ALGORITHMS AND PROTOCOLS FOR MULTI-CHANNEL WIRELESS NETWORKS

A Thesis
Presented to
The Academic Faculty

by

Sandeep Kakumanu

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
Dec 2011

ALGORITHMS AND PROTOCOLS FOR MULTI-CHANNEL WIRELESS NETWORKS

Approved by:

Professor Raghupathy Sivakumar, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Faramarz Fekri
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Ed Coyle
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Doug BLough
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Stephan Eidenbenz
Information Sciences (CCS-3)
Los Alamos National Laboratory

Date Approved: Dec 2011

To my mother,

who has always been by side,

during all the ups and downs in my life.

ACKNOWLEDGEMENTS

First, I would like to thank my adviser Prof. Raghupathy Sivakumar for his unwavering guidance and motivation. His high standards in clear thinking and effective communication are inspiring and will always help me stand in good stead. Second, I would like to thank Stephan Eidenbenz, my mentor for the summer of 2007 at the Los Alamos National Laboratory.

The GNAN research group gave me the opportunity to work with a group of talented and hard-working students. I would like to acknowledge Ram, Sriram, Cheng-Lin, Zhenyun for the valuable discussions which helped in every aspect of this work. I would also like to thank Yeonsik, and Yujie for their assistance and help.

Last but not the least, I would like to thank my mother(Raghava Kumari), my wife(Sravani), my brother(Venugopal), my sister-in-law(Jyothismavati), and my nephew and neice(Gautam and Asrita). I owe all my success to their love, support, sacrifice and encouragement. I can never thank them enough.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

A wireless channel is shared by all devices, in the vicinity, that are tuned to the channel, and at any given time, only one of the devices can transmit information. One way to overcome this limitation, in throughput capacity, is to use multiple orthogonal channels for different devices, that want to transmit information at the same time.

In this work, we consider the use of multiple orthogonal channels in wireless data networks. We explore algorithms and protocols for such multi-channel wireless networks under two broad categories of network-wide and link-level challenges. Towards handling the network-wide issues, we consider the channel assignment and routing issues in multi-channel wireless networks. We study both single radio and multi-radio multi-channel networks. For single radio multi-channel networks, we propose a new granularity for channel assignment, that we refer to as *component level channel assignment*. The strategy is relatively simple, and is characterized by several impressive practical advantages. For multi-radio multi-channel networks, we propose a joint routing and channel assignment protocol, known as *Lattice Routing*. The protocol manages channels of the radios, for the different nodes in the network, using information about current channel conditions, and adapts itself to varying traffic patterns, in order to efficiently use the multiple channels. Through ns2 based simulations, we show how both the protocols outperform other existing protocols for multi-channel networks under different network environments. Towards handling the link-level challenges, we identify the practical challenges in achieving a high data-rate wireless link across two devices using multiple off-the-shelf wireless radios. Given that the IEEE 802.11 a/g standards define 3 orthogonal wi-fi channels in the 2.4GHz band and 12 orthogonal wi-fi channels in the 5GHz band, we answer the following question: *"can*

a pair of devices each equipped with 15 wi-fi radios use all the available orthogonal channels to achieve a high data-rate link operating at 600Mbps?" Surprisingly, we find through experimental evaluation that the actual observed performance when using all fifteen orthogonal channels between two devices is a mere *91Mbps*. We identify the reasons behind the low performance and present Glia, a software only solution that effectively exercises all available radios. We prototype Glia and show using experimental evaluations that Glia helps achieve close to 600Mbps data-rate when using all possible wi-fi channels.

CHAPTER I

INTRODUCTION

1.1 Introduction

Wireless data networks have become ubiquitous over the last few years owing to the numerous advantages they provide over wired networks. The primary attraction towards wireless networks is the tether-less connectivity that these networks provide. Wireless networks have found applications in both civilian (Ex: wireless Local Area Networks or WLANs, Bluetooth devices, Cell-phones, and Sensor networks) and defense (Ex: Adhoc networks, and Sensor networks) domains. The focus of my research are multi-hop wireless networks that are a class of wireless data networks that need little to no infrastructure support. In spite of their impressive benefits, wireless multi-hop networks are severely limited in throughput capacity. The primary reason for this is the broadcast nature of wireless communication. A wireless channel is shared by all devices in the vicinity that are tuned to the channel. Although, at any given time, only one of the devices can transmit information. One way to overcome this limitation in throughput capacity is to use multiple orthogonal channels for different devices that want to transmit information at the same time. Two channels are considered orthogonal, if transmission of some information on one channel does not affect the transmission or reception of any information on the other channel and vice versa.

