Bipartite Graph Based Dynamic Spectrum Allocation for Wireless Mesh Networks

Jianjun Yang, Zongming Fei
The Laboratory for Advanced Networking
Department of Computer Science, University of Kentucky
Lexington, KY 40506-0046, USA

Abstract—The capacity of a wireless mesh network can be improved by equipping mesh nodes with multi-radios tuned to non-overlapping channels. By letting these nodes utilize the available spectrum opportunistically, we can increase the utilization of the available bandwidth in the spectrum space. The key problem is how to allocate the spectrum to these multi-radio nodes, especially when they are heterogeneous with diverse transmission types and bandwidth. Most of current work has been based on the conflict-graph model and given solutions that focused on either increasing bandwidth utilization or minimizing starvation. In this paper, we propose a new bipartite-graph based model and design an channel allocation algorithm that considers both bandwidth utilization and starvation problems. Our solution is based on using augmenting path to find a matching in the bipartite-graph and can minimize starvation and then maximize the bandwidth utilization. The simulations demonstrate that our algorithm can reduce the starvation ratio and improve the bandwidth utilization, compared with previous conflict-graph based algorithms.

I. INTRODUCTION

There is a common belief that we are running out of usable radio spectrum. However, the Federal Communication Commission reported a different story that at a given time and location, much of the precious spectrum lies idle [1]. It has been pointed out that in many situations, spectrum access is a more significant problem than physical scarcity of spectrum, in large part due to legacy command-and-control regulation that limits the ability of potential spectrum users to obtain such access [2]. This indicates that spectrum shortage results from the spectrum management rather than the physical scarcity of usable frequencies.

To manage the spectrum properly, we need to apply appropriate strategies to allocate the existing spectrum. A basic requirement is to avoid interference in that different links or users cannot use the same channel within their transmission range at the same time. Static spectrum allocation algorithms allocate fixed spectrum slices to each user. This can prevent interference but results in poor utilization and spectrum hole [2]. To solve the problem, various dynamic assignment solutions were proposed to allocate spectrum among heterogeneous users with diverse transmission types and bandwidth. The users can sense their available spectrum and utilize them opportunistically. This becomes possible because lower layer technical innovations equip nodes in wireless mesh network with multi-radios and enable them to access different channels at different locations and time. The dynamic allocation can be categorized into two types [3]. One is the hierarchical access model, in which users are divided into primary users and secondary users. The channels are assigned to the primary users first. The secondary users can use them only if the channels are released by primary users at certain time slots or they are free. The other is dynamic exclusive use model, in which the channels are allocated to a user for exclusive use at a certain time and may be re-allocated to a different user later. The objective is to improve the efficiency of using spectrum through flexible allocation of channels.

Most of existing work [4], [5], [9] has been based on the conflict-graph model, in which a vertex represents a user and an edge connects two users with conflict. The problem with the model is that one edge can represent the interference relationship only between a pair of users. If $k$ users interfere with each other over one channel, we need to have $\frac{k(k-1)}{2}$ edges. This makes the graph complicated. More importantly, the edge can only represent the interference relationship but nothing else such as channel bandwidth, which is essential for spectrum allocation considering bandwidth utilization. Therefore, it is hard to use graph-theory based algorithms to achieve the objectives because of lack of weights on the edges. Another problem with current solutions is that they focused on either increasing bandwidth utilization or minimizing starvation, but not both.

In this paper, we propose a new bipartite-graph based model and design a channel allocation algorithm that considers both bandwidth utilization and starvation problems. Our solution is based on using augmenting path to find a matching in the bipartite-graph and can minimize starvation and then maximize the bandwidth utilization. The simulations demonstrate that our algorithm can reduce the starvation ratio and improve the bandwidth utilization, compared with previous conflict-graph based algorithms.
The rest of this paper is organized as follows. After reviewing related work in Section II, we describe our system model in Section III. We then present our spectrum allocation mechanism in Section IV and performance evaluation in Section V. Section VI concludes the paper.

II. RELATED WORK

Most dynamic channel allocation mechanisms use heuristic algorithms to achieve the goal of increasing the bandwidth utilization. Zheng and Peng proposed a greedy algorithm for dynamic spectrum allocation [4]. In each step, the algorithm picks the vertex with the highest bandwidth and assigns the channel to its associated user. Then it cuts the edges that interfere with this user. It repeats these two steps until all the channels are allocated. This algorithm can reach near-optimal utilization without considering any other constraints. The problem with the algorithm is that it may cause starvation for some users.

