Outdoor WiFi-Mesh channel Assignment Protocol

CSC-575 Project E8 Final Report

Umang Patel
Mohammad Al Srouji
Deepti Chinta
Gaurish Deuskar

1. Abstract

There is growing interest in wireless mesh networks due to their inherent properties that allow lower implementation costs as well as more resilience in providing wireless coverage. The IEEE 802.11 Wireless standards allow multiple non-overlapping frequency channels which provide mesh routers with the capability of communicating simultaneously while maintaining minimal amount of interference, even though they are within direct interference range of one another. However, in order to achieve that, a channel assignment protocol is needed to minimize interference between adjacent links so as to achieve an enhancement in the available bandwidth. In this paper, we implement a centralized static channel assignment protocol with a minimum hop count routing algorithm that is capable of minimizing the interference between adjacent links in the wireless mesh network while maintaining complete network connectivity. We have performed extensive experiments to be able to verify our implementation and analyze some of our observations. In our implementation we focused on minimizing the interference between the links inside the network itself and did not take into consideration other coinciding or adjacent networks. An overview of the underlying software and framework that was used will be discussed as well as the results of the throughput tests that were performed on the outdoor wireless mesh nodes.

2. Introduction

Wireless mesh networks can be used in a large array of diverse applications ranging from providing broadband internet access to establishing a mobile military communication framework. Their reliability, large bandwidth, and wireless nature provides suitability for being implemented in locations were short term wireless coverage is needed or wire installations are too expensive or undesirable. However, channel assignment for wireless mesh networks is crucial for their viability, since interference from adjacent links could cause a large drop in the actual available bandwidth.

It was our aim to create an outdoor wireless mesh network and measure the performance of our implementation of channel assignment protocol using an assortment of diverse topologies. The outdoor mesh router was created using a personal computer with four wireless NICs placed on a trolley, to provide maneuverability, with an antenna arrangement using PVC pipes, antennas, and antenna cables. Our contribution is not limited to the actual performance of the channel assignment protocol but extends to various observations and obstacles that were encountered during the execution of the experiments.

In the following sections we explore some of the related works concerning channel assignment for wireless mesh networks, and then provide an overview of the underlying software framework that was used as well as description of our channel assignment algorithm implementation. Finally we delve into the main part of this paper which revolves around presenting the actual experimental results and a detailed analysis of those results.

[1] MeshBed is an outdoor test-bed being deployed at North Carolina State University for Wireless Mesh Network Research.

3. Related Work

In the channel assignment domain, two types of channel assignments, static channel assignment (channels are assigned at the time of deployment and are kept fixed throughout) and dynamic channel assignment (the assignment of channels to different link changes over time) have been studied, implemented and evaluated. The benefit of dynamic assignment over the static one is that a good dynamic solution can lead to better capacity utilization and minimum interference of the mesh network but it comes with the cost of computational overhead and complexity along with the problem of network partition in which the new channel assignment could accidently sever the connectivity between different nodes of the network.

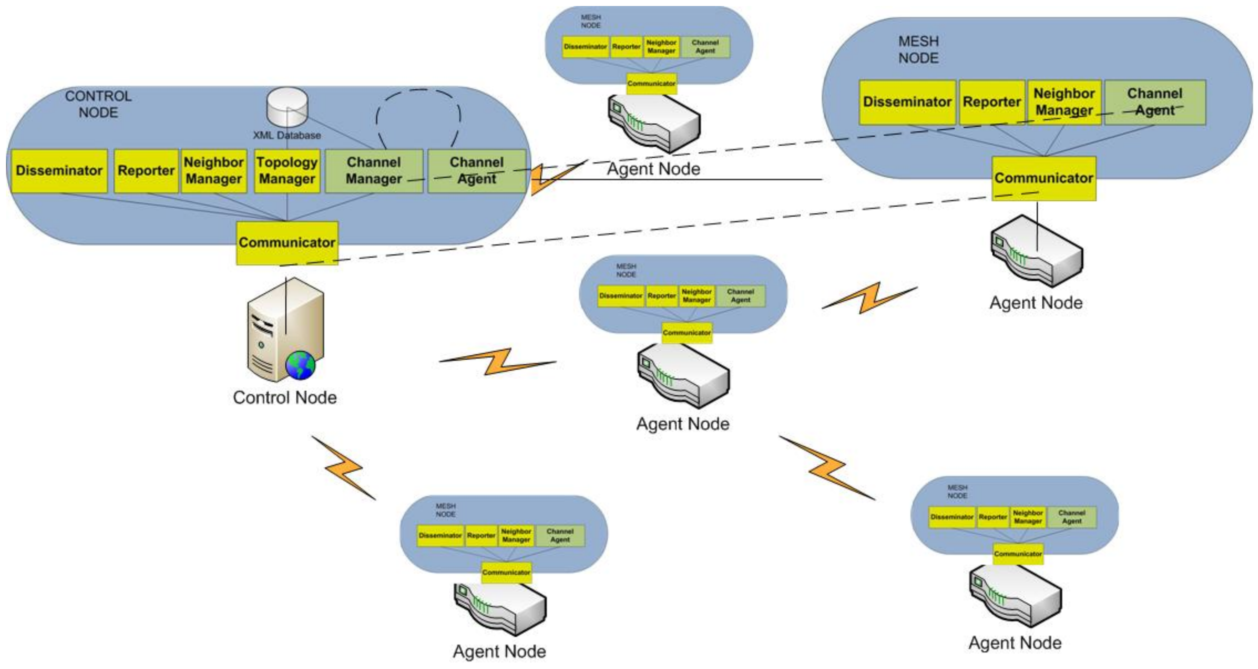
Paper [1] proposed a dynamic channel assignment scheme supporting the need for dynamic assignment over the static one. In this scheme, a gateway node (channel assignment server) periodically collects link state information (which includes interference levels on links) and neighborhood information (information regarding the neighbors) from each of the nodes in the mesh network and depending upon this, does the channel allocation by distributing it to all the nodes. In order to avoid the network partitioning problem, a default channel which has the minimum interference level of all the channels is used and at least one radio on each of the mesh routers is tuned to this channel. This enables connectivity even in case of node failures. They have given an interference estimating technique in which each node measures the level of interference for each channel. The level of interference of a channel is dependent upon the number of interfering radios on that channel and the traffic on that channel. They also propose a novel multi-radio conflict graph model to model the interference in the mesh network. This model is also adopted by us in our algorithm. Please refer to our algorithm for this model. The channel assignment algorithm used by [1] is a breadth first search one. In this, the channel assignment starts from the gateway node in a BFS manner and proceeds towards the edge nodes. This way, the links closer to the gateway get higher priority.