Wireless spectrum is divided in to multiple channels by industry standards for two main reasons: (a) to allow parallel utilization of the spectrum by multiple wireless technologies at the same time, and (b) the design of wideband wireless transceivers is very complex because of the frequency dependent components involved in the design.

For example, the IEEE 802.11a standard defines 12 orthogonal channels in the 5.2GHz spectrum and the IEEE 802.11g standard defines 3 orthogonal channels in the 2.4GHz spectrum. Traditionally, these channels are used by different networks operating in the same vicinity. However, it is possible to use these channels by different devices of the same network or by the same devices of the same network by the use of multiple transceivers or radios per device. Multiple channels can be used simultaneously in this fashion under several scenarios: (a) isolated environments where there are no other legacy devices using the channels and (b) opportunistic shared environments where the channels are not always used by other legacy devices. The use of multiple channels in multi-hop networks is non-trivial, because of the complexity involved in coordinating a distributed set of devices, to efficiently use the available channels. Thus my research answers the following question *"If multiple channel/radios are used for multi-hop wireless networks, how do we use them efficiently?"*

We divide the challenges involved in answering the above question in to network wide challenges and link level challenges. Network wide challenges are those affecting the entire set of wireless devices in the multi-channel wireless network. Link level challenges deal with problems with a single link across two wireless devices. In this thesis, we consider two important network wide problems in multi-channel multi-hop networks involving channel assignment and routing. At the link level, we identify the practical challenges associated with using multiple orthogonal channels for providing a high data-rate wireless link across two devices.

Multiple channels can be exploited by using a single radio (or interface) per device or by having multiple radios per device. In the former scenario, two devices, wishing to communicate tune their radios to the same channel and exchange information while other devices, in the vicinity, would be tuned to other channels. In the latter scenario, two devices can potentially tune to multiple channels at the same time, using the multiple radios, and communicate on multiple channels simultaneously. The decision to use a single radio or multiple radios per device depends on the implementation requirements, dictated by various factors including ease of deployment, compatibility of devices, and cost among others. We consider both scenarios in our work:

- First, we consider single-radio multi-channel multi-hop networks. In this work, we explore the *granularity of channel assignment decisions* that gives the best trade-off in terms of performance and complexity. By granularity, we refer to the scope of a channel assignment decision, in terms of the number of different entities the decision impacts and applies to. We explore the trade-offs of three existing granularities of channel assignment in such networks, and in the process we arrive at a new granularity for channel assignment that we refer to as *component based channel assignment (CBCA)*, which is the least complex of the ones identified above and hence is characterized by several impressive practical advantages. We analyze the theoretical gains of the proposed granularity. We propose centralized and distributed algorithms for realizing CBCA. We evaluate CBCA and compare it with other related work using simulations with the open source ns2 network simulator and a strawman testbed implementation.

- We then consider multi-radio multi-channel adhoc networks. In this work, we identify a special property of such networks known as the *interface insufficiency bottleneck*. This bottleneck results in a poor utilization of the available multiple channels and multiple radios, when traditional routing protocols are used. We propose a novel routing algorithm known as *Lattice routing* that uses multiple paths for every flow, to combat the bottleneck. The protocol is also dynamic in adjusting routes based on changing traffic conditions. We evaluate Lattice routing, with simulations using the open source ns2 network simulator and compare the proposed work with related works.

- Finally, we study practical link level problems in actually achieving a multi-channel wireless link between two devices. People in the research community working on related problems have traditionally assumed the feasibility of multi-radio usage as a given. However, we observe that while a single radio, multi-channel link is trivial to realize (since existing off-the-shelf wireless implementations allow a radio to switch across different available channels), realizing a multi-radio, multi-channel link is not straight-forward. The 802.11 a/g standards define a total of 15 orthogonal channels in the 2.4GHz and 5.2GHz bands. The theoretical maximum data-rate possible when using all the 15 channels using multiple-radios, each on a different channel is *600Mbps*. We find through experimental evaluation that the actual observed performance when using all fifteen orthogonal channels between two devices is a mere *91Mbps*! We identify the reasons behind the low performance and present Glia, a software only solution that effectively exercises all available radios. We prototype Glia and show using experimental evaluations that Glia helps achieve close to 600Mbps data-rate when using all possible wi-fi channels.

The rest of the thesis is organized as follows. In Chapter 2, we provide the origin and history of the thesis, where we discuss the work in related academic literature and commercial products. In Chapter 3, we present CBCA, the channel assignment and routing solution for single-radio, multi-channel adhoc networks. We provide both theoretical and practical evidence (in the form of simulation analysis and a strawman prototype implementation) for why CBCA outperforms existing work on single-radio multi-channel adhoc networks. In Chapter 4, we present Lattice routing, the routing solution for multi-radio, multi-channel adhoc networks. We provide simulation results to show the efficacy of the solution. In Chapter 5, we identify the practical challenges in using multiple Wi-fi channels and present a fully functional prototype implementation of Glia, our solution to overcome the challenges. The prototype is used to showcase the first high data-rate wireless link using only off-the-shelf components. In Chapter 6, we present the conclusion and discuss ideas for future work that can potentially spawn from this thesis.

