A Multi-Radio Unification Protocol for IEEE 802.11 Wireless Networks

Atul Adya, Paramvir Bahl, Jitendra Padhye, Alec Wolman, Lidong Zhou

Microsoft Research
One Microsoft Way, Redmond, WA 98052.

{adya, bahl, padhye, alecw, lidongz}@microsoft.com

July 2003
Technical Report
MSR-TR-2003-44

We present a link layer protocol called the *Multi-radio Unification Protocol* or MUP. On a single node, MUP coordinates the operation of multiple wireless network cards tuned to non-overlapping frequency channels. The goal of MUP is to optimize local spectrum usage via intelligent channel selection in a multihop wireless network. MUP works with standard-compliant IEEE 802.11 hardware, does not require changes to applications or higher-level protocols, and can be deployed incrementally. The primary usage scenario for MUP is a multihop community wireless mesh network, where cost of the radios and battery consumption are not limiting factors. We describe the design and implementation of MUP, and analyze its performance using both simulations and measurements based on our implementation. Our results show that under dynamic traffic patterns with realistic topologies, MUP significantly improves both TCP throughput and user perceived latency for realistic workloads.

I. Introduction

Despite a tremendous amount of research in single-radio multihop wireless networks, network capacity continues to limit the scale at which these networks perform well. Studies have shown that end-to-end throughput decreases rapidly as node density and number of hops increases [1], [2].

A fundamental reason for low network capacity is that wireless LAN (WLAN) radios cannot transmit and receive at the same time. Consequently, the capacity of relay nodes is halved. Another limiting factor on network capacity is the interaction between network congestion and the sub-optimal backoff algorithms in both the lower-layer MAC protocols [2] and the higher-layer transport protocols [3]. Network congestion increases as node density increases, and this leads to rapid degradation in throughput.

Another fundamental limitation of standard-compliant IEEE 802.11 WLAN radios [4] is that they operate over only a small portion of the available spectrum called a channel. Although multiple non-interfering channels are available, the IEEE 802.11 physical (PHY) layer is designed to use only a single frequency channel at any given time. This works well for infrastructure-based WLANs because additional capacity is obtained by dividing the physical space into "cells" and operating neighboring cells on non-overlapping (orthogonal) channels.

Unfortunately, this design is not appropriate for multihop wireless networks. The problem is that, if a wireless node chooses a channel that is orthogonal to the channel chosen by its neighbors, then these neighboring nodes are not able to communicate with each other. If nodes are allowed to switch channels dynamically, then coordination is necessary for them to agree on a common channel; such coordination is non-trivial. Further, the delay in switching channels tends to be on the order of a hundred milliseconds, which causes a significant decrease in performance. Also, it is possible that the node misses an RTS/CTS exchange on one channel when listening on another, causing the hidden terminal problem to re-surface. For all these reasons, to the best of our knowledge, systems built over the IEEE 802.11 standard operate using only one radio and one channel.

To not use all the available channels is equivalent to not using the entire available spectrum, which in turn is equivalent to artificially limiting the achievable bandwidth. To use the entire spectrum without incurring the cost of switching delays, one would have to use multiple radios tuned to specific channels.

We propose and evaluate a new link layer protocol, called the *Multi-radio Unification Protocol* (MUP), that coordinates multiple IEEE 802.11 radios operating over multiple channels. The objective is to exploit the available spectrum as efficiently as possible and extract the highest bandwidth possible from existing technology.

We have designed MUP with the following four goals:

- *MUP must not require hardware modifications*. MUP works over standard-compliant IEEE 802.11 hardware. MUP requires a priority mechanism such as that provided by the 802.11e standard [5] for which hardware will soon be available.
- *MUP must support existing applications and networking protocols*. MUP does not require any changes to the application, transport, or routing protocols.
- *MUP must inter-operate with legacy hardware*. MUP does not assume that every node in the network has multiple radios or is MUP-enabled. The protocol works correctly in mixed-mode topologies.
- *MUP must not require global knowledge of network topology*. MUP is not about assigning channels optimally in a multi-hop network, instead it is about using pre-assigned channels efficiently.

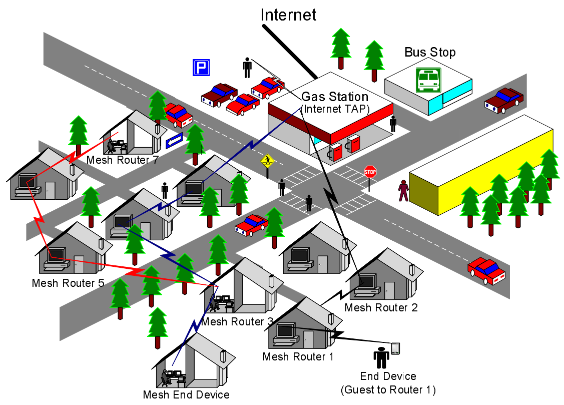

Fig. 1: Community mesh wireless networks

These goals stem from our desire to be pragmatic in designing a high-capacity multihop wireless network (mesh network) that can be deployed today. To the best of our knowledge, ours is the first paper that proposes a multiple NIC architecture that increases capacity in a mesh network by optimizing the use of the available spectrum locally with standard-compliant IEEE 802.11 hardware.

By preserving the IEEE 802.11 protocol; by requiring no changes to existing applications and protocols, and by ensuring interoperability with legacy nodes (nodes with a single IEEE 802.11 card) MUP can be deployed incrementally.

In this paper, we describe the design and implementation of MUP and analyze its performance using implementation and simulations. We evaluate MUP using a "realistic" setting that includes the network topology of a major city in the United States, and simulated web traffic.

II. Motivating Scenario

We have developed MUP to enable the wireless networking scenario shown in Figure 1. We are interested in building technology that allows neighbors to form a wireless community mesh network. There are several advantages to enabling such connectivity and creating a community resource. Groups of neighbors can use the mesh to share a fast and cost-effective Internet gateway; they can deploy cooperative distributed backup technology [6] and not have to worry about losing information due to disk failure; they can access and share locally relevant multimedia information.

The community mesh networking scenario described above has two key properties that affect our design: **routers are not mobile** and **power is not an issue**. Power is unlimited because mesh routers can be plugged into an electric outlet. Given the plummeting cost of IEEE 802.11 chipsets and unlimited power, we can include multiple radios in a mesh router and use them intelligently to increase the mesh network capacity.

III. Related Work

A few companies are field testing wireless mesh networks that provide broadband Internet access to communities that previously did not have such access [7], [8], [9], [10]. Our work is similar in spirit, but our approach differs in that we employ multiple radios in our router nodes to increase the capacity of the backbone mesh. None of the commercially available systems that we know of do this.

Several researchers have studied the effect of node density on end-to-end throughput and overall network capacity [11], [1], [2]. Using evidence from deployed IEEE 802.11 wireless meshes, these researchers conclude that the observed capacity is far below the theoretical optimum. Further, they observe that throughput degrades quickly as the number of hops increases. A reason for this is that the IEEE 802.11 MAC is inherently unfair and it can stall the flow of packets over multiple hops. Another reason is that these networks use only a small portion of the spectrum and a single radio for transmitting and receiving packets.

One way to improve the capacity of wireless meshes is to use a better MAC. Several proposals have been made in this regard [12], [13], [14], [15], [16], [17]. While the objective of these proposals is similar to ours, i.e. to exploit multiple non-interfering frequency channels, their approach is significantly different. These proposals require changes to the MAC and new hardware. In contrast, we do not require any changes to the IEEE 802.11 MAC protocol and consequently, MUP can be deployed incrementally on standard-compliant hardware. This we believe is MUP's primary strength and our key contribution.

An alternative way to improve capacity is to stripe the traffic over multiple network interfaces. Towards this end, there have been many proposals, including striping at the application layer [18], [19], [20]; at the transport layer [21], [22], and at the link layer [23], [24], [25]. Each approach has its advantages and disadvantages. Striping at the application layer yields poor aggregate bandwidth, sometimes even lower than that of the slowest connection, because a slow connection can stall faster ones [21]. Striping at the link layer (also referred to as inverse queueing) yields poor performance because the proposed mechanisms are highly sensitive to lossy links and to fluctuations in transmission data rates [26], a phenomena that is common in wireless networks.