Paper [2] States the drawbacks of a dynamic channel assignment scheme including the synchronization precision required and the delay associated during the switching of channels and proposes a static channel assignment scheme. In this scheme, the authors propose the concept of clusters of nodes wherein the clusters are connected with the help of backbone network. Thus a node can be situated on a backbone or can be a regular one. The channel assignment is done on the cluster basis. It considers 2 factors which doing channel assignment: network connectivity and network throughput.

We have derived some of the ideas from [1] to propose our static channel assignment algorithm. Ideas like Multi-radio conflict graph and channel assignment starting from the gateway node have been taken from [1]. The channel assignment algorithm is basically a prioritized graph coloring algorithm with links closer to the gateway node having higher relative priority to the ones farther away. We have chosen a static assignment scheme as opposed to the dynamic one due to time constrains and limited domain knowledge. [3] has made a good survey and has references to assignment schemes considering multiple interfaces with modifications in the MAC layer. We have not considered any idea corresponding to any modification in the MAC layer.

4. Wireless MeshBed[1] Software framework

In this section, we give an overview of the wireless MeshBed control plane software framework. The Channel Assignment software uses part of the functionalities provided by MeshBed control plane software to perform channel assignment on the network nodes that is why a general idea of the control plane is provided.



The above figure depicts one sample wireless mesh network and the software components of interest in the network nodes. One node in the network acts as the control node while other nodes in the network act as agent nodes. One wireless interface on each node is used only for control plane message exchanges while other wireless interfaces are used for data traffic. Also it is assumed that each node has the same number of wireless interfaces available. Following is a brief description on the functionalities provided by each module of control plane software. Please refer to [7] for more detail.

- **Communicator:** The communicator provides flexible and reliable communication between processes running on the same or different nodes. It is messaging system through which all the distributed processes can interact in a loosely coupled manner. It replaces client/server model with publish/subscribe relationship between individual processes, where a process sends or receives messages based on topics it is interested in. It provides a set of interfaces for application development and reduces the complexity of communication by hiding the intricacies involved in the underlying message processing. It runs on every node of the network.
- **Disseminator and Reporter:** These processes work together and they provide transparent and reliable transfer of messages to all network nodes even if no routing paths are established. Before routing paths are established, they deliver messages throughout the network via flooding. If routing paths are established then they are capable of using established paths to reach a specific node instead of flooding. They run on every node of the network.

- **Neighbor Manager:** This module collects neighbor information concerning single hop neighbors and makes this information available to other modules. It runs on every node of the network.
- **Topology Manager:** Topology manager runs on the control node and builds network topology by using the neighbor information provided by neighbor manager running on each node. It stores the topology information into a BDBXML Database that can be read by other module (example: routing or channel assignment etc.)

5. Channel Assignment Software Overview

The channel assignment software consists of Channel Manager and Channel Agent modules as depicted in the figure from the previous section. Following is the description of the modules.

- **Channel Manager:** Channel Manager runs on the control node. It utilizes services of Communicator, disseminator, as well as reporter to send/receive messages to and from channel agents running on network nodes. It performs the following sequence of operation once it initiated.
  - It reads the network topology generated by Topology Manager from the XML Database.
  - It performs Dijkstra's shortest path algorithm based on hop count on the network topology and builds a tree structure that connects control node to every other node of the network.
  - It then performs channel assignment for the links of the tree based on the algorithm mentioned in Appendix to minimize internal interference.
  - For each agent node, it builds the routing table entries and channel assignment of one or more wireless interfaces and sends them in separate messages using disseminator. The message format is as follows.

Field	**Size (in bytes)**
Channel Number	4
Iteration Number	4
Manager IP Address	16
Agent IP Address	16
Destination IP Address	16
Next-hop IP Address	16

  - Agent node switches its channel on the wireless interface that has the address as specified in the Agent IP Address field of the message (Third octet of IP address identifies Agent node and fourth octet of IP address identifies wireless interface on that particular agent node). Channel Manager sends new iteration number for each run to indicate to the Channel Agents that they need to perform routing flushing to remove their routing table entries and establish new routing table entries and channel assignment as sent in the messages of the new iteration. The Manager IP Address field is used by Channel Agents to send acknowledgement messages using the Reporter. Destination IP Address and Next-hop IP Address are used to set the routing entry. Channel Manager sends multiple messages to a Channel Agent if it requires setting multiple routing table entries.
  - After sending a message, Channel Manager waits for acknowledgement message to come from Channel Agent. It reattempts sending the message multiple times before moving to next message if no acknowledgement for that particular message is received.
  - It terminates once all messages are sent.