CHAPTER II

ORIGIN AND HISTORY OF THE PROBLEM

There has been significant work in the context of multi-channel wireless networks. The works most relevant to the present work are [13, 5, 30, 24, 3, 31, 6, 10, 11, 12]. The related work can be divided into the following categories, depending the network layer at which channel assignment strategies have to be implemented and theoretical work on capacity of wireless networks:

2.1 Capacity

There have been several approaches to determine the capacity of wireless networks [6, 10, 11, 12]. In [6], the authors derive the transport capacity of wireless networks under the arbitrary and random network model. The results are applicable to single channel wireless networks, or multi-channel wireless networks where every channel has a dedicated interface. [12] extends the results of [6], for multi-channel wireless networks with varying number of interfaces. The assumptions in this work are similar to those in [6]. While [2, 10] consider the problem of optimal channel assignment, scheduling and routing using a linear programming technique, their analysis is for a link-based channel assignment. [11] extends the analysis of [10] for multiple interfaces.

2.2 Multi-Channel Routing and Channel Assignment Approaches

In [13], a flow-based routing and channel assignment approach has been proposed for a single interface. The authors identify flow-based and node-based assignment as two possible approaches to channel assignment. These approaches are based on simple heuristics and the authors present a simulation analysis in restrictive environments.

In [30], a routing architecture for multi-channel packet-radio networks is proposed, for both single radio and multi-radio networks. As in the previous work, this work provides heuristics to perform routing. The authors identify the broadcast storm problem, but provide no solution for the problem. In [5], the authors propose a multi-radio routing protocol that routes traffic based on existing traffic conditions. However, this scheme requires as many radios as there are channels. Raniwala et al. [27] propose a load aware algorithm to dynamically assign channels in a multi radio mesh network. This scheme requires that anticipated traffic loads and paths traversed by flows are know before channel assignment takes place.

A number of multi-path routing protocols for adhoc networks have been proposed ([15, 22, 37, 20, 33]) in literature. AOMDV ([20]) is an extension of AODV that computes multiple loop free and link disjoint paths for every source destination pair. In [33], the authors propose a multi-path scheme for multi-channel single radio Mesh networks. This work uses two node-disjoint paths for every source destination pair and does not consider traffic conditions at a node location. Further, since the protocol is only for single radio networks, maximum improvement was shown to be be only about 2 times single channel performance. Split Multi-path Routing [15] finds multiple routes using the flooding behavior of RREQ packets.

2.3 Multi-Channel Link and MAC Approaches

[3, 31] are medium access control solutions for a multi-channel, single interface network. SSCH [3] is a link layer protocol for frequency hopping systems, where every node switches channels periodically following a pre-determined pattern. MMAC [31] uses a contention window based approach for channel agreement, and the data transmissions are scheduled in a periodic time-slotted manner. The above approaches are flow-unaware and cannot perform channel assignment at a granularity greater than a link. In [24], the authors describe a joint channel allocation, interface assignment

and MAC design for multi-radio mesh networks, where they formulate a non-linear mixed integer program. Realizing this scheme requires a complex central processing unit, and hence it is not practical for a realistic network. Similarly works such as [38], [9] and [18] also involve complex scheduling strategies that are difficult to realize.

2.4 Practical Multi-Radio Multi-Channel Usage

There have been some works that identify practical issues with using multiple radios on a single node. In [28], the authors study a three node, two-hop testbed, with the common node having two 802.11 radios. They study only the two-hop behavior of the network and conclude that if a single node contains 2 wireless cards alone, these cards will not be able to receive or transmit traffic at the same time, unless their antennas are separated by more than 35db. In [1], the authors identify the interference across two wireless interfaces on the same node, each using a different channel. Similarly, in [16, 19, 4],the authors argue that it is not possible to simultaneously use two radios on the same node. In [39], the authors study the challenges and opportunities for multi-radio coexistence on a single node. Unlike in other works, the authors study coexistence of radios belonging to heterogeneous technologies like 802.11, WiMAX, and Zigbee. In a presentation at the Spring Intel Developer forum '08 [42], the authors provide a direction towards hardware multi-radio integration. The goal is to design a single chip solution for wifi and WiMAX to coexist in the same band. The idea is to to use a high performance ADC (analog to digital converter), that reduces the impact of the leakage power.