Many of the striping strategies either require changes to the application and transport layer or they suffer from significant timeout problems due to packet resequencing. MUP does not require any changes in applications, transport, and routing protocols, nor does it suffer from the resequencing problem. Moreover, unlike MUP, proposals that incorporate striping do not necessarily work with legacy nodes, defined as nodes with a single network interface card. We provide an in-depth comparison between striping and MUP in Section VII-B.3.

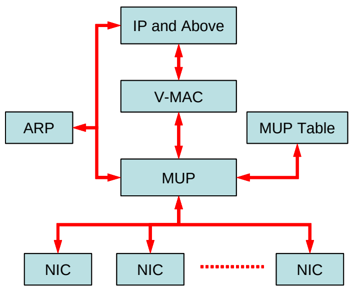

Fig. 2: MUP Architecture diagram

IV. Protocol Description

The high-level architecture of MUP is shown in Figure 2. MUP conceals multiple NICs from layers above it by presenting a single virtual interface. MUP then periodically monitors the channel quality on each interface, to each of its neighbors. Then, when it comes time to send a packet to a neighbor, it selects the right interface to forward the packet on.

The rest of the section is organized as follows. We begin by discussing several decisions we made while designing MUP. Then we delve into the details of the protocol. We discuss how a MUP-enabled node discovers other such nodes in its vicinity. We discuss the mechanism that MUP uses for determining the best channel for communicating with a neighboring node. Finally, we discuss how MUP is resilient against node failures.

A. Design Rationale

When constructing a multi-hop wireless network using IEEE 802.11 hardware, typically one chooses a single ad hoc network, described by a unique Service Set Identifier (SSID) [4]. Unfortunately, even when multiple 802.11 NICs are present on the host, each NIC converges on the same physical channel. As a consequence, because of contention only one NIC is used at any given time.

MUP "unifies" multiple radios such that frequency (channel) diversity is achieved while ensuring that all nodes are part of the same logical ad-hoc network. Furthermore, it provides a mechanism by which these nodes can make sensible decisions about which channel to use when communicating with immediate neighbors in the network, in a way that attempts to reduce interference and thus improve the overall capacity of a multihop wireless network.

1) Architecture

MUP is implemented at the link layer, so that network traffic can make use of the multiple interfaces without any modification to applications or to the upper layers of the network protocol stack. To hide the complexity of multiple network interfaces from applications and from the upper layers of the protocol stack, MUP exposes a single *virtual MAC address* in place of the multiple physical MAC addresses used by the wireless Network Interface Cards (NICs). Thus, we describe MUP as a multi-radio unification protocol, because from the application perspective the system operates as if there is only a single wireless network interface.

2) Wireless NIC to Channel Binding

MUP works on any node that has two or more wireless network interfaces. **At startup, the network interfaces a node are tuned to orthogonal channels. The channel assignment is hard-coded, and once the channel is assigned to the NIC it does not change**. Therefore, when the underlying 802.11 standard supports N orthogonal channels, MUP can provide performance benefits for nodes that have anywhere from 2 up to N wireless NICs. **In the rest of the paper, wherever we talk about switching to a channel or selecting a channel, we mean choosing the wireless network interface that operates on that channel**. In Section VI, we provide experimental analysis of the number of available orthogonal channels for IEEE 802.11a and IEEE 802.11b network cards.

3) Which Wireless NIC to Use?

The basic functionality provided by the MUP layer is a means of deciding which NIC to use, and therefore which channel to use, when communicating with a neighboring node. A naive approach would be simply to choose a channel at random; we refer to this approach as *MUP-Random*. By making use of multiple channels, this simple approach has the potential to reduce contention and thereby increase the overall network capacity. However, with MUP-random, a node might choose a channel currently used by a nearby node over an available idle channel. Thus, a key design goal for MUP is to find the best available channel for communication, based on current conditions of each channel.

MUP uses an abstraction that we call the *channel quality metric* to characterize the recent conditions of each channel. Our current technique for estimating channel quality is to send probe messages across each channel on a periodic basis, and then to measure the round-trip latency of these messages. For each neighboring node, a node computes its channel quality metric *independent* of its neighbors' decision. Independent channel selection simplifies the protocol design because no agreement is required between the sender and the receiver on which channel to use.

4) Local versus Global Decisions

MUP makes a decision about which channel to use for communication between a pair of nodes based on local information about channel quality. It is easy to see that such local optimization may not lead to a globally optimal allocation of channels. However, even with perfect knowledge of the traffic pattern, network topology, and interference pattern, the global optimization problem is believed to be NP-complete. **More importantly, it is also believed that solving the corresponding approximation problem is also hard.** In other words, for this problem, it is expensive to find a solution that is within a given factor of the optimal solution [27]. Finally, we view it as the job of the routing protocol to adapt to long-term global changes whereas the goal of MUP is to rapidly adjust to changes in local conditions.

In the remainder of this section, we describe in detail the two major components of MUP: neighbor discovery and classification, and the communication protocol between MUP-capable nodes.

Field	*Description*
Neighbor	IP address of the neighbor host.
Status	Indicates whether this neighbor is known to be MUP-capable or not.
MAC list	An array of MAC addresses associated with this IP address.
Quality list	An array of the channel quality values associated with each destination MAC address.
Channel	Current preferred channel to communicate with this host.
Selection time	Last time a channel selection decision was made (based on channel quality values).
Packet time	Last time a packet was either sent or received from this host.
Probe time list	List of times for unacknowledged probe messages

TABLE I: Summary of an Entry in the MUP Neighbor Table

B. Neighbor Discovery and Classification

The MUP implementation maintains a table of information about neighboring nodes. We refer to this table as the *MUP neighbor table*. A node uses this table to keep track of which nodes it has communicated with, and which of those nodes are MUP-capable. It also stores the per-interface MAC addresses, as well as the corresponding channel quality and channel selection information for each neighbor. Table I gives a high level description of the information maintained in the MUP layer for each neighbor.

When a MUP-enabled host first initiates communication with a neighboring host, it does not assume that the neighbor is MUP-capable. Therefore, communication is initiated using the ARP protocol [28]: an ARP request is broadcast over all the interfaces. If MUP receives an outgoing packet with a link-layer broadcast destination address, it broadcasts the packet over all wireless interfaces. All incoming ARP messages pass through the MUP layer, and MUP records any MAC address information in these messages. When the destination node receives ARP requests, it sends out ARP responses, ensuring that the MAC address contained in the ARP response corresponds to the network interface that it received the ARP request on. Once the originating host receives any of the ARP responses, it can begin communicating using the interface on which the response was received. If any additional ARP responses are received, MUP also records those additional MAC addresses. In summary, ARP is used as the first step of communication between nodes, which ensures that MUP-enabled nodes behave properly when communicating with legacy nodes that do not support MUP.

After the initial ARP, a MUP-enabled node also initiates the MUP discovery process to determine if the neighbor is MUP-enabled. Note that ARP responses for more than one network interface may have already been received, but the existence of multiple interfaces on a node does not mean that the node supports MUP. To determine whether or not the remote node supports MUP, a MUP "CS" (aka *Channel Select*) message is sent across all resolved interfaces. A MUP-enabled node will respond with a "CS-ACK" message (aka *Channel Select Acknowledgment*), whereas a legacy node will not. Timeouts are used to retransmit CS messages if necessary. After a certain number of failed retransmissions, the neighboring node is classified as a legacy node. Entries in the MUP neighbor table are deleted if no traffic is exchanged with that neighbor for a long period of time. The discovery and classification process described above is used when the next communication is initiated with that neighbor.

When a machine A determines that a neighboring machine B is MUP-enabled but A has not discovered all the MAC addresses of B, the MUP layer on machine A explicitly attempts to resolve the MAC addresses on the remaining channels. It does this by sending ARP requests along all of the radios where the destination MAC addresses are not yet known. As before, a timeout mechanism is used for retransmitting ARP requests. After a fixed number of failed retransmissions, the particular radio where the timeout occurred is considered disabled when communicating with that neighbor. This situation is most likely to arise when the radios have different ranges, but it could also occur due to external interference on certain channels.