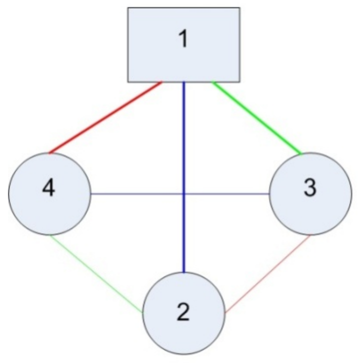
- **Channel Agent:** Channel Agent runs on all the nodes in the network. It utilizes services of Communicator and reporter to send/receive messages to and from the Channel Manager. It performs the following sequence of operations as soon as it starts.
  - It assigns IP addresses to the data plane wireless interfaces based on node id (example: if node id is 3 and it has three wireless interfaces then 10.0.3.2, 10.0.3.3, 10.0.3.4 are assigned) and puts them on the same 802.11 Ad-hoc BSS (Basic Service Set).
  - It waits for messages from Channel Manager that has matching Agent IP field.
  - For each matching message, it sets the channel and routing table entry based on the received message and sends acknowledgement message back to Channel Manager using Reporter.
  - If the matching message contains a new iteration number then it flushes existing routing table and establishes new routing table entries and channel assignment as sent in the messages of that hold the new iteration number.

6. Experimental results

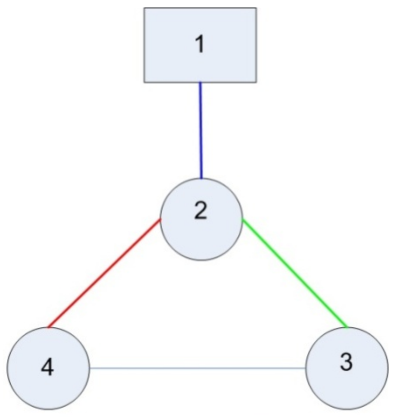
6.1 Experiments performed on actual mesh nodes

We had 4 mesh nodes each having 4 wireless mini-PCI cards inserted into a mini-PCI <-> PCI adapter. We worked on the 802.11g technology throughout. One of the nodes was taken as a gateway. The channel-assignment manager was run on the gateway whereas the agent was run on the other nodes.

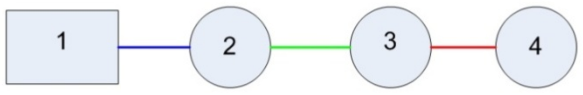
The following diagrams show the topologies and the channel assignment done by our algorithm. The square node is the gateway node whereas the round nodes are the other mesh nodes. The channel assignment manager is run on the gateway node. In the topology, 3 colors have been used viz. blue, green and red indicating channels 1, 6 and 11 respectively. A link between 2 nodes represents that the 2 nodes are neighbors of each other. In all the topologies we have considered, we have assumed symmetrical links (i.e the 2 nodes can communicate with each other if they have a link between them). The bold links indicate the presence of routes whereas the think links represents just the channel assignment. The routes as given in our algorithm are calculated using dijkstra algorithm from the gateway node. Thus in topology 1, the bold blue link 1-2 indicates that nodes 1 and 2 are connected and one radio on each of the nodes is tuned to channel 1. Also 1-2, 1-3 and 1-3 are the routes in the topology and the routing tables are set accordingly. Thin links 2-3 and 2-4 indicate that no actual link exists between the corresponding nodes but should a link be formed between the nodes by a routing algorithm, the will be assigned channels 6 and 11 respectively. For topology 2, the bold links indicate that the routes are 1-2-3 and 1-2-4 respectively (which means that node 2 is the next hop node for nodes 3 and 4 towards the gateway node 1 and vice-versa). The following figures show topology 1 (mesh topology), topology 2 and topology 3 (linear topology).



Topology 1



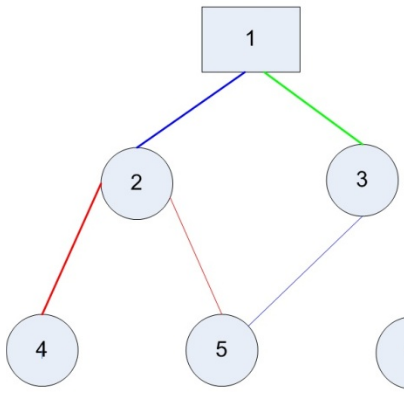
Topology 2



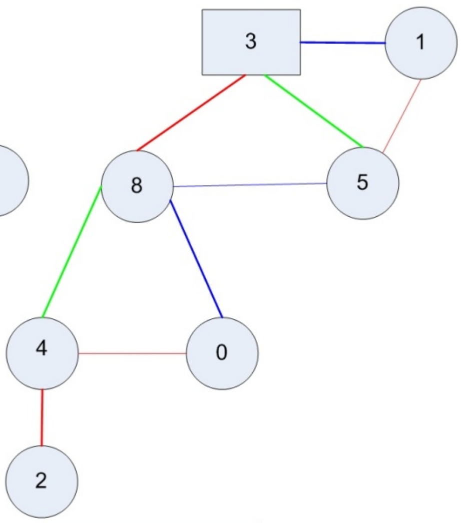
Topology 3

6.2 Experiments run on graphs

In these set of experiments, a virtual graph topology was considered and the channel assignment algorithm was run on the virtual graphs. . These experiments were basically performed to judge the behavior of the algorithm for different topologies. The following 2 figures show the topologies and their routing and channel assignments.



Topology 4



Topology 5

6.3 Throughput experiments with different antenna configurations

Once the channel assignment was done on the topology, UDP iperf flows were generated from the client mesh nodes to the gateway node and the throughput was observed. Testing was performed in 2 parts as follows:

Part 1: Testing with no antenna spacing.
In this set, the antennas were directly connected to the pigtails emanating from the desktop backs of each mesh node thereby having no space between the antennas. The results are shown for topologies 1 and 3. Results taken for topology 2 are not shown because of redundancy.

