Dynamic Interference Adaptation for Wireless Mesh Networks

Dheeraj Agrawal[1], Arunesh Mishra[1], Kevin Springborn[1], Suman Banerjee[1], Samrat Ganguly[2]

[1] University of Wisconsin-Madison [2]NEC Labs, NJ

{*dheeraj,arunesh,kevin,suman*}*@cs.wisc.edu* *samrat@nec-labs.com*

ABSTRACT

Most wireless meshes will have to operate within the crowded unlicensed spectrum that is also shared by numerous uncoordinated 802.11 hotspots [7]. This creates an unpredictable and variable spectrum space that mesh networks need to co-exist within. We propose a novel method for adapting to such external interference by dynamically changing the assignment of channels to the backbone links, yet retaining the same "logical" network-wide channel assignment. Called *Connected-component based Adaption*, this method ensures two important properties, (i) Allows distributed changes to the channels used by backbone links depending on local interference, and (ii) Allows a centralized algorithm to dictate the high-level channel and route assignments used by the network as a whole. We propose *MeshChop* as an randomized algorithm that uses channel hopping to achieve connected-component based adaptation. We show that MeshChop has minimal overheads and provides good link quality and throughput through dynamic adaptation. We present preliminary experimental results which show that MeshChop can achieve almost 80% improvement in throughput over non-adaptive schemes. We believe that component based adaptation using channel hopping is the right method to adapt to local interference conditions without causing network-wide changes.

1. INTRODUCTION

A lot of research has gone in coming up with efficient channel assignment schemes for multi-radio multi-hop wireless mesh networks. Past work [3, 4] have looked at centralized solutions which address this issue. These solutions typically look at the traffic pattern, capacity requirements and make routing and channel assignment. However, they do not consider the effect of interference from co-located wireless networks, such as hotspots and other commercial/non-commercial wireless deployments [7], on channel assignment.

The extent of interference from these co-located wireless networks will vary depending on the amount of actual traffic from them. Also, as observed from extensive traffic traces in hotspots, traffic from them varies quickly and unpredictably. This can be attributed to the number of users using these networks and their network usage patterns. It is difficult for a centralized assignment algorithm to efficiently adapt to such rapidly and unpredictably varying wireless environments. An efficient approach to address this problem of external interference is to move the task of making channel switching decisions from the centralized body to the routers themselves that experience the interference.

In this paper, we propose *MeshChop* that uses a unique *connected component* based channel hopping approach to make local adaptations to external interference and still reaps the benefits of centralized assignment. In particular MeshChop has two important properties

- *No change to network topology* - A simple localized scheme for channel re-assignment would be, for every router interface to independently make the decision to switch to a new channel if the current channel on which it is on gets congested above a threshold limit. This scheme has disadvantages. If two neighboring router interfaces A and B, initially on the same channel, independently decide to switch to another channel, they might lose their existing connectivity. This will breakdown all flows between the link A-B. The flows will resume only after the routing algorithm running over the mesh network discovers new routes for these flows which is a slow process and can take a few seconds. This will greatly bring down network throughput and have disastrous effects on real time applications such as VoIP. Even worse, if a router interface loses connectivity with neighboring interfaces after switching to a new channel, it might not be able to find connectivity to any other neighboring interface on that channel. This will partition the mesh network and stop all flows passing through that router interface. MeshChop uses a neat connected component based channel adaptation approach to overcome this problem. In this approach, all interfaces that are on the same channel and act as a single-hop link to each other together switch to a new channel. This maintains connectivity between existing neighbors while still moving the interfaces on to a new non-congested channel.

- *Minimum to zero overheads* - A typical channel adaptation scheme, centralized or localized, that adapts to external interference will have some basic overheads. Firstly, to decide a new channel for a router interface that experiences congestion on the current channel, a channel adaptation scheme will need to probe the quality of other channels. This involves considerable overhead [9]. Secondly, there will be communication overheads related to channel switching. If the channel adaptation is done independently by every router interface there will be communication overheads in discovering dead (broken) and new links formed due to channel switching. If the scheme is centralized, there will be communication overheads in frequently propagating channel quality information to the central server responsible for channel assignment decisions. Meshchop cuts down the overheads drastically by employing a simple randomized channel re-assignment technique using only one time information.