C. Steady-state MUP Communication

When two MUP-capable nodes communicate, they periodically test the quality of all channels available to them. Also on a periodic basis, they decide which channel to communicate over for the next time period, based on their estimate of recent channel quality.

1) Selecting the Communication Channel

Of all the NICs available to it, MUP selects the NIC with the best channel quality. The basic technique used to estimate channel quality is to send probe messages over each channel and measure the round-trip time of the probes. The motivation for using round-trip latency is that probe messages sent on a heavily-utilized channel are likely to take longer to gain access to the medium than probes sent on a lightly-used channel. Further, since external conditions such as interference from microwaves and portable phones reduce the likelihood of the probe messages and probe ACKs getting through, the round-trip times increase (or a timeout occurs in MUP) when these conditions exist.

As we discovered through experience with our implementation, queuing delay for the probe packets can be a significant issue when the node that is sending the probe is also sending large quantities of data. To resolve this issue, we require that the network interfaces used by MUP allow probes packets to be sent at a higher priority or to be placed at the head of the NIC transmit queue. Such support is available as part of the 802.11e draft standard [5], and 802.11e hardware is expected to be available within a few months.

The 802.11e standard allows 8 separate priority queues per station. The station chooses the priorities for each traffic category. Each station runs an independent 802.11 MAC protocol for each priority-level, and this protocol includes a priority-specific backoff component. By scheduling the probes over a high priority queue, we can reduce or eliminate the queueing delay problems discussed above.

Once a channel is selected, MUP sticks with it for a significant time period (on the order of seconds). This is to balance the overhead of measurement traffic with the agility of the protocol to adjust to changing network conditions.

Parameter	Description
α	Weighting factor in Eq. 1
T_{cs}	‘CS” message period.
(T_{Dmin},T_{Dmax})	Randomized interval for deciding which channel to use.
T_{gc}	Minimum idle period before a node is deleted from the table.
p	Percentage improvement required to switch channels.

TABLE II: MUP Configuration Parameters

To calculate the channel quality of a given channel, a node sends CS messages on a fixed periodic basis. A typical value for this time period would be every half a second. When a node receives a CS message, it immediately responds with a CS-ACK message. When the sending node receives the CS-ACK message, it incorporates the round-trip time (RTT) measurement into a weighted average called smoothed RTT (SRTT) as follows:

$$SRTT = \alpha * RTT_{new} + (1 - \alpha) * SRTT \qquad (1)$$

This weighted average is used as the channel quality estimate. MUP does not use SRTT to make any fine-grained calculations about channel loads, so a rough estimate indicating that one channel is carrying significantly more traffic than the other is sufficient.

In some cases, either the CS or the CS-ACK message may be lost entirely. MUP detects lost messages in one of two ways. The CS and CS-ACK messages contain matching sequence numbers, so the end nodes can detect when a CS-ACK arrives in the wrong order. In this case, MUP assumes that all out-of-order CS messages were lost. Further, in the case where no CS-ACK messages arrive at the sender, after a time period of 3 times the current SRTT estimate, the protocol decides that the CS is lost. For each lost probe, the protocol assigns a packet loss penalty of 3 times the current SRTT estimate.

2) Switching Interfaces/Channels

MUP uses a randomized time interval to decide when to change the selected channel for each neighbor. Typical values for this interval are in the range of 10 to 20 seconds. The interval is randomized to avoid synchronized switching across a set of nodes. The decision of whether or not to switch channels is based on the SRTT estimate described above. The channel quality values are compared across all available channels, and the channel with the best quality is selected if it provides a certain percentage improvement over the currently selected channel. A typical value for this threshold is 10%. If the improvement is less than the threshold, then the current channel remains selected.

Table II provides a summary of all the tunable parameters described above that potentially affect either the overhead or the performance of MUP.

Once the decision to switch to a different channel has been made, the node immediately begins sending outgoing packets over the newly selected interface. Thus, there is a possibility that immediately after a channel switch, some packets being sent over the newly selected interface will depart the host before some of older packets that were queued to go via the old interface. Such re-ordering of packets can be detrimental to TCP performance, as a burst of three or more packets arriving out of order causes TCP to halve its congestion window.

We considered an alternative design where, once the decision to switch channels has been made, a node stops sending data via the old interface and queues up new data until the old interface queue drains. This solution has the benefit that it reduces packet re-ordering. However, we discard this solution for two reasons. First, it introduces what may potentially be a significant delay. Presumably, the channel switch decision was made because conditions on the old channel became poor; thus, it may take a long time for queue on the old interface to drain. The second reason is that, there is a reasonable chance that the packet re-ordering will not cause TCP to halve the congestion window: a likely form of reordering is that packets going via the old and new interfaces will be interleaved. Thus, as long as the interleaving of packets does not cause three duplicate acknowledgments at the TCP level, TCP performance will not be significantly harmed.

D. Handling Failures

MUP is resilient to node failures. The information stored in the MUP neighbor table is all soft-state, so when a node crashes and then reboots, its MUP neighbor table starts out empty. On an as-needed basis, the recovered node simply performs the neighbor discovery and classification steps described in Section IV-B for each neighbor that it needs to communicate with. Because each MUP-enabled sender makes its decision independently about which channel to use, there is no possibility that a node crashing and losing its neighbor table will lead to any sort of inconsistent behavior.

V. IMPLEMENTATION

We have implemented MUP as a kernel level driver in Windows XP. This driver is a Windows NDIS intermediate driver [29] that sits under the networking layer but above the link layer. Due to space constraints, we provide only a high-level description of our implementation. The driver performs multiplexing across multiple physical interfaces for packet sends and demultiplexing across the interfaces for packet reception to give the appearance of single network interface and MAC address to the upper layer protocols and applications. In the Windows operating system, this approach requires the driver to act both as a miniport driver and as a protocol driver. MUP-enabled nodes periodically exchange CS and CS-ACK messages to test channel quality, via this driver. These messages sent in Ethernet packets whose type field is set to a special value. When an ethernet packet of this special type is received by a legacy (non-MUP capable) node, the network stack silently discards the packet. At MUP-capable nodes, such packets are intercepted and processed by the MUP driver.

VI. INTERFERENCE EXPERIMENTS

In order to deploy MUP and to determine the number of radios to use at each node, one needs to understand the number of available orthogonal channels. Table III shows the spectrum and channelization structure of the 802.11a, 802.11b, and 802.11g standards, along with the number of channels that are theoretically orthogonal. From this table, it appears that 802.11b and 802.11g have three orthogonal channels, while 802.11a has thirteen orthogonal channels. In theory, radios that operate on non-overlapping channels should not interfere with each other. In practice, due to signal power leakage, radios that are physically close to each other may interfere even while operating on non-overlapping channels. When building a multi-hop network with forwarding nodes that have multiple radios, the radios attached to a node will be physically close to each other. The following experiments investigate the impact of interference caused by signal power leakage, to determine the number of available orthogonal channels for both 802.11a and 802.11b, when the radios are in close proximity.

Standard	Frequency Range (GHz)	Orthogonal Channels	Channel Width (MHz)
IEEE 802.11a	5.15-5.35, 5.725-5.850	13	20
IEEE 802.11b,g	2.400-2.4835	3	22

TABLE III: Spectrum and channels over which the IEEE 802.11 standards operate in the United States

We begin with an overview of our experimental setup, and then we summarize our results. We used three different configurations for the experiments reported here: Netgear WAB501 cards in 802.11a mode; Netgear WAB501 cards in 802.11b mode; and Cisco Aironet 340 802.11b cards. The surrounding environment was pristine for 802.11a, whereas for 802.11b the nodes were within range of a number of corporate access points. We monitored the load generated by the infrastructure network using Airopeek [30], and performed our experiments when the infrastructure network was basically idle. We performed three independent trials and we report the mean values.

We were prevented from using the Netgear cards in true multi-hop configuration, due to device driver problems. Therefore, we emulate a multi-hop configuration using four nodes (labeled A, B, C, and D), where all nodes are within communication range of each other, and nodes B and C in close proximity to each other. For the Netgear cards, a separation of 6 inches between the cards on nodes B and C created significant interference. For the Cisco cards, they only appear to generate interference in the vertical plane, so we placed the laptops on top of each other – the resulting separation was 3 inches.