Another approach [4] is to pick up the vertex with the highest label, which is defined as the bandwidth divided by the number of users interfering with each other over this channel. This approach tries to maximize the utilization and minimize the interference. However, it cannot allocate the spectrum to maximal number of users.

Marina and Das [5] proposed a centralized greedy heuristic algorithm called CLICA for spectrum allocation. They use the node's degree of flexibility as a guide in determining the order of coloring decisions. Each node is associated with a priority. It colors the node with the lowest priority and each of its adjacent nodes and update the graph until all the node are colored. This algorithm can achieve minimal interference and maintain a topology in a network. However, it does not consider the total bandwidth utilization.

III. SYSTEM MODEL

We assume that each user is configured with multiple interfaces and each interface can use a different channel. The channels differ from each other in bandwidth and transmission range and are orthogonal. The channels within a user’s range are available to the user. A channel might be available to multiple users, but it can only be allocated to one of them if they are within transmission range of each other, otherwise they will conflict. During a certain period of time, users are competing for available channels. A spectrum allocation wants to achieve the following objectives: (i) the number of users with allocated channels is maximal; and/or (ii) the sum of the allocated channel bandwidth is maximal.

We use a Bipartite graph $G = (V, E)$ to model the conflicts and available bandwidth among different users. In our model, the vertex set is composed of elements from two subsets, the user set $U$ and the channel set $C$. That is, $V = U \cup C$ and $U \cap C = \emptyset$. Edge $e \in E$ is in the form of $(u, c)$ where $u \in U$ and $c \in C$. Edge $e = (u, c)$ means that channel $c$ is available to $u$. For each user vertex $u \in U$, there is at least one edge connecting it. Otherwise, we can remove the node from the graph. The same is true for channel vertexes. We can further define a weight function $W : E \rightarrow \mathbb{R}^+$ over the edge set $E$. The weight $W(e)$ of edge $e = (u, c) \in E$ is the bandwidth that node $u$ can get if it uses channel $c$.

![Fig. 1. An initial bipartite graph for representing conflicts](image)

Generally, if multiple user vertexes connect with the same channel vertex, they will conflict with each other. However, this depends on how the set of channel vertexes is defined. Fig. 1 illustrates a case where users $u_1$, $u_2$, $u_3$, and $u_4$ can all possibly use the same frequency represented by channel $c_1$. Assume that $u_1$ and $u_2$ are close to each other and $u_3$ and $u_4$ are close to each other, but $u_1$, $u_2$ are far away from $u_3$, $u_4$. So $u_1$ will interfere with $u_2$ while $u_3$ will interfere with $u_4$, but $u_1$, $u_2$ will not interfere with $u_3$, $u_4$ with regard to channel $c_1$. In this case, we can split $c_1$ into two channels, $c_{1,1}$ and $c_{1,2}$, to represent the channel in different locations. We then let $u_1$ and $u_2$ connect with $c_{1,1}$ and $u_3$ and $u_4$ connect with $c_{1,2}$, as shown in Fig. 2. With this simplification, we can make sure that if two user vertexes connect with the same channel vertex, they will interfere with each other. For the rest of this paper, we will assume that this split has always been done.

The spectrum allocation problem is to find a subgraph $G' = (V, E')$, where $E' \subseteq E$, such that for any $c \in C$, there exists only one $u \in U$, such that $(u, c) \in E'$. However, it is fine for a $u \in U$, there are multiple edges connecting with it. The maximal bandwidth utilization problem can be defined as finding $G' = (V, E')$ such that $\sum_{e \in E'} W(e)$ is maximized.

The goal of minimizing starvation is to have as many users as possible being allocated with some channels.
We can define $U' = \{ u \in U : \exists c \in C, (u, c) \in E' \}$. The minimal starvation allocation problem can be defined as finding $G' = (V, E')$, such that $|U'|$ is maximized.

This model represents the availability and interference directly and concisely. For instance, when there are $k$ users who interfere with each other with regard to a channel, there are only $k$ interference edges. More importantly, the weight of an edge represents the bandwidth of the channel. Hence, we can use graph-theory based approaches to solve the problem.