MeshChop borrows the concept of opportunistically and greedily switching between different channels using channel hopping from our prior work [7] to balance the channel bandwidth given to every wireless node. MeshChop can be summarized as follows. As a starting point we assume that a good centralized algorithm has already made a channel assignment and assigned routes to the mesh network. Time is divided into slots each slot being a few seconds long. At the beginning of each slot, every *connected component*, which consists of nodes on the same channel (explained later), switches to one of the available non-overlapping channels randomly. Different random channel hopping sequences for every *connected component* can be provided by assigning a unique seed to each *connected component* when the centralized algorithm is run. The centralized algorithm can then be run periodically (on the order of hours) to account for any long term capacity requirement or traffic pattern changes.

The key contributions of the paper are as follows

- Connected component based adaptation as the right approach to co-exist with other wireless networks which act as interference.
- Channel hopping based scheme, called MeshChop, as the mechanism to realize connected component based adaptation with low overheads.

The rest of the paper is organized as follows. Section 2 introduces the idea of connected components which forms the basis of MeshChop. Section 3 illustrates the algorithm for *MeshChop* and evaluates its efficacy using some preliminary experimental results. Finally, we conclude the paper in section 4.

2. CONNECTED COMPONENT BASED ADAPTATION

In this section we explain the importance of connected component based channel adaptation scheme to adapt to interference from co-located wireless networks. Figure 1 shows a mesh network of 5 routers with routers *a* and *b* acting as gateways. Routers c and d have two interfaces while routers a, b, e, f and g have one interface. Assume, a snapshot of the mesh network with channels assigned by a centralized channel assignment algorithm to each of the router interfaces as shown in Figure 1.

This assignment will work well as long as the *traffic pattern* in the mesh remains nearly same and there is no *external interference*. If either of these variables change, the current channel assignment might not be suitable and the channels might have to be re-assigned. In the past, people have looked at changing traffic patterns and solutions aimed at adapting to them [4]. However, people have looked little into the other variable i.e external interference [8]. In this paper, we study the effects of external interference on a mesh network.

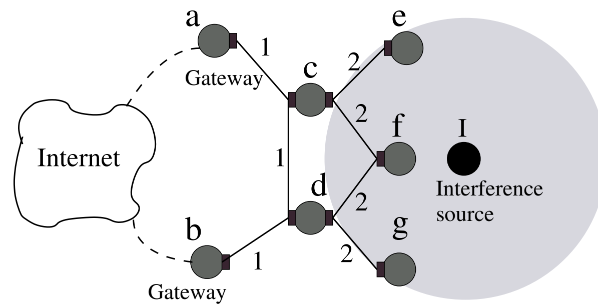

Figure 1: A mesh topology with 2 connected components formed by routers on non-overlapping channels 1 and 2. Routers a and b act as gateways.

Effect of external interference. To understand the gravity of *external interference* consider an access point (AP) (shown as I in Figure 1) from a co-located hotspot on channel 2. The interfaces of routers c,d,e,f and g which are on channel 2 will all suffer congestion due to interference from I thereby degrading throughput from these routers. The extent of interference will depend on the traffic from I. In the given mesh this single source of interference I can drastically degrade the throughput of multiple flows g-d-b, f-c-a, e-c-a etc. Given the mesh network topology in Figure 1, it would have been a better channel assignment if routers c,d,e,f and g were on a different channel (e.g. channel 3).

Variation in external interference. The nature of interference from co-located wireless networks might exhibit variation. For e.g., in Figure 1 if traffic from node I drops, the routers on channel 2 will be able to sustain their respective flows. These events are unpredictable as seen from hotspot traces. Therefore, it becomes difficult for a centralized algorithm to identify at the granularity at which it needs to perform channel re-assignment. Also, the time scale at which external interference varies might be small such as a few seconds. In such a case, the centralized assignment algorithm, due to computational and communication overheads, will not be able to effectively adapt to rapidly varying external interference. This leaves a scope for some local adaptation schemes to fill the gap between no change in assignment and frequent centralized assignment.

Connected component driven adaptation. For a local channel based adaptation scheme to be successful, it needs to *incur minimum overheads* and at the same time provide *throughput benefits*. As a plausible solution, every router interface can independently decide to switch to a new channel when it sees excessive external interference on the current channel. This might cause *two* problems. Firstly, it might cause problems of *route stabilization* by allowing a router interface to independently decide the channel to switch to, it might lose connectivity to current neighboring interfaces and create broken links. In Figure 1, f will lose connectivity with the interfaces of routers c and d on channel 2. Any routing protocol running over this mesh network, will take some time to stabilize the network to account for the broken links and the newly formed links. In the