Channel bonding techniques have been known for some time and have been proposed for the new 802.11n standard [45]. However, the standardized Channel bonding in 802.11n is only for 2 adjacent channels. Further, new physical hardware conforming to the 802.11n standard is necessary for getting the benefits of such channel bonding. The 802.11n hardware is, however, compatible with existing 802.11 a/b/g devices.

The maximum application bandwidths of commercial 802.11n equipment is in the order of 180Mbps [46]. Efforts are on for ratifying a new wifi standard known IEEE 802.11 Very High Throughput (VHT) [44]. Throughput in excess of one gigabit per second, using 100MHz of bandwidth in either the 5.2GHz or 60GHz spectrum, is the goal of this initiative. The new standard would likewise need new hardware. It is not yet clear if the new standard would be backward compatible with existing 802.11 a/b/g devices. Other wideband solutions have been shown to work in principle by works such as [25, 35, 50]. In [25], the authors present a wideband solution in the 5.2GHz spectrum, known as SWIFT, that can coexist with other narrow band devices in the same frequency by weaving together non-contiguous unused frequency bands. The maximum bandwidth shown by SWIFT is close to 500Mbps. All these wideband solutions need new physical hardware and are not compatible with other wifi devices [51, 49]. Advanced antenna technologies, like directional antennas, MIMO, and adaptive antenna arrays have been developed for existing standards. However, these technologies require additional hardware level modifications. While these products are backward compatible with other wifi devices, and conform to existing 802.11 standards, they require new physical hardware to provide higher bandwidths. The maximum per-client bandwidth advertised by such products is 300Mbps. Several wireless networking companies offer multi-radio wifi APs [46, 40]. However, these products bind the radios on different bands (2.4GHz and 5.2GHz). The multiple radios cannot be used to operate in the same frequency band. The maximum advertised throughput using such products is around 300Mbps. Advanced physical layer techniques like [7, 17], can also be used to provide a high bandwidth wireless link. However, these techniques require major changes to existing standards and also need new physical hardware. While these advanced physical layer techniques could be made to be standards compliant, they require new physical hardware to obtain the benefits. Such advanced techniques have only been demonstrated at bandwidths of around 11Mbps (802.11b).

In [26], the authors present 2P, a MAC protocol for long-distance 802.11 mesh networks. The proposed work uses two radios, with directional antennas operating on the same channel, at every node. Although the directional antennas face different directions, it is found that some amount of leakage power from one antenna, causes problems at the other antenna because of side-lobes. 2P works only on one channel and is is not backwards compatible with other legitimate 802.11 traffic. WildNet [23] builds upon 2P to improve the loss resiliency of long distance mesh networks. Both these works use only 2 radios at a single node.

A commercial product called 802.11abg+n is manufactured by Xirrus, Inc [51]. The product is a 16 radio wifi AP with directional antennas. The AP uses 16 radios to divide 360 degrees into 16 sectors, each of which is served by a separate radio. However, the AP cannot use 16 different omni-directional radios. More importantly, the notion of providing bandwidth aggregation is not supported on a single link to a single client. Hence, the throughput deliverable to a single client is restricted to that of a single radio.

CHAPTER III

SINGLE RADIO MULTI-CHANNEL AD-HOC NETWORKS: CHANNEL ASSIGNMENT AND ROUTING

3.1 Introduction

Multi-channel wireless data networks have garnered increasing attention over the last few years because of the great promise they hold in terms of achievable spectral efficiencies. In this work, we consider a specific sub-topic of the above general area: *adhoc networks* with nodes equipped with a *single radio or interface that can operate on multiple channels.* Within this context, an important problem to solve for attaining any of the perceived benefits of a multi-channel environment is one of *channel assignment*. Simply put, the channel assignment problem asks: *On which of the available channels should a node transmit at any given point in time?* The problem is not a new one and has been answered with a different extent of efficacy by several related works, with solutions such as SSCH [3], MMAC [31], MCP [13], DCA [36] etc.

In this work, we explore the *granularity of channel assignment decisions* that gives the best trade-off in terms of performance and complexity. By granularity, we refer to the scope of a channel assignment decision in terms of the number of different entities the decision impacts and applies to. Briefly, examples of different granularities include (i) *packet* - channel assignment is performed on a per-packet basis at a given node and the decision does not apply to subsequent packets or other entities; (ii) *link* - channel assignment is performed for a link between two given nodes, and all packets between the two nodes will be transmitted on the same channel for the duration the decision is valid; and (iii) *flow* - all packets belonging to a flow are sent on the same channel. Approaches such as DCA fall under the category of packet-level channel assignment, approaches such as MMAC and SSCH fall under the category of link-level channel assignment, and approaches such as MCP fall under flow-level channel assignment.