During the experiment we tune each hop (A-B, and C-D) to a specific channel. Then node A initiates a bulk TCP transfer to node B, and simultaneously C initiates another transfer to D. We repeat the experiment for various channel assignments, and observe the impact on the throughput of the two TCP connections.

Figures 3 and 4 show the results of these experiments. In Figure 3, we see that the interference behavior of the Cisco cards is very different from that of the Netgear cards. For the Netgear cards, channels 1 and 6, 6 and 11, and 1 and 11 all interfere with each other. Therefore, for a multi-hop configuration there is only one orthogonal 802.11b channel.

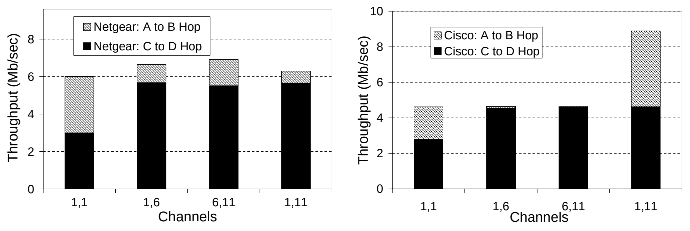

Fig. 3: Interference for 802.11b using Cisco and Netgear adapters

For the Cisco cards, we observe interference between channels 1 and 6 and between 6 and 11, but not between 1 and 11. Therefore, the Cisco cards provide two orthogonal 802.11b channels in a multi-hop configuration. In Figure 4, we see that adjacent 802.11a channels interfere (e.g. 60 and 64), but non-adjacent channels (e.g. 56 and 64, 52 and 64) do not interfere. Therefore, the Netgear cards provide seven orthogonal channels in a multi-hop configuration: 36, 44, 52, and 60 in the low band; 149, 157, and 165 in the high band. We also performed the 802.11b experiments with a physical separation of over 1 foot between nodes B and C. In these experiments we see no interference for either brand of cards between channels 1 and 6, 6 and 11, and 1 and 11.

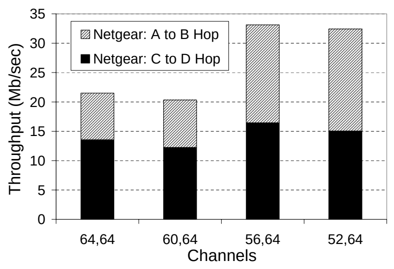

Fig. 4: Interference for 802.11a, using Netgear adapters.

The main conclusion of these experiments is: in a multi-hop configuration, interference between radios on forwarding nodes may significantly degrade throughput. However, the extent of the interference appears to be dependent on the specific hardware chosen, so the number of truly orthogonal channels must be determined experimentally. Finally, building custom hardware for a multi-hop forwarding node may allow the designer to place the radios far enough apart that interference is not a serious problem.

VII. Protocol Evaluation

We study the performance of MUP using a combination of our kernel implementation and simulations. Because we don't yet have IEEE 802.11e hardware, the bulk of our evaluation is performed with simulations.

A. *Implementation results*

We begin by a simple experiment that shows the channel selection behavior of our implementation. We follow this with an investigation of queuing delay on our prototype that demonstrates the need for IEEE 802.11e hardware support.

1) *Illustration of channel switching*

To illustrate the channel switching behavior of MUP, we ran a simple experiment with 4 machines named A, B, C and D. Each machine is equipped with two IEEE 802.11b NICs (Cisco 340). On each node, one NIC is tuned to Channel 1 and the other to channel 11. All machines are within a few feet of each other. Machines A and B are MUP-enabled, whereas C and D are legacy machines. Since C and D are not MUP capable, we manually assign separate IP addresses to their NICs such that NICs tuned to channel 1 are on the same subnet, and NICs tuned to channel 11 are on a different subnet. This is illustrated in Figure 5.

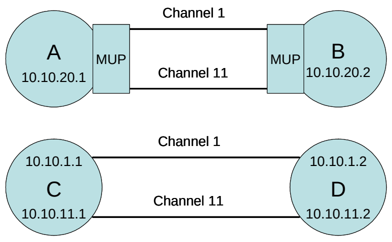

Fig. 5: Experimental setup to illustrate channel switching

Throughout this experiment, A sends CBR traffic to B at the rate of 50Kbps. The results of the experiment are shown in Figure 6. The top graph shows the SRTT measured by node A to B, on channel 1. The middle graph shows the SRTT on channel 11. The bottom graph shows the channel being used by A to send data to B.

Just before time 50, A is using channel 11 to send data to B. At time 50, C starts a large TCP transfer to D using NICs tuned to channel 11. This results in increased contention on channel 11, as evinced by the increased SRTT on that channel. As a result, after a short delay imposed by the hysteresis mechanism, A switches to channel 1 to send data to B. The TCP transfer between C and D ends around time 125. The contention on channel 11 subsides, as evinced by the drop in SRTT. At time 130, we start a new TCP transfer between C and D on channel 1. The contention, and hence the SRTT on channel 1 increases. After hysteresis delay, A switches back to channel 11 to send data to B.

These results show that our MUP driver chooses channels effectively according to the observed load using its probe latency mechanism.

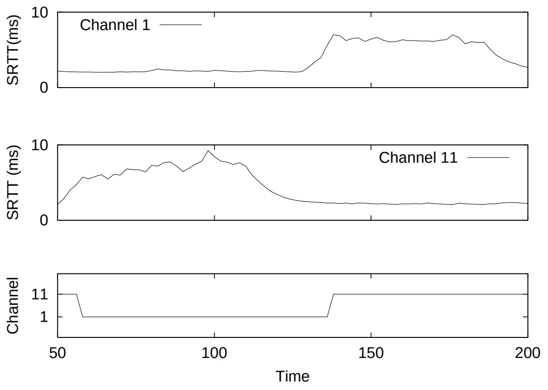

Fig. 6: Channel Switching

2) Impact of queuing

In the previous section, we showed an illustrative example of MUP's channel switching behavior. Note however, that there was very little traffic between A and B - only 50Kbps. This traffic is not sufficient to cause queuing at the network interfaces or in other buffers in the kernel. To investigate the impact of queuing delays on measured RTTs, we conducted the following experiment.

The experimental setup was the same as in the previous section. However, we modified the implementation slightly, so that A and B always used channel 1 to send data to each other, regardless of SRTT values. At time 150, A starts a large TCP transfer to B. At the same time, C starts a large data transfer to D on channel 11.

In Figure 7, we show the SRTT measured by A to B on both the channels. As we can see, the SRTT on channel 1 is significantly higher than the SRTT on channel 11. This is despite the fact that the traffic volume on both channels is approximately equal. The reason for large SRTT value on channel 1 is that the CS packets get queued behind the TCP data packets on node A.

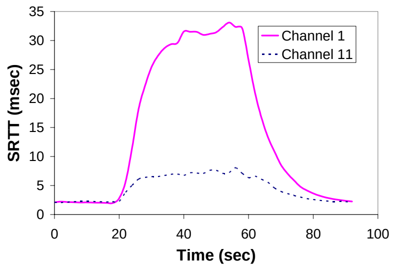

Fig. 7: Impact of queuing

The results show that the impact of queuing on SRTT is significantly higher than the impact of contention. To negate the impact of queuing, we have to be able to insert the CS and CS-ACK packets at the head of the queue, by assigning higher priority to them. It is possible to do this with the new IEEE 802.11e hardware. However, this new hardware is just coming to the market, and we were unable to get it in time for our experiments. Thus, in the rest of the paper, we assume this ability, and rely on simulations to evaluate the performance.

B. Simulation Results

For the purpose of this study, we have implemented MUP in the NS [31] simulator. For all the simulations discussed in this section, we use omni-directional antennas. We slightly modify the wireless node model in NS to assign higher priority to probe packets. We make no changes to the model of the physical channel, and the model of the IEEE 802.11 MAC. MUP is implemented just above the MAC layer, by modifying the code that handles link-layer forwarding and ARP.