IV. BIPARTITE-MATCHING BASED SPECTRUM ALLOCATION MECHANISMS

In this section, we propose to use the bipartite graph matching algorithm to solve the spectrum allocation problem. The solution to the matching problem allocates at most one channel for each user. Our approach is to repeat the process multiple times so that a user can be allocated with multiple channels. We will start with a simple solution of using maximal cardinality matching problem to find the solution to the spectrum allocation problem for minimizing starvation. Based on that, we will then propose a solution to the maximal weight matching problem, which will be used as the basis for the final dynamic spectrum allocation algorithm that minimizes starvation and then maximizes bandwidth usage.

A. Maximal Cardinality Matching Problem

For a given Bipartite graph $G = (V, E)$, a matching $M$ is a subset of $E$ such that any two edges in $M$ are disjoint. A maximum cardinality matching is a matching that maximizes the number of edges. A naive algorithm to achieve maximum matching is to find out all the matching and select the matching with maximal edges. Its time complexity is exponential. We will present an alternative approach based on augmenting path [14].

Suppose $M$ is a matching of graph $G$. The vertexes adjacent to the edges in $M$ is said to be matched. If $P$ is a path connecting two unmatched vertexes in $G$ and the edges belonging to $M$ and not belonging to $M$ appear in $P$ alternately, then $P$ is an augmenting path based on $M$.

The augmenting path from $v_i$ to $v_j$ has three characteristics:

1) The number of hops in an augmenting path from $v_i$ to $v_j$ is an odd number.
2) Neither $v_i$ nor $v_j$ belongs to $M$.
3) A larger matching $M'$ can be obtained by $M$ and an augmenting path $P$ based on $M$. Let $M' = M \oplus P$. That is, the larger matching $M'$ includes the edges that either belong to $M$ or belong to $P$ but do not belong to both $M$ and $P$.

We present the solution to the maximal cardinality matching problem in Algorithm 1. Initially, it sets the largest edge matching as $\emptyset$, and divides the edges as matched and unmatched edges. Then from line 6 to line 22, it tries to find an augmenting path based on $M$ to increase the matching cardinality. The method to augment matching edges is to let $M = M \oplus P$. If no augmenting path can be found, which is determined by line 23, the algorithm stops and outputs the maximal cardinality matching.

![Fig. 2. A simplified bipartite graph for representing conflicts](image-url)
Fig. 3. A matching $M$ of graph $G$

For example, in Fig. 3 we show a matching $M$ represented by solid lines in graph $G$. We can use Algorithm 1 to find an augmenting path $P$ based on $M$, as shown in Fig. 4. By combining $M$ and $P$, we obtain a larger matching $M' = M \oplus P$ in Fig. 5.

Fig. 4. An augmenting path based on $M$

Fig. 5. The larger matching $M' = M \oplus P$

B. Maximal Weight Matching Problem

To achieve maximal weight matching, we would like
to use augmenting path approach to enlarge the matching gradually and reach our objective in the end. The problem is how to find an augmenting path $P$ based on a tentative matching $M$ with $k$ edges, and make $M' = M \oplus P$ the maximal weight matching with $k + 1$ edges. A brute-force search algorithm is to list all the possible matching with $k + 1$ edges and select

the maximal one. Its time complexity is exponential. We will make use of a modified Bellman-Ford algorithm to solve the problem in Algorithm 2.

Initially, it sets the largest weight matching as $\emptyset$ and divides the edges as matched and unmatched edges. Then from line 6 to line 19, it tries to find an augmenting path based on $M$ to enlarge the matching weight by the Bellman-Ford shortest path algorithm. The weight of unmatched edge is negative, and Bellman-Ford algorithm permits negative distance path to exist. Hence once the path is shorter, the weight is bigger. The method to augment matching edges is still to let $M' = M \oplus P$.

If no augmenting path can be found, the algorithm stops and outputs the maximal weight matching.

Fig. 6 illustrates the maximal weight matching with two edges. The weight of each edge is shown as a number beside it. The dark vertexes denote matched vertexes and the solid lines denote matched edges, while dash lines denote unmatched edges.