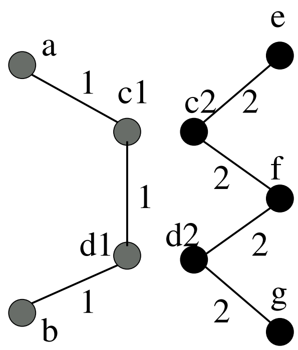

Figure 2: The two connected components for the mesh network in Figure 1. c1,c2 represent interfaces 1 and 2 of router c. Same for router d.

meantime, flows using the broken links will suffer. Secondly, by independently switching to a new channel a router interface might not only lose connectivity with existing neighbors but also be unable to get connectivity with any other interface thereby causing *network partition*. In Figure 1 if f is assigned channel 3, it will lose connectivity with c and d and, will not be able to get connectivity with any other router. Ramachandran et al. [8] have looked at centralized solutions which employ per node re-assignment. To overcome the problem of partition, they require one interface of every router to be on a default channel. This guarantees connectivity but degrades the quality of the mesh network due to multiple interfaces using on the same channel.

A more efficient approach is to switch the affected router interfaces to a different channel without losing connectivity with existing neighbors. This will remove the problems of *route stabilization* and *network partition*. In Figure 1, such an approach would imply switching the interfaces of routers c,d,e,f and g together from channel 2 to channel 3. These interfaces together form a *connected component*.

Formally, a connected component can be defined as follows. Consider an undirected graph $G = (V, E)$ where V is the set of all interfaces of mesh routers. An edge $(u, v) \in E$ indicates that router interfaces u and v are on the same channel and form a link with each other. A connected component in the graph G compromises of vertices (router interfaces) in G that are on the same channel and is a maximal connected subgraph in G. The connected component in G is the corresponding connected component in the mesh network. Figure 2 shows the connected components for the mesh network in Figure 1.

Applicability of connected component. MeshChop applies connected component as a tool to improve throughput of the mesh network in presence of co-located wireless networks. But this approach also has a hidden cost. It is possible that an external interference source may interfere with only one or two routers out of the many nodes in a connected component. In that case, switching all the router interfaces of that connected component might expose the other router interfaces in the connected component to new external interfering sources. For e.g. assume in Figure 1 only router interface of f faces interference from I. Now, if the entire connected component e-$c2$-f-$d2$-g is switched from channel 2 to channel 3 because of interference to f, all the interfaces of the connected component (other than f) that saw no interference on channel 2 might now see new interfering sources on channel 3. This will not be a serious problem if the number of router interfaces in a connected component are either located close to each other such that most of them suffer the same external interference or the number of router interfaces in a connected component is small. We try to incorporate the second approach of having small connected components by keeping a constraint in the centralized algorithm to keep the number of router interfaces in the connected component below a threshold, T_{thresh}.

3. MESHCHOP

MeshChop utilizes the concept of connected component to improve throughput of a mesh network affected due to interference from co-located wireless networks. In this section, we first describe the the MeshChop algorithm and then evaluate its efficacy through some preliminary experimental results.

3.1 MeshChop Algorithm

MeshChop is based on the idea of switching wireless router interfaces to a better channel if the current channel gets congested due to interference from co-located networks. To determine the next best channel to switch on is a tough question for *two* reasons. Firstly, it is tough to accurately probe the link quality on a channel [9]. Secondly, probing channels involves considerable overheads due to time spent switching onto every channel and probing them. Therefore in MeshChop, we employ a *randomized* scheme i.e. all router interfaces of a connected component switch to the same random non-overlapping channel. In this scheme, every connected component is assigned a randomized channel hopping sequence such that at any point in time the current channel assignment in the mesh is *isomorphic* to that assigned by the centralized algorithm. This implies that the logical channel assignment to interfaces in mesh remain the same during channel hopping. Only the physical channel assignment is changed to counter external interference. Random channel hopping allows the connected component to spend its time between channels which are congested and those which are not and gives an average case behavior which is far better than the worst case where a connected component experiences continuous external interference. Also, MeshChop has *minimum overheads* as the router interfaces do not need to probe channels and also don't need to exchange any messages to switch to a channel. MeshChop has the following key steps.

Step 1: Centralized channel assignment. In this step, a good centralized channel assignment algorithm assigned channel to all router interfaces based on capacity requirements.

Step 2: Connected component identification. This is the first and the most basic step of MeshChop. In this step, MeshChop identifies the connected components in the mesh after the centralized channel assignment. This can be done by using simple graph theory techniques.