We begin by showing that the one-hop round trip time is a good measure of load on a wireless channel. Next, we consider a scenario in which MUP-enabled hosts operate together with legacy hosts. We then perform MUP parameter sensitivity analysis. We also show that when several MUP-capable hosts

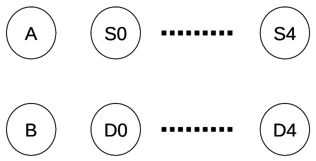

Fig. 8: Topology for RTT measurements

operate together, they do not flip-flop between the channels in a synchronized fashion. Finally, we consider a complex scenario based on a community mesh network. Using the topology data from a real neighborhood, we show that MUP significantly reduces the delay experienced by web users.

1) RTT is a reasonable measure of load

MUP relies on one-hop RTT measurements between a pair of hosts to determine the quality of channel between those hosts. MUP uses a smoothed RTT (SRTT) as described in Section IV, to distinguish between the quality of available channels. In this section, we only consider the impact of load on the quality of the channel.

We consider the 12 node topology shown in Figure 8, which consists of several more nodes. Each node is located within the communication distance of others. *None of the nodes are MUP-capable. There is only one wireless channel available, and all nodes are tuned to it.*

Node A pings node B every 0.5 seconds, and uses these samples to compute SRTT using the low-pass filtering mechanism described in Section IV.

The rest of the nodes are labeled $S_0 \ldots S_4$ and $D_0 \ldots D_4$. Node S_i sends CBR traffic to node D_i at 200Kbps. Node S_0 starts sending at time 50, and stops at time 500. Node S_1 starts at 100 and stops at 450, and so on. Thus, the number of active senders varies with time as shown in the bottom half of Figure 9. The top half of the Figure shows the SRTT value computed by A, using $\alpha = 0.1$. The results show that in this scenario SRTT is a good indication of the load on the channel. In Figure 10, we plot the same result, but with different values of α. As one might expect, SRTT is smoother as the value of α decreases. However, even with $\alpha = 0.5$, SRTT is a reasonable indicator of load on the channel.

We repeated the experiments using TCP connections between S_i and D_i and observed that SRTT is a reasonable estimator of load on the channel.

While previous scenarios illustrate how SRTT changes with offered load, the load change occurred at a coarse granularity, i.e., once every 50 seconds. A significant portion of current Internet traffic is made of short web transfers, generating bursty traffic. We now examine how SRTT performs as a measure of offered load in such a scenario. We use the same topology as before. We place five web servers at nodes $S_0 \ldots S_4$. Five clients are located at nodes $D_0 \ldots D_4$. The clients download web pages from the web servers using the web traffic model prescribed in [31]. Node D_0 starts downloading pages at time 0, and nodes $D_1 \ldots D_4$ join in at 50 second intervals. The bottom half of Figure 11 shows the number of client sessions active at any time. The top half shows the SRTT, measured by node A using $\alpha = 0.1$. As expected, the SRTT value

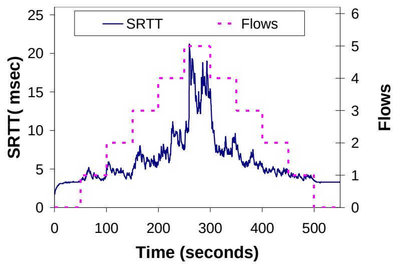

Fig. 9: SRTT: CBR traffic, $\alpha = 0.1$

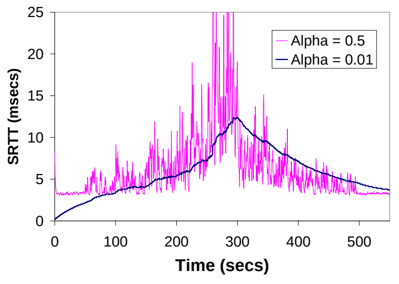

Fig. 10: SRTT: CBR traffic, $\alpha = 0.01, 0.5$

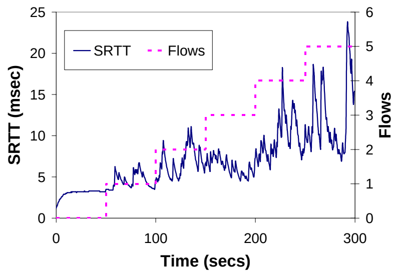

Fig. 11: SRTT: Web traffic, $\alpha = 0.1$

varies significantly, compared to previous scenarios of long-lived flows. However, we note that the value appears to be increasing with the number of active sessions, thus providing a coarse indicator of channel load. As discussed earlier, MUP does not use the SRTT value for fine-grained computations of channel load - MUP merely uses it to check if one channel is significantly more loaded than the other.

2) *Benefits of intelligent channel selection*

In this section, we consider a scenario in which MUP-enabled hosts operate together with legacy hosts. The results are based on the 16-node grid topology shown in Figure 12. The nodes are stationary, and are spaced 200 meters apart from each other. AODV [32] is used for routing among the nodes.

The traffic pattern in this network is as follows. An FTP session is established between nodes S and D, which are in the opposite corners of the grid. Since the nominal range of 802.11 PHY modeled in NS is approximately 250 meters, this session has to travel over multiple hops. The FTP connection runs over TCP, and it always has data to send. In addition to the TCP traffic, there are 4 UDP flows in the network, whose sources and destinations are selected at random at the start of the simulation. Each UDP flow independently oscillates between an ON period and an OFF period. During the ON period, the sender of the UDP flow sends data to the receiver at 50Kbps. Duration of successive ON periods are independently drawn from a Pareto distribution with a mean of 2 seconds, and shape of 1.2. During the OFF period, the UDP flow transmits no data. The duration of successive OFF periods are also independently drawn from a Pareto distribution with mean of *off* seconds, and shape of 1.2. We vary *off* to generate different levels of UDP traffic. We define the intensity of UDP traffic to be the ratio between the mean ON period, and the mean OFF period. For example, if *off* $= 1$, then we say that the traffic intensity is 2. If *off* $= 4$, then the traffic intensity is 0.5. Compared to

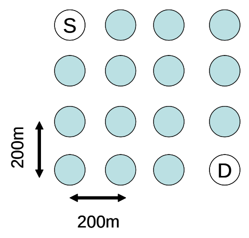

Fig. 12: Topology to demonstrate benefits of intelligent channel selection

the steady traffic used for some of the previous scenarios, the traffic in this scenario is more dynamic. Such ON-OFF traffic has been used in other studies [33] as well.

The goal in this scenario is to measure the improvement in the performance of the TCP connection as a result of using MUP, under different levels of background UDP traffic intensity.

To establish the baseline case, we set all 16 nodes to be legacy nodes, operating on the same channel, C_0. Then, we start the TCP and the UDP flows, and measure the throughput of the TCP connection over a period of 300 seconds. Next, we randomly select half of the 16 nodes to be MUP capable. MUP-capable hosts can communicate on channels C_0 and C_1, which are orthogonal, but identical in all other respects. The remaining hosts are legacy nodes that can comminuate only on channel C_0. MUP–capable nodes use parameters shown in Table IV. We once again measure the throughput of the TCP connection. Finally, we repeat the experiment again, by using MUP-Random, instead of MUP on MUP-capable nodes. MUP-Random switches channels at random, as described in Section IV.

Parameter	Value
α	0.1
T_{cs}	0.5 seconds
T_{Dmin}	10 seconds
T_{Dmax}	20 seconds
p	10%

TABLE IV: Parameter settings

In Figure 13, we plot the improvement in the throughput of the TCP connection, measured as a percentage of the baseline case (i.e., the case where all nodes are legacy nodes), for both MUP and MUP-random. The improvement is plotted against values of UDP traffic intensity. The numbers are averaged over 5 runs. The results show that both MUP-random and MUP provide significant improvements in TCP throughput. The gains provided by MUP are significantly higher than MUP-random, and tend to increase with increasing traffic intensity.