The different channel assignment strategies have different trade-offs in terms of the overall performance they can achieve, and the complexity and hence the practical overhead incurred in realizing them. We explore these trade-offs and in the process arrive at a new granularity for channel assignment that we refer to as *component-based channel assignment*, which assignment is the least complex of the ones identified above. Hence it is characterized by several impressive practical advantages, including (i) no changes to the off-the-shelf radio hardware or MAC algorithms, (ii) no synchronization requirements, (iii) no channel scheduling overheads, and (iv) no switching between channels to serve data flows. Surprisingly, we also show that the theoretical performance of the component-based channel assignment strategy does not lag significantly behind the optimal possible performance even under worst case conditions, and for most practical scenarios does the same as the optimal. Further, we show that when coupled with its several practical advantages, it significantly outperforms other strategies under most network conditions.

Briefly, the component-based channel assignment strategy involves assigning a single channel to all nodes belonging to a component formed by nodes belonging to mutually intersecting flows. For example, if flow $f1$ intersects with flow $f2$ and flow $f2$ intersects with flow $f3$, then all nodes on the paths traversed by the three flows are assigned to operate on the same channel. We show that such a simple strategy can result in considerable performance gains through both theoretical and quantitative analysis. We also propose centralized and distributed routing layer algorithms that effectively realize the strategy. Thus, the contributions of this work are threefold:

- We identify a new granularity for channel assignment that is component-based and show that the strategy has several theoretical and practical benefits.

- We present centralized and distributed routing algorithms that realize the component-based channel assignment strategy effectively.
- We show through a testbed implementation using off-the-shelf hardware the ease of deployment of the component-based strategy.

3.2 Background

In this work, we consider the problem of channel assignment for different flows in the following context:

- *Network Model:* We consider a multi-hop, adhoc network where there are multiple channels available in the network.
- *Transceiver Model:* We assume that all nodes in the network are equipped with a *single* half-duplex transceiver.
- *Flow Model:* We consider the case where flows can either be single hop or multi-hop. Also, a node can potentially serve one or more flows.

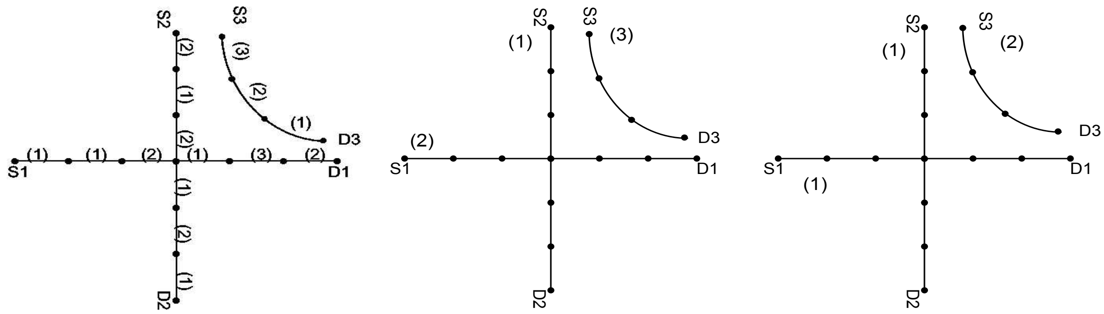
Given the context, channel assignment in a multi-channel adhoc network can be done in one of the following three ways[1]:

3.2.1 Link-Based Channel Assignment

We refer to a multi-channel assignment as *link-based assignment when different links in the flow graph, induced by the different flows in the network, have the capability to choose any of the channels.* In this type of assignment, each link in a flow can potentially operate on a different channel. Figure 1 (i) illustrates the link-based channel assignment for a topology with three flows and three channels. In a link-based assignment, we observe that different links in the flow can potentially be assigned to

[1] We have identified packet based channel assignment as another type of channel assignment. However, it has been shown in [3, 31], that channel assignment at such a fine granularity may not be feasible in a practical setting because of the various overheads involved.

different channels. Thus, the link-based channel assignment leverages the presence of multiple channels to increase the spatial reuse at the granularity of a link.



(i) (ii) (iii)

Figure 1: Topologies to illustrate (i) link, (ii) flow and (iii) component based channel assignment

3.2.2 Flow-Based Channel Assignment

We refer to the channel assignment as flow-based assignment *when all links in a flow are assigned to a single channel, but different flows have the capability to operate on different channels.* Thus, the channel assignment is performed at the granularity of a *flow*. Figure 1 (ii) illustrates the flow-based channel assignment for the the same topology. The two intersecting flows and the third flow can potentially operate on different channels. However, all the links in a particular flow operate on the same channel.

