $y \leq d + 3(1 + z(\Delta))$. In this case, $\chi(\mathcal{G}) \leq d + 3(1 + z(\Delta))$ which implies $\chi = O(z(\Delta))$. Since, a polynomial time algorithm can derive an upper bound on chromatic number of a graph which is at best $O(\Delta)$ [25], therefore, we should have $z(\Delta) = \Omega(\Delta)$. ■

Proof of Lemma 4.4: We will use the result from Lemma 4.3 to prove this. Let an arbitrary graph $\mathcal{G}$ be given as input to $POLY$ whose chromatic number needs to be determined. Further, each node is assigned the same availability set. We also set the the effective neighbor count of each node as same as its degree. We start by assigning $\Delta + 1$ colors in each node's availability set and decrementing that number by 1, until $POLY$ is not able to assign at least 1 unit of throughput to every node. At that point, let the number of colors in each node's availability set be y. Therefore, two bounds can be obtained on the chromatic number ($\chi(\mathcal{G})$):

- $POLY$ was able to split color $\mathcal{G}$ with $y + 1$ colors. Therefore, from Lemma 3.2, $\chi(\mathcal{G}) \leq y + 1$.
- $POLY$ was not able to color $\mathcal{G}$ with y colors. Therefore, the guarantee provided by $POLY$ implies that OPT will give a max-min of at most $f(\Delta)$ on $\mathcal{G}$ with y colors. Clearly with $\frac{y}{1+f(\Delta)}$ colors, the OPT will not be able to split color $\mathcal{G}$ as well. Using Lemma 3.1, we can claim that with $\frac{y}{1+f(\Delta)}$ colors, it is impossible to properly color $\mathcal{G}$. This implies that $\chi(\mathcal{G}) \geq \frac{y}{1+f(\Delta)}$

From the above two statements, we can conclude that $\frac{y}{1+f(\Delta)} \leq \chi(\mathcal{G}) \leq y$. However, from Lemma 4.3, we know that this is possible only if $f(\Delta) = \Omega(\Delta)$. ■

Proof of Lemma 4.5: As in Lemma 4.4, we will prove this Lemma using the result from Lemma 4.3. The given arbitrary graph (say $\mathcal{G}$) whose chromatic number needs to be determined is provided as input to $POLY$. Further, each node is assigned the same availability set. We also set the the effective neighbor count of each node as same as its degree.

We start by assigning only 1 color to each node's availability set and incrementing that number by 1, stopping as soon as $POLY$ is able to assign at least 1 unit of throughput to every node. At that point, let the number of colors in the node's availability set be y. Clearly, with ny colors, $POLY$ will be able to assign n units of throughput to each node. We now start decrementing the number of colors in the availability set of each node and stopping as soon as $POLY$ reports the first term of sequence as exactly n. Let $ny - h(n)$ be the number of colors in the availability set of each node at that time. With $ny - h(n)$ colors, $POLY$ reports max-min as n. Therefore, using the above inequality, for the same number of colors, OPT will report max-min as at most $O(\Delta) + ng(\Delta)$. Also, it can be seen that OPT will not be able to split color $\mathcal{G}$ with $\frac{ny-h(n)}{O(\Delta)+ng(\Delta)+1}$ colors. Therefore, using Lemma 3.1, we can claim that $\chi(\mathcal{G}) \geq \frac{ny-h(n)}{O(\Delta)+ng(\Delta)+1}$. Now, there are two possibilities:

- $h(n) = \omega(n)$. In this case, $\lim_{n\to+\infty}(ny - h(n)) < 0$. This implies that it is possible to split color G with 0 colors and that cannot be true.
- $h(n) = \Theta(n)$. In this case, we repeat the procedure with very large value of n. Therefore, $\chi(\mathcal{G}) \geq \frac{ny-h(n)}{O(\Delta)+ng(\Delta)+1}$ will reduce to $\chi(\mathcal{G}) \geq \frac{y-d}{g(\Delta)}$ where d is some constant.
- $h(n) = o(n)$. We repeat the procedure with very large value of n. Therefore, $\chi(\mathcal{G}) \geq \frac{ny-h(n)}{O(\Delta)+ng(\Delta)+1}$ will reduce to $\chi(\mathcal{G}) \geq \frac{y}{g(\Delta)}$.

In all the cases possible, it is possible to claim that $\chi(\mathcal{G}) \geq \frac{y-d}{g(\Delta)}$ where d is some constant. Using Lemma 3.2 and our observation, we can also deduce that $\chi(\mathcal{G}) \leq y$. Using both the claims, we can say that $\frac{y-d}{g(\Delta)} \leq \chi(\mathcal{G}) \leq y$. Again using Lemma 4.3, we know that this is possible only if $g(\Delta) = \Omega(\Delta)$. ■

Lemma A.1: For every channel c_j, the system throughput guaranteed by $Split$ is orderwise at most Δ times smaller than the system throughput given by OPT for that channel.

Proof of Lemma A.1: In Phase 1, $Split$ assigns c_j to a maximal independent set of nodes. Also, by the time the algorithm terminates, it ensures that nodes that have c_j assigned form a maximal independent set with some of their neighbors included. The OPT algorithm may have assigned c_j to a maximum of N nodes resulting in the system throughput of at most N. We know that a maximal independent set of a graph is at most Δ times smaller than the number of nodes in the graph. Therefore, the number of nodes that were assigned c_j by $Split$ is at most Δ smaller than the throughput achieved by OPT on c_j. Also, by sharing the color in $Split$, this set of nodes could have reduced their throughput by at most a constant factor (as discussed in section II). Therefore, the system throughput guaranteed by $Split$ for color c_j is orderwise at most Δ times smaller than throughput guaranteed by OPT. ■

Proof of theorem 4.3: Directly follows by applying Lemma A.1 over all colors and then taking the summation. ■