Since MUP-Random switches channels randomly, i.e., without any consideration to the quality of the available channels, we argue that most of the gains provided by MUP-random come simply from the fact that two channels provide additional data-carrying capacity. On the other hand, MUP takes quality of the channels into account, while making the selection. Since half the nodes in this scenario operate only on one channel, intelligent channel selection provides better performance than

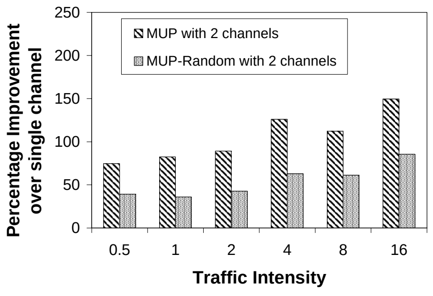

Fig. 13: Benefits of intelligent channel selection in presence of legacy nodes

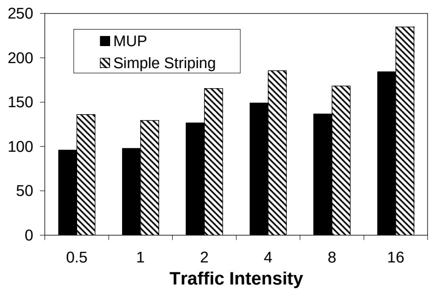

Fig. 14: Simple striping in absence of legacy nodes

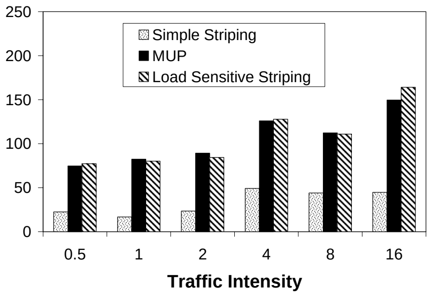

Fig. 15: Simple and load-sensitive striping in presence of legacy nodes

switching channels at random. Thus, the results show that the gains provided by MUP are not only due to the increased capacity provided by additional channels, but also due to the intelligent channel selection.

3) Comparison of MUP with Striping

The multi-radio design allows us to send and receive packets simultaneously. However, at any given time, MUP uses only one network interface to send packets to its neighbor. The reader might wonder why we do not use all the network interfaces to send packets, i.e. why we do not incorporate striping into our protocol. We now compare the performance of MUP with different striping protocols and show that the gains due to striping are very sensitive to the workload and environmental conditions.

We consider the following simple striping algorithm. Whenever a node can talk to a neighbor over more than one interface, it uses the interfaces in a round-robin fashion to send packets to that neighbor. The round robin algorithm operates on a per-neighbor basis, a node may use the same interface to send successive packets if they are destined to different neighbors.

To evaluate the performance of this simple striping algorithm in comparison with MUP, we once again consider the simulation setup discussed in the previous section. The baseline scenario remains the same: all nodes have only one radio. However, to evaluate MUP and striping, we assume that all 16 nodes have two radios. This is different from the previous section, where we assumed that only half the nodes had two radios. All other simulation settings are the same. We use the same performance metric as before - improvement in the throughput of the corner-to-corner FTP connection, in presence of on-off UDP traffic.

We might expect that the simple striping algorithm to perform poorly, since TCP packets might get reordered, and the throughput of the FTP connection will suffer. However, it turns out that this is not the case, and striping performs better than MUP in this scenario. This is because both the channels are equally loaded, and reordering of TCP packets is minimal. Since TCP *does* tolerate upto 3 out of order packets before reducing its contention window, the performance of the TCP connection does not suffer. This is illustrated in Figure 14. The graph shows that striping provides significant performance gains not only over one-radio baseline, but also over MUP.

However, when we go back to the original scenario, in which half the nodes have only one radio, we find that the simple striping algorithm performs poorly compared to MUP. In this setting, the load on the two channels is unequal. Therefore, under the simple striping algorithm, reordering of TCP packets is more severe than previous case. This results in poor performance, as shown in Figure 15.

We have designed a more sophisticated versions of the basic striping algorithm, which take into account the load on the two channels as measured by SRTT. The algorithm then stripes only when the two channels are deemed equivalent. Performance of one such scheme is also shown in Figure 15. In this scheme, a node stripes data to a given neighbor over two radios, as long as the SRTT to that neighbor over both the radios is within 10% of each other. At all other times, the nodes run the MUP protocol. As we see, the performance of this hybrid intelligent striping scheme is much better than the simple striping algorithm, but very similar to that of MUP.

We considered several other striping schemes but none of them provided significant additional benefits over MUP when the channels were unequally loaded. Moreover, we found that the performance of such striping schemes is very sensitive to the parameter values. With incorrect parameter values, the performance quickly degenerate to levels comparable to simple striping. In future, we plan to investigate adaptive mechanisms for hybrid striping schemes. However, for the rest of this paper, we focus only on MUP.

4) Analysis of parameter sensitivity

To evaluate the sensitivity of the MUP's performance to values of various parameters, we run the simulation described in Section VII-B.2, with all nodes being MUP capable, in different parameter settings. We use the settings in Table IV as the base parameter settings and vary one parameter at a time for each set of simulations.

Table V reports the relative performance for the MUP protocol with respect to the performance in the the base parameter setting.

The calculation of the entries in the Table is best explained by an example. Consider the value when $\alpha = 0.9$ and $off = 2$. This value is obtained as follows. Recall that the metric of performance in Section VII-B.2 is the throughput of the corner-to-corner TCP connection. We carry out a simulation for $off = 2$, and with all parameters set to their value in Table IV. Let us assume that the throughput of the TCP connection in this simulation is X. We then repeat the simulation, except that we set $\alpha = 0.9$. Let the throughput

	$off = 1$	$off = 2$	$off = 4$
$\alpha = 0.1$	1.00	1.00	1.00
$\alpha = 0.5$	0.97	0.98	0.98
$\alpha = 0.9$	0.98	0.98	1.00
T_{cs}=0.25 sec	0.91	0.86	0.92
T_{cs}=0.5 sec	1.00	1.00	1.00
T_{cs}=1 sec	1.02	1.02	1.06
T_{Dmin},T_{Dmax} = 5, 10 sec	0.97	0.99	1.02
T_{Dmin},T_{Dmax} = 10,20 sec	1.00	1.00	1.00
T_{Dmin},T_{Dmax} = 15,30 sec	0.98	0.89	0.97
p = 10%	1.00	1.00	1.00
p = 20%	1.02	0.99	0.99

TABLE V: Performance sensitivity with one varying parameter: relative performance with respect to base settings in Table IV

of the TCP connection in the second simulation be Y. Then $0.98 = Y/X$. Thus, the entries report the impact of changing a single parameter in the base setting shown in Table IV.

We have also done the same study by varying multiple parameters at the same time. The results are comparable to those reported here. *Based on this sensitivity study, we conclude that MUP is not very sensitive to parameter setting changes in the given simulations.*

5) Prevention of synchronized channel selection

MUP-enabled nodes make independent decisions about channel selection based only on locally available information about channel load. Thus, it is possible that many nodes will simultaneously detect that a given channel is busy, and switch to another channel. The second channel will now experience heavy loads, and after some time, all nodes may switch back to the first channel. Such synchronized behavior will negate any gains afforded by availability of multiple non-interfering channels. As described earlier, MUP incorporates several mechanisms to damp such synchronized channel oscillations. To illustrate the effectiveness of these mechanisms, we carried out the following simulation. The topology for the simulation is shown in Figure 16. The nodes are arranged in a 6x6 grid, spaced 200 meters apart. Each node is equipped with two wireless NICs, tuned to two orthogonal channels, viz. channel 0 and 1. The range of the radios is assumed to be 250 meters. All the sources are located at the bottom, and all the destinations are at the top. Each source sends a TCP flow to its corresponding destination. The TCP flows start within first five seconds of the simulation, and last for the entire duration. The total simulated time is 300 seconds. MUP parameters values are those shown in Table IV.

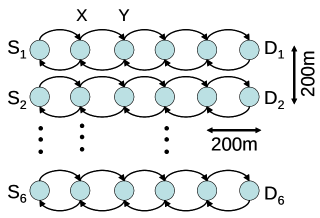

Fig. 16: Topology to illustrate de-synchronization mechanisms in MUP

In this scenario, each node can communicate with its horizontal and vertical neighbor. Hence, we may say that there are a total of 120 unidirectional wireless "links" in this scenario. Due to the traffic pattern, and the routing algorithm used (shortest path), only 60 of these 120 links are active during the simulation. These links are shown in the figure. For example, node S1 sends data to node X (TCP packets on their way to D1), and node X also sends data to S1 (TCP ACKs from D1 on their way to S1). Thus, there are two active links between S1 and X. On the other hand, S1 never sends or receives any data from S2, so there are no active links between S1 and S2.