Step 3: Time synchronization. To avoid temporary link

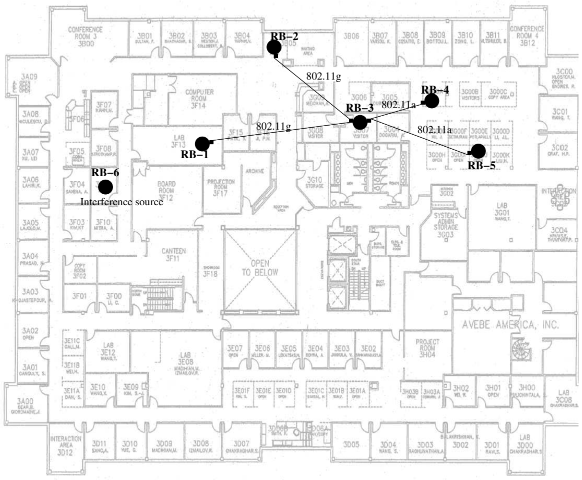

Figure 3: Testbed for mesh network. Nodes RB-1,RB-2 and interface 1 of node RB-3 are configured on 802.11g and nodes RB-4,RB-5 and interface 2 of node RB-3 are configured on 802.11a. RB-6 is configured on 802.11g and acts as the external interference source. It is in transmission range of RB-1 and interference range of RB-2.

connectivity failures due to asynchrony in channel switching times of the router interfaces of a connected component, we run a time synchronization protocol over the mesh network.

Step 4:Channel hopping. Once the connected components are identified, MeshChop assigns a randomized channel hopping sequence to the connected components which is isomorphic to the centralized channel assignment. This hopping sequence for a connected component is sent out to all the routers whose interfaces belong to that component. For distributing the hopping sequence to respective routers, MeshChop utilizes the underlying mesh network. All connected components follow their hopping sequence until the centralized algorithm builds a new logical channel assignment for the mesh network.

3.2 Experimental Results

Here, we quantify the benefits of randomized channel hopping for countering external interference using a small implementation based experiment over a mesh network of five routers. Through our experiments we find that MeshChop gives 2X improvement over schemes which do not adapt to external interference. However, due to the randomized nature of the algorithm it gives an average case performance and so its benefits lie between the case where there is no adaptation to external interference and the case when there is no external interference. With new wireless cards in the market that can switch on the order of 100μs, the overheads in channel switching can be easily amortized by switching on the order of seconds [7]. These experiments, however, do not provide detailed analysis and scenarios of performance of MeshChop. We are currently investigating that and aim to study MeshChop based on a full-fledged testbed based implementation.

Figure 3 shows the testbed for the experiment. It consists

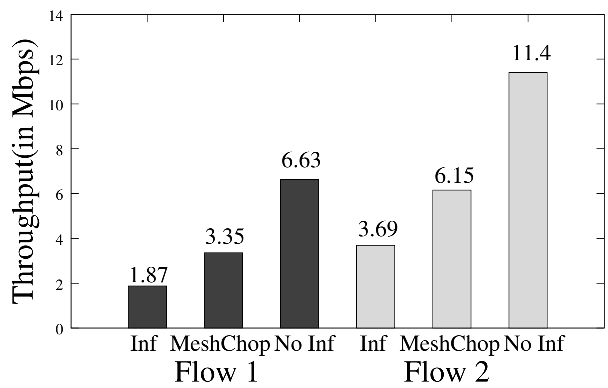

Figure 4: Figure shows throughput for the two flows for three scenarios - no external interference (No Inf), with external interference and using MeshChop and (MeshChop), with external interference and without using MeshChop (Inf)

of 5 machines RB-1 to RB-5 running Linux kernel 2.4 and using atheros chipset based wireless cards. Interface 1 of RB-3, RB-1 and RB-2 are configured on the same channel in 802.11g band and form one connected component. Interface 2 of RB-3, RB-4 and RB-5 are configured on the same channel in 802.11a band and form the other connected component. The *two* interfaces of RB-3 are configured on different bands because we saw significant interference between the two interfaces even when they were on non-overlapping channels. We generated two constant bit rate UDP flows *flow1* RB-2 to RB-5 and *flow2* RB-1 to RB-4 using *iperf* to see the variability in the effects of interference on these flows. We found the effects to be similar. Another machine RB-6 is used to act as an external source of interference and is configured in 802.11g such that it is in transmission range of RB-1 and interference range of RB-2. RB-6 broadcasts packets at the rate of 4Mbps. The driver was suitably modified to allow this. Also, the RB-6 switches between channels 1,6 and 11 in 802.11g mode at every 200s interval to mimic the variability in external interference. Figure 4 shows the throughput obtained for the two flows for three set of experiments. In the first set (*Inf*), RB-1,RB-2, and interface 1 of RB-3 are configured on static channel 6 in 802.11g mode. RB-4, RB-5 and interface 2 of RB-3 are configured on static channel 36 in 802.11a mode and RB-6 acts as interfering source. In the second set, MeshChop was run on the mesh network with a hopping duration of 20s. In the third set (*No Inf*), the interference source was removed.