3.3 Component-Based Channel Assignment

We refer to the channel assignment as component-based *when all links in a connected component induced by the underlying flow graph operate*[2] *in a single channel. However, different connected components can potentially operate on different channels.* A connected component in a flow graph is defined as the largest subgraph, such that there exists a path between any node in the subgraph to all other nodes in the subgraph. Figure 1 (iii) illustrates the component-based channel assignment for the same

[2] *The set of active edges carrying flow traffic in the network.*

topology. The two intersecting flows[3] form a connected component and operate on a single channel, while the third flow is an independent component and can potentially operate on a different channel. All the links in a particular component operate on the particular channel assigned for the flow. Thus, we leverage the presence of multiple channels at the granularity of a component.

Although the component-based model is simple, one of the contributions of this work is to show that this model has equal if not better performance over the more complex link and flow-based approaches.

3.4 Motivation

In this chapter, we compare component-based with link-based and flow-based channel assignment using intuitive, quantitative, and practical analysis. For the intuitive analysis, we compare component-based with only link-based, as it has been established that for a given flow graph, the link-based approach provides the optimal performance [2, 11]. However, for quantitative results and practical reasoning, we compare all three approaches.

3.4.1 Simple Topologies

In this section, we provide intuitive evidence for why a component-based channel assignment is efficient. We consider a few practical topologies and perform the slot and channel assignment for component-based and link-based channel assignment.

Topology 1:

Figure 2(i)(a) shows the slot and channel assignment for a single flow using a single channel[4]. We observe that it is possible to come up with a schedule where links within the same contention region are assigned to different slots. This sequence is repeated

[3] Two flows are said to be intersecting, if there is a common node in the set of active nodes for each flow, which serves both flows.

[4] For topologies 1-3, component-based assignment reduces to that of a single channel where only one channel is utilized.

across different contention regions. If W is the link capacity, we observe that this slot allocation scheme yields a flow capacity of $\frac{W}{3}$, assuming a two-hop interference region.

Figure 2(i)(b) shows the link-based slot and channel assignment where the per-flow capacity is $\frac{W}{2}$. We observe that, irrespective of the number of channels and the slot schedule, the flow capacity is always limited to $\frac{W}{2}$, as each node is equipped with a single half-duplex radio. Thus, the flow capacity of single and multi-channel assignment for a single flow is of the same order. Note that this is valid irrespective of the number of hops in the flow.

Topology 2:

Figure 2(ii) shows the single-channel and link-based multi-channel slot and channel assignment for two intersecting flows. Figure 2(ii)(a) shows a single-channel slot assignment that will guarantee an aggregate flow capacity of at least $\frac{W}{3}$.

Figure 2(ii)(b) shows a link-based slot assignment that yields an aggregate flow capacity of $\frac{W}{2}$. Note that, irrespective of the number of channels, the capacity around the bottleneck (intersection) node can at most be $O(W)$. Thus, for intersecting flows, there is no benefit in using multiple channels.

Topology 3:

Figure 3(iii) shows the single-channel and multi-channel assignment for multiple, non-contending bisecting flows. We observe that the aggregate flow capacity scales with the number of flows as each flow achieves a per-flow capacity of at least $\frac{W}{6}$. In fact, for some flows, the flow capacity is $\frac{W}{4}$. Thus, for the given topology, the aggregate flow capacity for a single channel is $O(F * W)$, where $F = 6$ is the total number of flows in this example.

For a multi-channel scenario with a single radio, the maximum achievable aggregate flow capacity for F flows is $O(F * W)$. Figure 3(iii)(b) confirms this observation, where the per flow capacity of each flow never exceeds $\frac{W}{3}$.

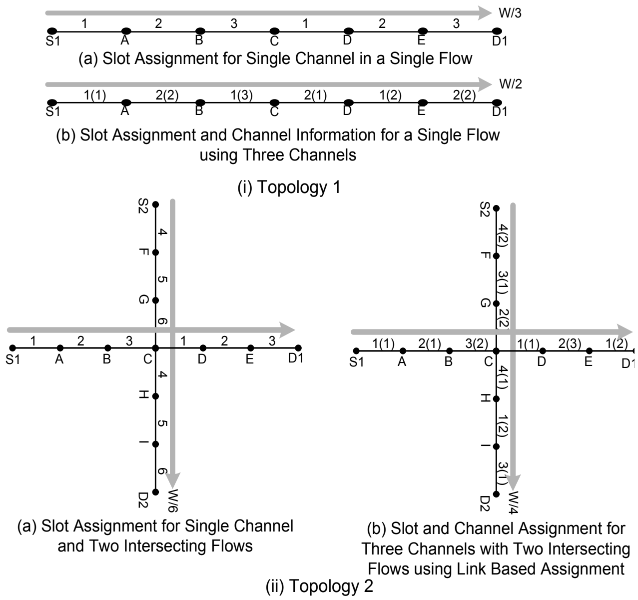


Figure 2: Slot assignment for simple topologies 1 and 2

Topology 4:

Finally, when F flows contend in a region as shown in Figure 3(iv), the component-based channel assignment reduces to a flow-based channel assignment. Figure 3(iv)(a) shows the slot and channel assignment for three contending[5] but non-intersecting flows. If each component operates on a separate channel as shown in the figure, the per-flow capacity is still $O(W)$. So, for the F flows, where $F = 3$ in Figure 2 (iv)(a), the aggregate flow capacity is $O(F * W)$. This is also the maximum achievable flow capacity for a link-based channel assignment as shown in Figure 2(iv)(b).