Details:
Flow 1: UDP flow from node 2 to node 1
Flow 2: UDP flow from node 3 to node 1
Flow 3: UDP flow from node 4 to node 1

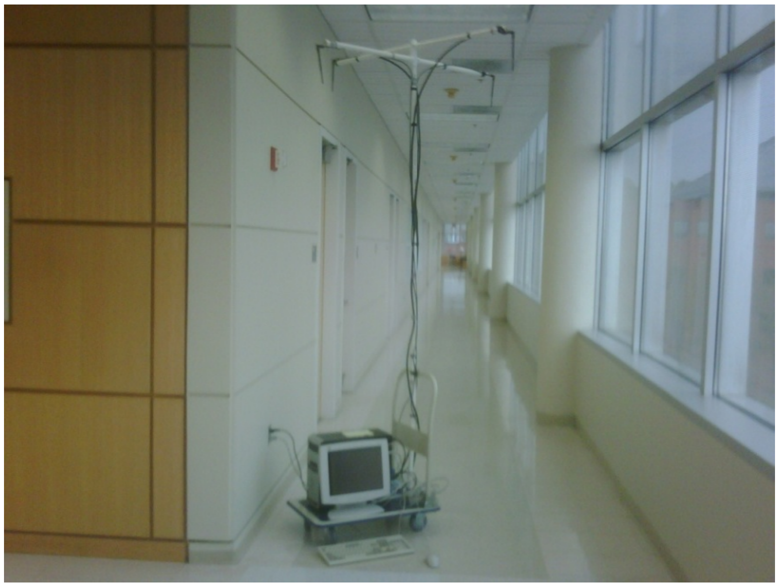
Table1.1 (Topology 1)

Test	Flow	Throughput
1	3	770 Kbps
2	2	22.7 Mbps
3	3	16.2 Mbps
4	1 + 2	Flow 1: 8.91 Mbps Flow 2: 2.5 Mbps
5	1 + 2 + 3	Flow 1: 2.12 Mbps Flow 2: 2.58 Mbps Flow 3: 6.7 Mbps

Table 1.2 (Topology 3)

Test	Flow	Throughput
1	3	770 Kbps
2	2 + 3	Flow 2: 1.26 Mbps Flow 3: 16 Kbps
3	1 + 3	Flow 1: 1 Mbps Flow 2: 773 Kbps
4	1 + 2 + 3	Flow 1: 1 Mbps Flow 2: 736 Kbps Flow 3: 0 Kbps

Part 2: Throughput tests with antenna spacing
In this set of tests, a "Christmas tree" framework of antennas was formed. The antennas were connected to the pigtail output through 9 feet antenna cables. The following picture shows the arrangement. In this set, only 2 nodes were considered. The channel assignment algorithm was not performed on this set up, instead, flows on different channels were generated manually. The main objective of this set of tests was to find whether antenna spacing yields any improvement in throughput compared to antenna placement with no spacing.



Details:
Flow 1: Flow between 2 radios of the 2 nodes on channel 1
Flow 2: Flow between 2 radios of the 2 nodes on channel 6
Flow 3: Flow between 2 radios of the 2 nodes on channel 11
Iperf UDP testing was done with client buffer size 50M

Table 2:

Test	Flow	Throughput
1	1	27 Mbps
2	2	22.6 Mbps
3	3	28 Mbps
4	1 + 2	Flow 1: 14 Mbps Flow 2: 8 Mps
5	1 + 3(with flow 3 now between positions B of both the sides instead of position C)	Flow 1: 22 Mps Flow 3: 16 Mbps
6	1 + 3	Flow 1: 27 Mbps Flow 2: 27 Mbps
7	1a(this flow between positions A) + 1b (this flow between positions B) + 3	Flow 1a: 14 Mbps Flow 1b: 11 Mbps Flow 3: 17 Mbps
8	1 + 2 +3	Flow 1: 18 Mbps Flow 2: 3.63 Mbps Flow 3: 19.3 Mbps
9	1a(this flow between positions A) + 1b (this flow between positions B)	Flow 1a: 13 Mbps Flow 1b: 13 Mbps

Part 3: Throughput tests w.r.t crosstalk possibility
Tests were performed only with flows 1 and 3 with each one with fixed pair of antennas between machines M1 and M2. However for each iteration, the interfaces on flows 1 and 11 were changed and results were obtained. The important thing in these tests is that the antenna positions were fixed

throughout the tests. We are trying to find whether there is any “crosstalk” between the wireless cards on the same machines. 2 servers were run on machine M2 whereas 2 clients were run on machine M1. In the table, for the throughput columns, ‘I’ represents the throughput when the flows (1 and 3) were run individually whereas C represents the throughput when the flows (1 and 3) were run together (co-operatively)

Table 3.1:
On machine M1: Server interface ath0 for flow 1 and interface ath1 for flow 3.

Test	Client flow 1 interface	Client flow 3 interface	Throughput Flow1 (I) Mbps	Throughput Flow3 (I) Mbps	Throughput Flow1 (C) Mbps	Throughput Flow3 (C) Mbps
1	Ath0	Ath1	26	26	25	7
2	Ath0	Ath2	26	27	23	14
3	Ath0	Ath3	26	26	23	13

Table 3.2:
On machine M1: Server interface ath0 for flow 1 and interface ath2 for flow 3.