As is clear from Figure 4, *MeshChop* performs far better than the *Inf* case where the mesh network is configured on static channels and represents the worst case scenario and gives a throughput benefit of about 80% for *flow1* and 90% for *flow2*. However, MeshChop gives an average case performance and so its benefits are half way between worst case (*Inf*) and best case (*No Inf*) as seen in Figure 4. Figure 5, shows the variation of throughput of *flow1* with time when using MeshChop. When the interference source RB-6 and, RB-1 and interface 1 of RB-3 are on same channel, throughput of *flow1* drops drasti-

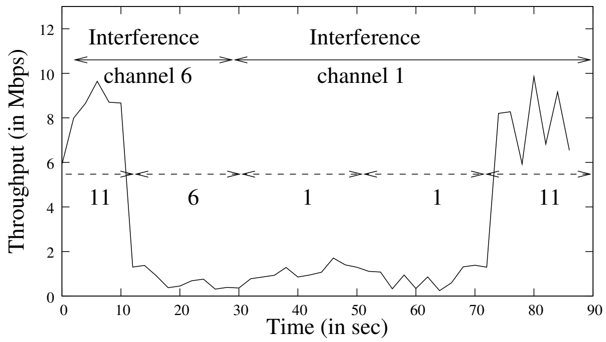

Figure 5: Figure shows variation of UDP throughput of flow1 with time.

cally and when they are not on the same channel, throughput of *flow1* stays high ($\geq$ 6Mbps). MeshChop keeps switching the flows between the good and bad scenarios to improve upon the worst case performance. This results clearly show that simple randomized schemes of channel adaptation can greatly boost the throughput of mesh networks facing external interference. Currently, we are investigating schemes that can more intelligently adapt by characterizing external interference and making deterministic choices to spend more time on channels that offer less congestion.

4. CONCLUSION AND FUTURE WORK

In this paper, we have addressed the problem of interference to a mesh network from co-located wireless networks. We propose channel hopping in mesh on a per connected component basis to tackle the problem. Our scheme uses randomized channel hopping scheme and provides considerable throughput improvements over a non-adaptive scheme determined through test-bed based implementation. This paper provides an insight into the potential of simple adaptive channel hopping schemes for countering interference from external wireless networks. The channel hopping scheme provides a low overhead solution for throughput improvement even in scenarios where a mesh network experiences interference from dynamically changing external wireless environments. We are currently investigating methods to improve our scheme by studying and characterizing external interference from sources such as hotspots and using that as a feedback to make more determinstics choices in switching channels. This will allow the mesh routers to spend more time on channels which are less congested. Further, we plan to a full-fledged testbed based implementation in an environment with existing commerical deployments such as hotspots acting as external sources of interference and characterize the benefits of our channel hopping based approach.

5. REFERENCES

[1] P. Bahl, R. Chandra, and J. Dunagan, “Ssch: Slotted seeded channel hopping for capacity improvement in ieee 802.11 ad-hoc wireless networks,” in *ACM MobiCom ’04*.

[2] Pradeep Kyasanur and Nitin H. Vaidya, “Capacity of multi-channel wireless networks: impact of number of channels and interfaces,” in *ACM MobiCom ’05*, New York, NY, USA.

[3] Randeep Bhatia Mansoor Alicherry and Li Li, “Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks,” in *ACM MobiCom ’05*.

[4] A. Raniwala and T. Chiueh, “Architecture and algorithms for an ieee 802.11-based multi-channel wireless mesh network,” in *Infocom ’05*.

[5] P. Steenkiste A. Akella, G. Judd and S. Seshan, “Self management in chaotic wireless deployments,” in *ACM MobiCom 2005*.

[6] “Earthlink, google win san francisco wi-fi bid,” http://www.bbwexchange.com/publications/page1417-123662.asp.

[7] A. Mishra et al, “Distributed channel management in uncoordinated wireless environments,” in *ACM MobiCom ’06*.

[8] K. Almeroth K. Ramachandran, E. Belding and M. Buddhikot, “Interference-aware channel assignment in multi-radio wireless mesh networks,” in *Infocom ’06*.

[9] K. Kim and K. Shin, “On accurate measurement of link quality in multihop wireless mesh networks,” in *ACM MobiCom ’06*.