3.4.2 Quantitative Results

In the previous section, we observed that component-based and link-based channel assignment provide similar aggregate capacity for the topologies considered. In this section, we observe the performance of link-based, flow-based and component-based channel assignment for a random network through simulation results.

[5]Two flows are said to be contending, if there is at least one node in the set of active nodes for one flow that is within the interference region of the set of active nodes for the second flow.

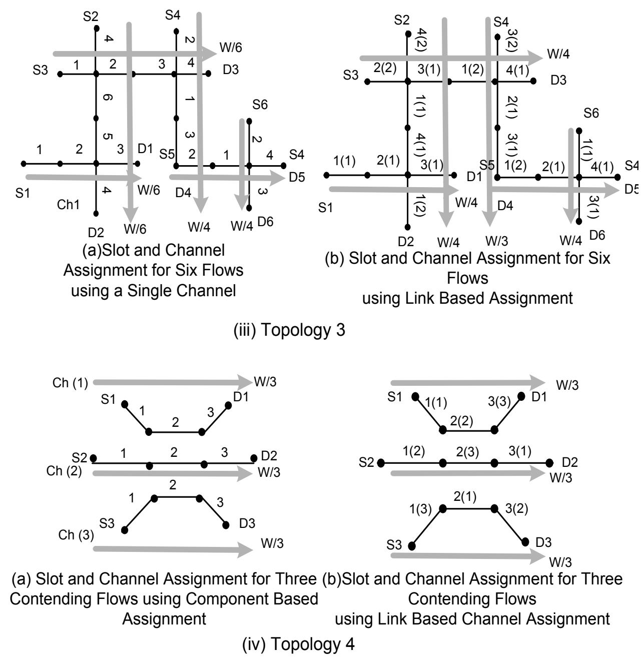
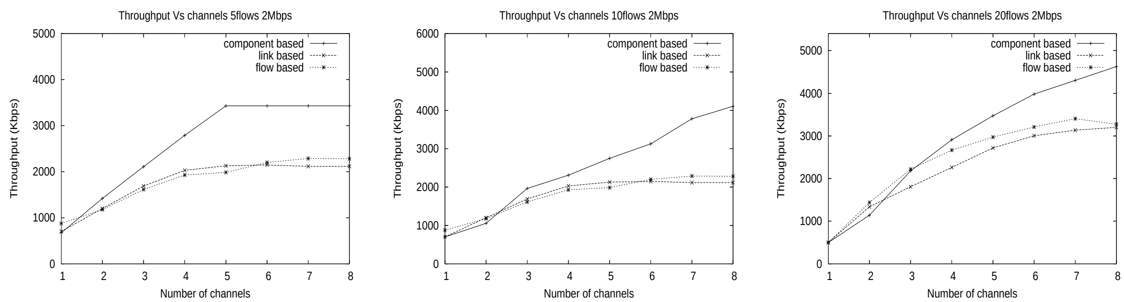


Figure 3: Slot assignment for simple topologies 3 and 4

Figure 4 compares the average throughput for component-based with flow and link-based channel assignment using $ns2$ simulations. We consider a network of size $750m \times 750m$ with 100 nodes randomly deployed with a transmission range of 250 m, channel data rate of 2 Mbps, and varying number of flows. The other details of the simulation setup and the competing approaches are described in Section 3.7.

Figures 4 (a)-(c) compare the average throughput for all three types of channel assignment for five, ten and twenty flows. The total number of channels is varied from one to eight. Figure 4 (a) shows the average throughput for all three approaches for five flows. For the component-based approach, we observe that there is a linear increase in the average throughput from about 700 Kbps for one channel to about 3500 Kbps for five channels. Note that there cannot be any further increase beyond five channels as there are only five flows. The linear increase in throughput is due to the different components or flows being assigned to different channels when the number of channels is increased. For the flow-based and link-based, the average throughput saturates at about 1800 Kbps and 1500 Kbps respectively. This is due to several practical constraints, such as lack of synchronization, inefficient scheduling, and the penalty incurred in switching between channels (switching delay). Figures 4 (b), (c) show the throughput variation with increasing number of channels for 10 and 20 flows. We observe that the difference between component-based and link-based and flow-based decreases with an increasing number of flows. This is due to the increase in the number of intersecting flows.