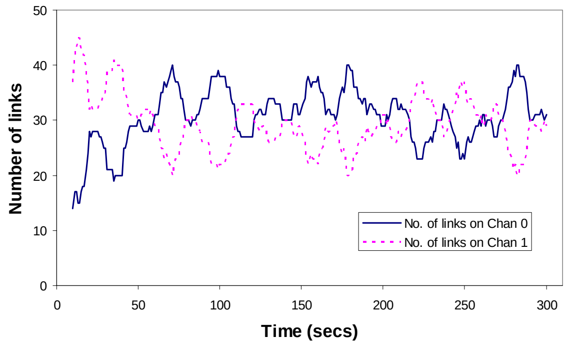

Fig. 17: Effectiveness of de-synchronization mechanisms in MUP

For each link, MUP makes a decision about which channel to use at the originating node, based on observed local conditions. So, for example, at a given time instance, the two links between S1 and X may operate on two different channels, or they might not. If the de-synchronization mechanisms in MUP are effective, the number of links using channel 0 and 1 will be approximately equal at any given time. The graph in Figure 17 show that these mechanisms are indeed effective, and the number of links on two channels is roughly equal. We further illustrate this point in Figure 18. The figure shows the channels occupied by two links, from S1 to X and from X to Y, over the duration of the simulation. It is seen that the channels to not change in synchrony with each other.

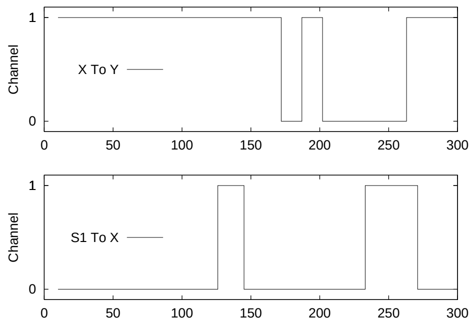

Fig. 18: Channels occupied by links (i) from S1 to X, and (ii) from X to Y

6) Web traffic in a real topology

In the previous section, we have considered performance of MUP in simple topologies. We now consider a more complex and realistic topology. We gathered data about positions of houses in Greenlake, a suburb of Seattle. In this section, we consider a 1000m$\times$1000m area of that neighborhood, as shown in Figure 19(a). There are 252 houses in this area. We select 35 of these houses at random, and assume that

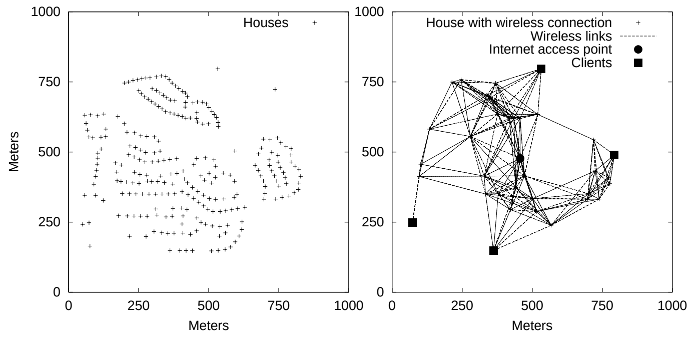

Fig. 19: (a) Houses in a suburban neighborhood (b) Formation of a mesh network in the suburban neighborhood among 35 randomly selected houses

these houses have decided to join a community wireless mesh network to share an Internet connection. The connectivity pattern formed by these houses, assuming a communication range of 250 meters is shown in Figure 19(b). The figure also shows that we have selected a house that is approximately at the center of the topology to serve as the Internet gateway. We assume that AODV is used for routing. We assume that the nominal throughput of each channel is 2Mbps.

Parameter	Average	Distribution
Time between pages	2 sec.	Exponential
Objects per page	4	Constant
Object Size (KB)	8	Pareto (shape=1.2), rounded up

TABLE VI: Parameter settings for web traffic generation

To explore the performance improvement provided by MUP in such a setting, we begin by assuming that 4 of the 35 houses are surfing the web at some point in time. There is no other traffic in the network. These four houses are shown as web clients in Figure 19(b). We assume that the web server is located at the Internet access point. This ignores the impact of wide area Internet. However, since the main objective of our simulations is to study the performance of the wireless part of the network, we believe that this does not weaken our results. The web traffic for the four web clients follows the model supplied in [31]. The model is parameterized as shown in Table VI. We use HTTP 1.0 protocol for simplicity.

We consider three deployment scenarios. First, in the baseline case, we assume that all nodes are legacy nodes, and they operate on a single channel. We call this *Scenario I*. In the *Scenario II*, we assume that half of the nodes are MUP-capable and can communicate on two channels. Finally, in *Scenario III*, we assume that all nodes are MUP-capable, and can communicate on two channels. For each MUP-capable nodes, the parameter values are set as shown in Table IV. We simulate an interval of 20 minutes in which all clients are active.

Scenario	Average TCP throughput (Kbps)
I	106
II	136
III	180

TABLE VII: Average throughput

We consider two metrics of performance. The first metric is the throughput of individual HTTP transfers. Since we have used HTTP 1.0, each object is downloaded via a separate TCP connection. The average TCP throughput in each scenario is shown in Table VII. The results show that MUP significantly improves the average throughput of TCP connections. With full deployment, MUP provides 70% improvement in throughput. Even with partial deployment, the throughput improvement is almost 30%.

The second metric of interest is the user-perceived page latency. We define user-perceived page latency as the time between the first object on a page is requested to the time the last object on that page finishes download. This is an underestimate of the actual value, since the full value must include rendering delays. However, since rendering delays are not impacted by underlying network conditions, we believe that our definition is appropriate. In Figure 20, we plot the CDF of the user-perceived page latency. The results show that MUP significantly reduces the median user-perceived latency. The reduction in response time is over 40% all nodes are MUP-capable. However, even when only half the nodes are MUP-capable, the reduction in response time is over 20%.

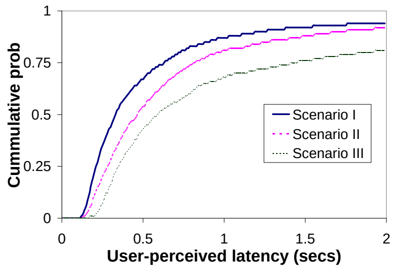

Fig. 20: CDF of User-perceived page latency

These results show that MUP performs well in complex topologies, and with complex traffic patterns. Moreover, the results indicate that it is possible to deploy MUP in an incremental fashion, as its benefits will start to accrue incrementally as well.

C. Summary

We summarize the conclusions of our performance analysis as follows: The simple experiments with our kernel implementation of MUP show:

- MUP chooses channels effectively according to the observed load.
- The queuing delay underscores the need for prioritizing CS and CS-ACK packets.

Our simulation results show:

- SRTT is a good measure of the load on the channel.
- Load-sensitive channel selection provides significant benefit.
- Under dynamic traffic patterns, and in realistic mesh topologies, MUP significantly improves the TCP throughput and user-perceived latency of web browsing.
- Striping does not provide consistent improvement over MUP in realistic settings.
- MUP performance is not very sensitive to the values of various tunable parameters.

In closing, since MUP can inter-operate with legacy nodes, and can be deployed incrementally, the performance benefits are also incrementally realized.

VIII. Future Work

We intend to carry this work forward along the following three directions.

- **Channel quality metric:**
  Researchers have investigated link stability and channel quality metrics such as average signal strength [34] and expected-throughput [35] in the context of path selection in routing protocols. We intend to evaluate and compare these metrics against our own SRTT metric. Further, we plan to investigate combination of these with SRTT to determine if channel quality estimation can be performed quickly and more accurately.
- **Node density:** If there are N MUP-capable nodes within communication distance of each other, The current version of the protocol will cause $O(N^2)$ CS and CS-ACK messages to be sent every probe interval. We believe that by using broadcast probe messages this overhead can be reduced to $O(N)$. We hope to prove this over the next few months.
- **Impact of mobility:** Although our motivating scenario of community neighborhood mesh networks does not require support for mobile routers, we would like to evaluate MUP for the mobile case as well. The main challenge with mobile nodes is to estimate the link quality quickly. We expect MUP's performance to be a function of the call to mobility ratio (CMR).