Test	Client flow 1 interface	Client flow 3 interface	Throughput Flow1 (I) Mbps	Throughput Flow3 (I) Mbps	Throughput Flow1 (C) Mbps	Throughput Flow3 (C) Mbps
1	Ath0	Ath1	24	21	16	7
2	Ath0	Ath2	26	24	14	8.5
3	Ath0	Ath3	21	21	14	18

Table 3.3:
On machine M1: Server interface ath1 for flow 1 and interface ath2 for flow 3.

Test	Client flow 1 interface	Client flow 3 interface	Throughput Flow1 (I) Mbps	Throughput Flow3 (I) Mbps	Throughput Flow1 (C) Mbps	Throughput Flow3 (C) Mbps
1	Ath0	Ath1	28	28	25	19
2	Ath0	Ath2	26	20	20	13
3	Ath0	Ath3	25	21	18	14

7. Analysis

7.1 Channel assignment algorithm analysis

The channel assignment algorithm first calculates the routes from the gateway to all the other nodes using dijkstra algorithm, assigns channels to the links on the thusly formed minimum spanning tree and thereafter starts assigning channels to the rest of the links in the graph. In this way, the links that are on the minimum spanning tree get their channel assigned before the other links in the graph. This means that the links on the MST have higher priority over the other links. The rationale behind this approach is that the routing in the mesh network will most probably be done using dijkstra algorithm (Dijkstra algorithm gives shortest number of hops from each node to the gateway. Shortest hops are preferable because mostly they yield a good throughput).

Also, the link coloring (channel assignment) starts from the gateway. The rationale behind that is links closer to the gateway will carry more traffic (originating as well as relayed traffic) compared to links farther from the gateway. Hence if closer links are given more priority, then they will have a high probability of being interfered with other links.

However, the algorithm chooses a color for a link only on the basis of colors that are assigned to its neighboring links, i.e colors of links in 2 hop neighborhood are not considered. This is depicted in topology 5 where link 1-4 is assigned the same color (Red) as that of link 3-8. Hence nodes 8 and 4 become hidden from each other and simultaneous transmissions on the links will result in collisions. The important thing to note here is that link 3-8 is closer to gateway and hence more "important" as it carrier more traffic than other links farther from gateway. Hence the algorithm could have performed better if link 1-4 would have been assigned either Blue or Green color. This would have made that link to interfere either with links 8-4 or 8-0 but these links are "less important" compared to link 8-3 and the overall throughput of the network would have been better.

7.2 Throughput and antenna spacing analysis

Tables 1.1, 1.2 and table 2 are comparable since all of them show the throughput results for simultaneous flows on different channel for single hops. As it can be seen, with antenna spacing, the throughput for different simultaneous flows increases. For table A, even if the simultaneous flows are on orthogonal channels (1, 6 and 11), the throughput results show that they interfere with each other drastically degrading the throughput. Antenna spacing definitely increases the throughput.

7.3 Throughput and crosstalk possibility analysis

From the results of tables 3.1, 3.2 and 3.3, we can see that even for same antenna placement and same flows (1 and 3), we get varying results for flow throughputs when run individually. We can see that, when interfaces ath0 and ath2 are used on M1 (server), the co-operative throughput is low compared to the same when interface pairs (ath0, ath1) and (ath1, ath2) are used on M1 (server) are used for the flows. This may suggest the possibility of crosstalk amongst the mini-PCI wireless cards.

[4] shows that the crosstalk effect due to multiple wireless cards on the same machine can degrade the performance. It also gives reference to [5] which says the same. [6] however says that the experimental results which they have conducted using 4 mini PCI cards on the same machine show no performance degradation due to cross talk effect.

8. Conclusion

In this paper, we have implemented a static channel assignment protocol to solve the need to assign channels in order to minimize interference between adjacent links using the protocol described in Appendix. In addition to that, we performed extensive experiments using different topologies to measure the performance and throughput of our implementation. Some of the results obtained from the testing that was performed were quite unexpected such as experiencing lower throughput when using channels 1 and 6 or channels 6 and 11 together on adjacent links although they are non-overlapping.

It has been proved that it is extremely vital to perform channel assignment in wireless mesh networks to increase the available bandwidth in the network. But, our experiments also proved the importance of having a superior antenna arrangement to provide for better bandwidth and less interference even though the actual channel frequencies assigned are non-overlapping. Wireless mesh networks will undoubtedly be fertile ground for a lot of research ideas and implementations and the interest in this field is growing tremendously. Some of the future researches that could be conducted can range from trying to incorporate other sources of interference into the channel assignment protocol, conducting extensive tests on the optimal assortments of antennas, as well as trying to provide bandwidth guarantees for mesh routers within the mesh network.