In Fig. 7, a tentative matching $M$ has two edges. There are multiple options for 3 edges matching. By algorithm 2, we found that augmenting path $P$ is $(u_4,s_2,u_2,s_3,u_3,s_5)$ because it is the shortest path with length -16. So the sum weight gained from this path is maximal. Let $M' = M \oplus P$, then $M'$ is $((u_4,s_2),(u_2,s_3),(u_3,s_5))$. It is the maximal weight match-

---

**Algorithm 2** MaxWeightMatching($G$)

1: $M \leftarrow \emptyset$
2: $E_{match} \leftarrow \emptyset$
3: $E_{unmatch} \leftarrow E$
4: $k \leftarrow$ number of match edges
5: AugExist $\leftarrow$ True
6: while AugExist do
7: for any edge $e \in E_{match}$ do
8: Let its direction be from $S$ to $U$
9: for any edge $e \in E_{unmatch}$ do
10: Let its direction be from $U$ to $S$
11: $W_e = -|W_e|$
12: Add vertex $S$ and $D$
13: for any left vertex $u_i$ in $E_{unmatch}$ do
14: Add edge $(S, u_i)$
15: for any right vertex $c_j$ in $E_{unmatch}$ do
16: Add edge $(c_j, D)$
17: From $S$ to $D$, find the shortest path $P$ by Bellman-Ford algorithm
18: $M_{pre} \leftarrow M$
19: $M \leftarrow M \oplus P$
20: Update $E_{match}, E_{unmatch}$ and $k$ according to the edges in $M$
21: if $M_{pre} == M$ then
22: AugExist $\leftarrow$ False
23: Output $M$
Algorithm 3 FinalMatching(G)

1: let $G' = (V', E')$
2: $V' \leftarrow V$
3: $E' \leftarrow \emptyset$
4: while $E \neq \emptyset$ do
5: $M = \text{MaxWeightMatching}(G)$
6: for any edge $e_i \in M$ do
7: $e_i \leftarrow e_i$'s right vertex
8: delete $c_i$ from $G$
9: delete edge adjacent to $c_i$ from $G$
10: $G' = G' \cup M$
11: Output $G'$

V. PERFORMANCE EVALUATION

We conduct our performance evaluation using a noiseless immobile radio network environment, where the nodes are distributed in a given area and may each have a different transmission range and bandwidth. Since what we are interested in is to compare our results with the outputs generated by other allocation approaches, we convert this network to a weight graph $G=(V,E)$ where the weight represents the bandwidth. We set the number of users to be 30 and let the number of channels vary from 25 to 50 with increment 5. We set the probability that an edge exists between any pair of user and channel nodes to be 0.2 and the edge weight is uniformly distributed from 1 to 9. For each configuration, we generate 10 graphs and conduct the experiment 10 times based on the graphs. We calculate the average value from the experiments as the result for each configuration.

We use two metrics to evaluate the performance. One is the sum bandwidth of all users by the allocation. The other is the allocation ratio of the number of users who are allocated with at least one channel over the number of users who are competing for the spectrum pool.

We compare our solution with three other approaches. The first one is NMSB (Non-collaborative-Max-Sum-Bandwidth). In each step, this approach picks the vertex with the highest bandwidth and assigns it to its associated user. Then the algorithm remove the edges that interfere with this user until all the channels are allocated. The second one is CMSB (Collaborative-Max-Sum-Bandwidth). This approach picks up the vertex with the highest label, defined as the bandwidth of a channel divided by the number of users interfering with each other with regard to this channel. The process is repeated until all channels have been allocated. The third approach is MINSTARVE, which tries to allocate the channels to the users who have not been allocated before. In this approach, each user has a priority. The user’s priority is decreased by one if the user is allocated with a channel. The algorithm takes care of higher priority users first.
In Fig. 8, we evaluate the sum bandwidth of these approaches. The number of channels changes from 25 to 50. The results show that our approach can achieve near optimal sum bandwidth, similar to NMSB. Both NMSB and our approach are about 10% to 25% higher than the other two approaches.

![Fig. 8. Sum bandwidth of allocated channels](image)

In Fig. 9, we evaluate the allocation ratio of the four approaches. It shows that our algorithm can reach near 100% allocation ratio. MINSTARVE is close to 80%, while the other two range from 40% to 70%. It illustrates that our mechanism can better avoid starvation.

![Fig. 9. Allocation ratios](image)

VI. CONCLUSION

In this paper, we present a bipartite graph based mechanism to assign channels to users who can opportunistically utilize its available spectrum. The objective is to consider both bandwidth utilization and starvation issues. We propose a bipartite-graph model and explore corresponding algorithms to dynamically allocate channels to competing users. The simulations demonstrate that our approach can result in significant performance benefits in the heterogeneous environment.

REFERENCES