IX. Conclusions

We are interested in building an easy-to-deploy scalable multi-hop wireless network with existing IEEE 802.11 hardware. We want to build our network with a specific scenario in mind: community networking. In this scenario we are not constrained by the cost of radios or their power consumption, because radio costs are plummeting and we can place the routers in the neighborhood houses. In this paper, we describe a protocol that helps us realize these goals. We summarize the contributions of this paper as follows.

• We show that, to build a high capacity multi-hop wireless network, we must use the entire available spectrum. Furthermore, we show that utilization of the available spectrum can be locally optimized with coordinated use of multiple standard-compliant wireless cards.

• We show that radios that are physically close interfere with one another even when operating over non-overlapping channels. Without enough channel separation, signal power leakage results in lowering their throughput and hence the overall network capacity.

• We describe MUP, the Multi-radio Unification Protocol, which coordinates the use of multiple IEEE 802.11 wireless LAN cards in a multi-radio multi-hop wireless network. MUP works with existing hardware and requires no changes to applications, transport, or routing protocols. MUP works correctly with legacy nodes, defined as nodes that have only one radio, and hence is incrementally deployable.

• We evaluate MUP using realistic topologies and traffic. We use a node topology derived from a real neighborhood in a city in the United States. Using this complex topology and web traffic patterns, we show that nodes in a MUP enabled multi-radio multi-hop network achieve 70% increase in throughput and 50% improvement in delay.

• We describe a kernel implementation MUP. We describe the experiments we carried out on this implementation which validate the fact that MUP is able to switch channels effectively according to the observed load. Our experience with the implementation also points to the need of prioritizing the probe packets.

Acknowledgments

We would like to thank Rich Draves and Brian Zill for helping us with the testbed and the experiments.

References

[1] Jinyang Li, Charles Blake, Douglas S. J. De Couto, Hu Imm Lee, and Robert Morris, "Capacity of ad hoc wireless networks," in *ACM Mobile Computing and Networking (MobiCom)*, 2001, pp. 61–69.
[2] S. Xu and T. Saadwi, "Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks," *IEEE Communications Magazine*, vol. 39, no. 6, June 2001.
[3] M. Gerla, R. Bagrodia, L. Zhang, K. Tang, and L.Wang, "Tcp over wireless multihop protocols: Simulation and experiments," in *Proceedings of the IEEE International Conference on Communications (ICC '99)*, June 1999.
[4] IEEE802.11b/D3.0, "Wireless LAN Medium Access Control(MAC) and Physical (PHY) Layer Specification: High Speed Physical Layer Extensions in the 2.4 GHz Band," 1999.
[5] IEEE802.11e/D3.0, "Draft Supplement to Standard for Telecommunications and Information Exchange Between Systems - LAN/MAN Specific Requirements - Pert 11: Medium Access Control (MAC) Enhancements for Quality of Service(QoS)," May 2002.
[6] L. P. Cox, C. D. Murray, and B. D. Noble, "Pastiche: Making backup cheap and easy," in *Symposium on Operating Systems Design and Implementation (OSDI)*, Boston, MA, Dec 2002.
[7] Mesh Networks Inc., "Mesh networks technology overview," http://www.meshnetworks.com.
[8] Nokia Networks, "Nokia rooftop wireless routing - white paper," http://www.americasnetwork.com/americasnetwork/data /article-brief/americasnetwork/412002/34898/article.pdf, 2001.
[9] Radiant Networks, "Reach for everyone: Meshworks," http://www.radiantnetworks.com/meshworks/mwlesummary.asp.
[10] Invisible Networks, "Cost-effective rollout of wireless broadband networks," http://www.invisible.uk.net/how/.
[11] Benjamin A. Chambers, "The grid roofnet: A rooftop adhoc wireless network," in *M.S. Thesis, MIT*, Cambridge, Massachusetts, June 2002.
[12] Z. J. Haas and J. Deng, "Dual busy tone multiple access (dbtma) - a multiple access control scheme for ad hoc networks," *IEEE Transactions on Communications*, vol. 50, no. 6, pp. 975–985, June 2002.
[13] A. Muir and J. J. Garcia-Luna-Aceves, "A channel access protocol for multihop wireless networks with multiple channels," in *Proceedings of IEEE ICC*, Atlanta, Georgia, June 7-11 1998.
[14] Asis Nasipuri and Samir R. Das, "A multichannel csma mac protocl for mobile multihop networks," in *Proceedings of IEEE WCNC*, New Orleans, September 1999.
[15] Jungmin So and Nitin H. Vaidya, "A multi-chanel MAC protocol for ad hoc wireless networks," Technical report, Dept. of Com. Science, Univ. of Illinois at Urbana-Champaign, Urbana-Champaign, IL, Jan 2003.
[16] Z. Tang and J. J. Garcia-Luna-Aceves, "Hop-reservation multiple access (HRMA) for ad-hoc networks," in *INFOCOM*, March 1999.
[17] A. Tzamaloukas and J.J. Garcia-Luna-Aceves, "A receiver-initiated collision-avoidance protocol for multi-channel networks," in *Proceedings of IEEE INFOCOM*, Anchorage, Alaska, 2001.

[18] M. Allman, H. Kruse, and S. Ostermann, "An application-level solution to TCP's satellite inefficiencies," in *Workshop on Satellite-Based Information Services (WOSBIS)*, Rye, NY, November 1996.

[19] T. Hacker and B. Athey, "The end-to-end performance effects of parallel TCP sockets on a lossy wide-area network," in *Proceedings of IEEE IPDPS*, Fort Lauderdale, FL, April 2002.

[20] H. Sivakumar, S. Bailey, and R. Grossman, "PSockets: The case for application-level network striping for data intensive applications using high speed wide area networks," in *Proceedings of IEEE Supercomputing (SC)*, Dallas, TX, November 2000.

[21] Hung-Yun Hsieh and Raghupathy Sivakumar, "A transport layer approach for achieving aggregate bandwidths on multi-homed mobile hosts," in *MOBICOM'02*, Atlanta, GA, September 2002, pp. 83–94.

[22] L. Magalhaes and R. Kravets, "Transport level mechanisms for bandwidth aggregation on mobile hosts," in *ICNP*, Riverside, CA, Nov 2001.

[23] H. Adiseshu, G. Parulkar, and G. Varghese, "A reliable and scalable striping protocol," in *SIGCOMM*.

[24] J. Duncanson, "Inverse multiplexing," *IEEE Communications Magazine*, vol. 32, no. 4, pp. 34–41, April 1994.

[25] C. B. Thaw and J. Smith, "Striping within the network subsystem," *IEEE Network Magazine*, vol. 9, no. 4, pp. 22–32, July 1995.

[26] A. Snoeren, "Adaptive inverse multiplexing for wide-area wireless networks," in *IEEE GLOBECOM*, Rio de Janeireo, Brazil, Dec 1999.

[27] Kamal Jain, Jitendra Padhye, Venkat Padmanabhan, and Lili Qiu, "Impact of interference on multi-hop wireless network performance," To appear in Proccedings of MOBICOM 2003.

[28] David C. Plummer, "An ethernet address resolution protocol," in *IETF RFC 826, Network Working Group*, November 1982.

[29] Microsoft Corporation, "Network driver interface specification," 2002.

[30] "Wildpackets airopeek," http://www.wildpackets.com.

[31] NS (Network Simulator), ," 1995, URL http://www.isi.edu/nsnam/ns/.

[32] Charles E. Perkins and Elizabeth M. Royer, "Ad hoc on-demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, February 1999.

[33] Sally Floyd, Mark Handley, Jitendra Padhye, and Joerg Widmer, "Equation-based congestion control for unicast applications," in *Proceedings of SIGCOMM 2000*, August 2000.

[34] R. Dube, C. D. Rais, K. Y. Wang, and S. K. Tripathi, "Signal stability based adaptive routing (ssa) for ad-hoc mobile networks," in *IEEE Personal Communications*, 1997, vol. 4, pp. 36–45.

[35] Douglas De Couto, Daniel Aguayo, John Bicket, and Robert Morris, "High-throughput path metric for multi-hop wireless routing," To appear in Proccedings of MOBICOM 2003.