9. References

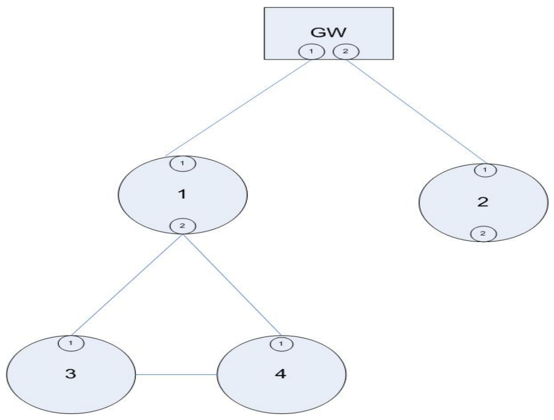
1. "Interference-Aware Channel Assignment in Multi-Radio Wireless Mesh Networks", Krishna N. Ramachandran, Elizabeth M. Belding, Kevin C. Almeroth, Milind M. Buddhikot, IEEE Workshop on Wireless Mesh Networks (WiMesh), 2005
2. "Static Channel Assignment for Multi-Radio Multi-Channel Multi-Hop Wireless Networks", Xufei Mao, Xiang-Yang Li, S. Kami Makki, **Proceedings of the 11th annual international conference on Mobile computing and networking**
3. "Routing and Interface Assignment in Multi-Channel Multi-Interface Wireless Networks", Kyasanur P., Vaidya, N.H., IEEE **Wireless Communications and Networking Conference, 2005**
4. "Experimenting with a Multi-Radio Mesh Networking Testbed", J. Robinson, K. Papagiannaki, C. Diot, X. Guo, and L. Krishnamurthy, In Proc. of WinMee, 2005
5. "Routing in multi-radio, multi-hop wireless mesh networks", R. Draves, J. Padhye, and B. Zill, In *ACM Mobicom*, Philadelphia, PA, 2004
6. "Heraklion *MESH*: An Experimental Metropolitan Multi-Radio Mesh Network", In **Proceedings of the the second ACM international workshop on Wireless network testbeds, experimental evaluation and characterization**
7. "Communication Framework User Manual", JunBum Lim, NC State University Wireless MeshBed Repository

Appendix

The Channel Assignment algorithm

Considerations:

We consider a connectivity graph (network topology) as shown in figure 1. This topology can be obtained from the topology manager of the mesh control plane. The topology shows a gateway (GW) with 2 radios connected to nodes 1 and 2 each having 2 radios and node 1 which connects to nodes 3 and 4 having 1 radio each. Now our aim is to use this topology as an input to the channel assignment algorithm which will run on the control node (gateway in this case) and will disseminate each node the channel assignment commands.



From this connectivity graph, a multi-radio conflict graph [1] is constructed as shown in figure 2. In the MCG graph, a link between any 2 radios in the connectivity graph is represented by a vertex. 2 vertices in the MCG graph are connected if the corresponding 2 links interfere with each other in the connectivity graph. A vertex in the MCG is denoted by [a-i , b-j] which represents a link between radio 'i ' of node 'a' and radio ‘j’ of node ‘b’. Also, in the connectivity graph, we assume that any 2 links of the same node interfere with each other which is further reflected in the MCG.

Figure 1

Also, in the MCG, we associate with each node a priority which is used by the channel assignment algorithm. In the simplest case, this priority depends upon the number of children of each of the 2 nodes that form a link. By children of a given node, we mean the number of nodes that use the given node to relay traffic back and forth from the gateway. For example, the number of children of node 1 in the connectivity graph of fig. 1 is 2. This number can also be found out from the topology manager. Hence the priority of a vertex [a-i , b-j] in MCG could then be just the average number of children of nodes a and b or the least of the number of children those 2 nodes have. The rationale behind choosing this metric is that with higher number of children, a node has higher amount of traffic and hence its corresponding links need to be given priority while assigning channels. Other additional factors that we may consider in future for the priority assignment are: delay and interference due to external sources on the links of the connectivity graph.

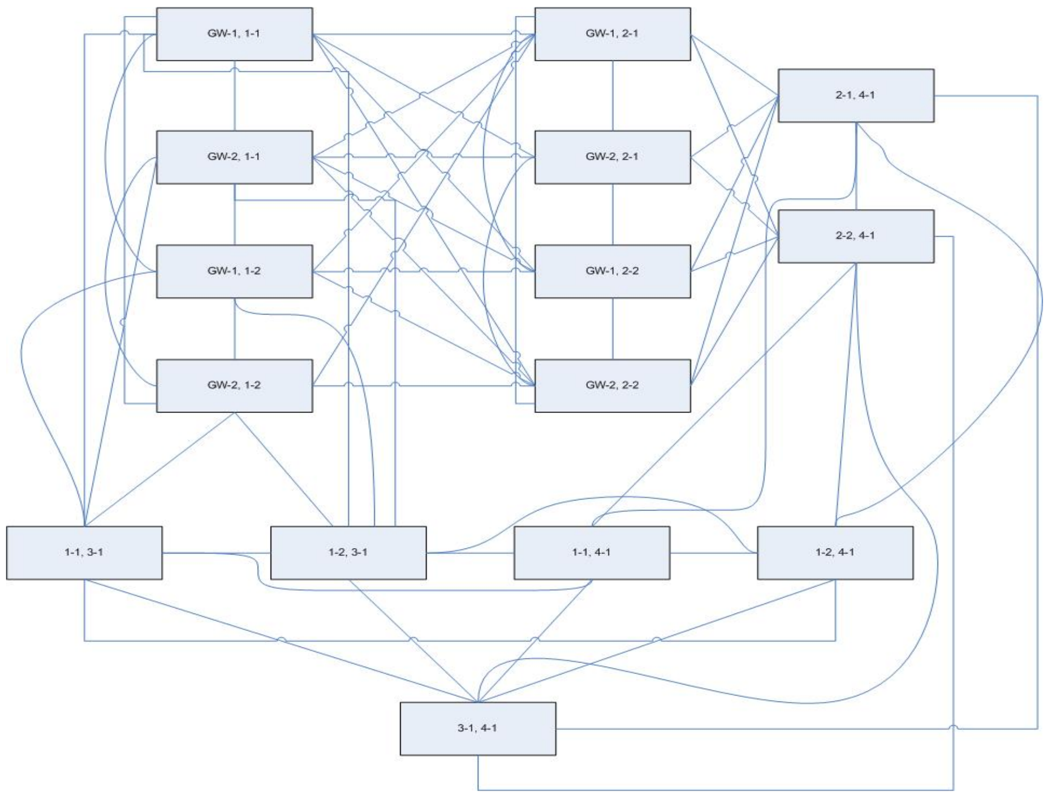


Figure 2

Algorithm:

Many ideas of the algorithm that we have shown below are derived from [1]. The algorithm is basically an edge coloring algorithm, with each vertex having a certain priority. The input to the algorithm is the MCG and a set of colors (the available channels in this case). We consider the following terms (some of which are derived from [1]):

1. Permanent color: If a permanent color is given to a vertex, then it means that that particular link in the connectivity graph has been permanently assigned the corresponding channel.
2. Temporary color: If a temporary color is given to a vertex, then it means that that particular link in the connectivity graph has been temporarily assigned the corresponding channel.
3. Color count: This is a list of each color and the number of times it is used to color a vertex in the MCG. While choosing from a set of colors, the algorithm picks up that color which has the least usage.

Now we present some conditions that the algorithm will encounter and the handling of those conditions:

A. If an uncolored vertex [a-i, b-j] in MCG is to be given a color, then assign it a color that is different from the colors of its adjacent vertices and which has the least usage count. If there is no color that is different from the color of adjacent vertices, then just give the vertex a color with the least usage count.
B. If a vertex [a-i, b-j] in MCG is given a color, then choose all the vertices having 'a-i' of 'b-j' in them and remove them along will all their links in the MCG. (This means that if a link between nodes a and b in the connectivity graph is assigned a channel, then no other links between a and b is to be considered for channel assignment)
C. If a vertex [a-i, b-j] in MCG is chosen to be colored, and it is assigned a temporary color, then look for any other vertex [a-k, b-l] (that is any other link in the connectivity graph between nodes having radios a and b) which is uncolored and color it using condition A. If no such uncolored vertex exists, then one of the vertices is to be chosen and made its temporary color permanent. This vertex is the one which has the least *conflict* with its adjacent vertices when its color is made permanent. The conflict of a given vertex with its adjacent vertices is equal to the sum of the priorities of the adjacent vertices which have the same color as that of the given vertex.
D. If a vertex in MCG gets colored by 2 or more temporary colors, then remove the vertex from the MCG along with its links. By this we mean that the physical link in the connectivity graph between the 2 radios is not possible.

Using the aforementioned 4 conditions, the algorithm is as follows:

Algorithm:

Input: MCG, a set of colors

```
1. while (! All vertices colored)
2. choose an uncolored or a temporarily-colored vertex [a-i, b-j] with the highest priority
   if ([a-i, b-j]  is uncolored)
           Assign it a permanent color according to condition A
   Else
           Choose vertex [a-k, b-l] related to vertex [a-i, b-j] and color it according to condition C
```

3. **3.** Remove vertices related to either [a-k, b-l] or [a-i, b-j] from the MCG according to condition B
4. **4.** Assign *temporary color* equal to the permanent color of [a-i, b-j] to all vertices [X-X, a-i] , [a-i, X-X], [X-X, b-j] , [b-j, X-X]. (Same is the case if vertex [a-k, b-l] is chosen)
5. **5.** If for a vertex v, if it gets assigned 2 temporary colors, remove vertex v and its links from MCG according to condition D.

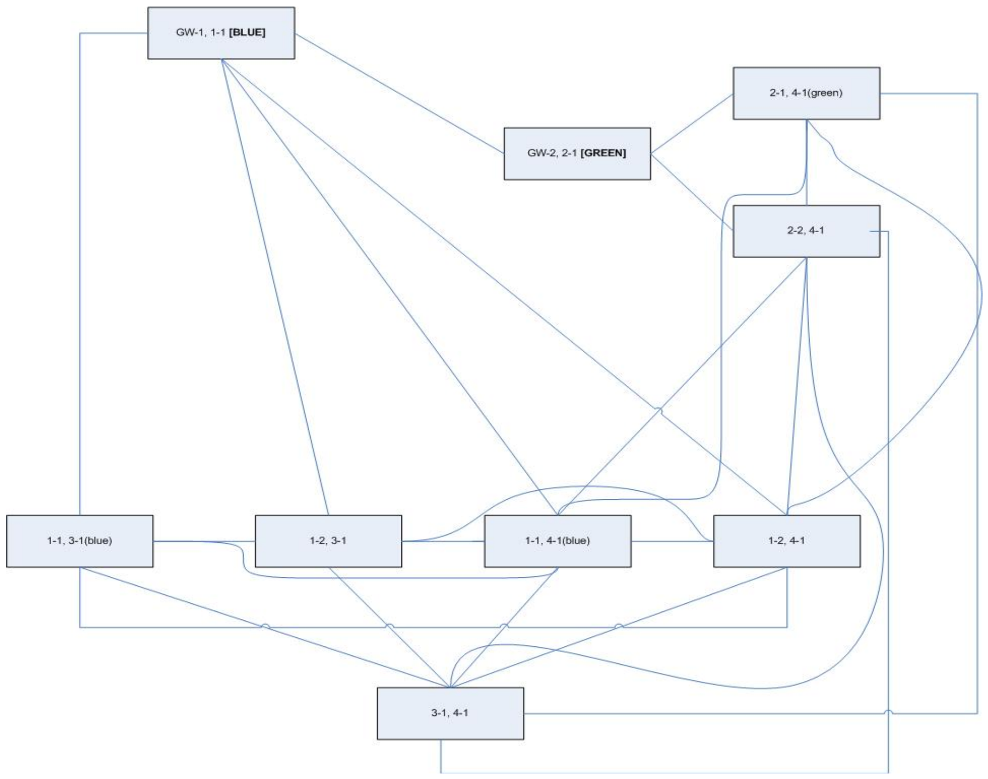


Figure 3

Example:

Now we will consider an example of channel assignment. The topology we will consider is that shown in figure 1 and its MCG in figure 2.

Let us consider the link priority assignment scheme that we have discussed above (i.e the children based scheme). We now list each step that the algorithm performs for channel assignment on the MCG. Also let us consider that we have 3 colors {BLUE, GREEN, RED}

1. Vertex with highest priority is [GW-1, 1-1]
2. It is permanently assigned color BLUE. (permanent colors are denoted by upper case)
3. From condition B, vertices [GW-1, 1-2], [GW-2, 1-1] and [GW-2, 1-2] and their links are removed from the MCG.
4. Also vertices [1-1, 3-1], [1-1, 4-1], [GW-1, 2-1] and [GW-1, 2-2] are temporarily assigned blue colors. (temporary colors are denoted by lower case)
5. The next highest priority vertex chosen is [GW-1, 2-1]

6. Since it is temporarily colored, according to condition C, an uncolored vertex [GW-2, 2-1] is chosen and permanently assigned the lease usage color GREEN.
7. From condition B, vertices [GW-1, 2-1], [GW-1, 2-2] and [GW-2, 2-2] and their links are removed from the MCG. Also vertex [2-1, 4-1] is temporarily assigned green color. Thus we get the resultant MCG graph shown in figure 3.
8. The next vertex with highest priority is [1-1, 3-1]. Since it has a temporary color blue, according to condition C, uncolored vertex [1-2, 3-1] is chosen and is permanently assigned the color RED whose usage count is 0.
9. From condition B, vertex [1-1, 3-1] is removed.
10. Vertices [1-2, 4-1] and [3-1, 4-1] are temporarily assigned color red.
11. The next highest priority vertex is [1-1, 4-1]. Since it has a temporary color blue, and since vertex [1-2, 4-1] also has a temporary color (red in this case), according to condition C vertex [1-2, 4-1] is chosen since it has the least conflict. ([1-2, 4-1] conflicts with low priority vertex [1-2, 3-1] and [1-1, 4-1] conflicts with high priority vertex [GW-1, 1-1]). Vertex [1-2, 4-1] is permanently assigned RED color.
12. According to condition B, vertex [1-2, 4-1] and its links are removed from the MCG.
13. Also, vertices [2-1, 4-1] and [2-2, 4-1] are assigned temporary color red.
14. Now since vertex [2-1, 4-1] have 2 temporary colors namely green and red, it and its links are removed from MCG.
15. Next high priority vertex [2-2, 4-1] is chosen and since it has a temporary red color, it is permanently assigned RED color according to condition C.
16. The last vertex [3-1, 4-1] is chosen and since it has a temporary red color, it is permanently assigned RED color according to condition C.
17. The resultant MCG graph obtained is shown in figure 4.
18. Colors blue, green and red correspond to channels and they are assigned to the topology graph accordingly.

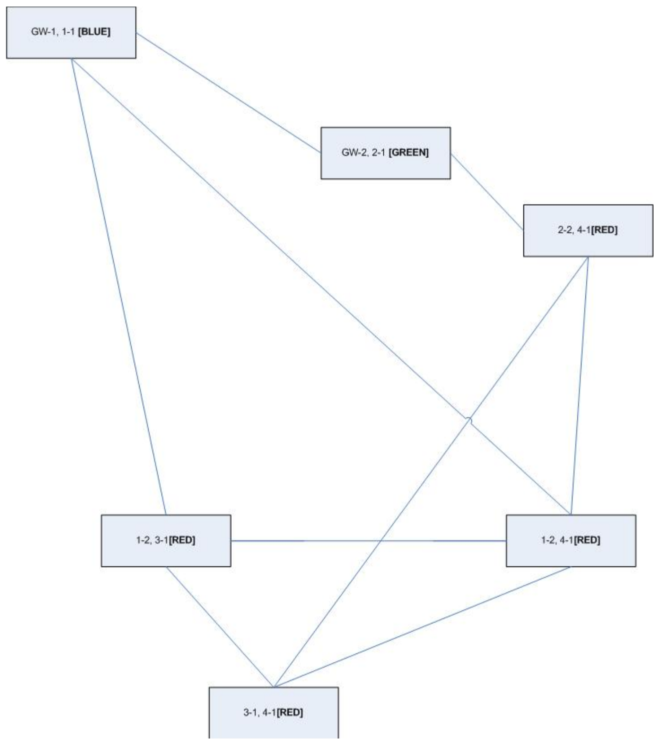


Figure 4

Actual algorithm invocation on the Channel Manager:
In the channel algorithm, the aforementioned algorithm is invoked thusly:

1. Run dijkstra algorithm on the graph 'g' with gateway node G to get MST(G). MST(G) determines the routing tables on each of the nodes for routes to and from the gateway node G.
2. For each link l(G) in MST(G), run the CA algorithm
3. For each link in the residual graph g-MST(G), run the CA algorithm.

1st Feb 2010

Addendum:

1. Addressing scheme: The Meshbed operates in a /16 subnet (each card has an IP address with netmask 255.255.0.0)
2. Database change: The database has been migrated from BDbXML to MySQL owing to the concurrency problems in BDbXML. A set of APIs has been provided for the MySQL database which can by used by any of the application modules.
3. IP address interpretation of each node: Formerly, the IP address of a card in the channel assignment code used to take the format of:

    10.0.<node ID>.<radio ID>

   Now the new format of the IP address is:

    10.2.<radio ID>.<node ID>

This change was done to make the channel assignment code compatible with the MeshBed software.
4. Routing change: The channel assignment software used to do a simple host-specific routing after the channel assignment. This has been removed from the software and now only channel assignment is done. Routing is explicitly taken care of by the Routing manager/agent which is also an application in the MeshBed software.