(a) Throughput for 5 flows (b) Throughput for 10 flows (c) Throughput for 20 flows

Figure 4: Average throughput (Kbps) vs. no. of channels for varying number of flows for link, flow and component based channel allocation

3.4.3 Practical Considerations

Thus far, we have compared the performance of component-based with link-based and flow-based assignment through simulation results and for simple topologies. Here, we describe some of important practical limitations of link-based and flow-based assignment that are not present in component-based channel assignment.

- *Hardware/MAC Changes:* Most of the current realizations of the link-based approach are performed at the MAC layer [3, 31]. Even for a flow-based approach, modification is required at the MAC layer to accommodate fine-grained *switching* at the intersection points [13]. This imposes the need to build customized wireless cards to support customized MAC-layer functionality. For this reason, standard off-the-shelf wireless cards cannot be used. However, a component-based approach is able to achieve almost identical benefits without imposing any requirements for changes in MAC hardware or software.

- *Switching Delay:* Link-based and flow-based approaches require switching when an intersection node serves two links or flows in different channels[6]. For a typical 802.11a card, the switching delay is on the order of 80-100 μs [3]. Consider the example where the data packet size is 1 KB. The packet transmission time is given by $8000/(54 \times 10^6) = 160$ μs. Thus, the switching delay in this example is of the same order as the packet transmission time. Further, the end-to-end delay for each packet transmission in a flow will increase as the switching delay is additive across all nodes that perform switching. It has also been observed that the network capacity degrades as a function of $\frac{S}{S+T}$, where S is the switching delay and T is the transmission time [12].

- *Synchronization Requirements:* Another important consideration in link-based and flow-based approaches is the need to perform synchronization at the intersection nodes [3, 31]. When a common node serving two links (or flows), A and B, performs switching from A to B, it requires that (i) the receiver for that particular link (or flow), B, also be on the same channel, and (ii) the sender of the previously served link (or flow), A, does not transmit packets for the duration of time when the common node is serving B. Constraint (i) is required for efficient operation, while constraint (ii) is required to prevent the previous from triggering unnecessary route failures (stable operation). In link-based and flow-based approaches, both constraints need to be addressed. However, in a component-based approach, a connected component is on a single channel and does not suffer from these issues.

- *Scheduling Overhead:* A problem associated with synchronization is the need

[6]The frequency of switching is dependent on the specific protocol and could potentially be at the granularity of a packet [36].

to perform efficient scheduling for all the links or flows that operate on different channels and pass through a common node. The common node needs to inform the schedule for neighboring nodes that operate on different channels. The overhead involved in this process makes the link-based and flow-based approaches less desirable. An alternative to avoid synchronization and scheduling in the link-based and flow-based approaches is to use a control channel for control packet transmissions and perform data transmissions on the remaining channels [36]. However, this is not desirable in a single-radio scenario, as it requires frequent switching between data channels and a control channel.

3.5 Theoretical Analysis

In this section, we derive analytical results for the following problem: Given a *flow-graph* in a *random* network, determine *aggregate flow capacity* bounds for link-based and component-based channel assignment. For the derivation of these bounds, we assume the underlying network graph is planar [6], ensuring that the flow graph is also planar. The notations used in the derivation of these results are shown in Table 1.

Based on the insights gained in the slot and channel assignment for simple topologies in Figure 2, we make the following observations:

- *Observation 1:* For a single flow in the network, the capacity of a single-channel assignment and multi-channel assignment is of the same order.

- *Observation 2:* When there are F non-contending and non-intersecting flows in the network, the aggregate flow capacity of a single-channel assignment and multi-channel assignment is of the same order.

- *Observation 3:* When F non-contending flows in the flow graph intersect at a single point, the aggregate flow capacity of a single-channel assignment and a

Table 1: Notations for capacity analysis.

Variable	Description
W	Capacity of a single channel
$G(V,E)$	The underlying network graph
V	Set of vertices in the network graph
E	Set of edges in the network graph
F	Total number of flows in the network
$\Lambda(i)$	Aggregate flow capacity of i flows
$G'(P,L)$	The flow graph for the underlying network
P	Set of vertices in the flow graph
L	Set of links in the flow graph
c	Total number of channels
Δ	Maximum number of contending flows in the flow graph
Γ	Maximum number of intersecting flows in the flow graph

multi-channel assignment is of the same order.

- *Observation 4:* When F non-intersecting flows in the flow graph contend in a single contention region, the aggregate flow capacity for component-based and link-based assignment is of the same order.

We now present the upper and lower bounds of capacity for link-based and component-based channel assignment. Any given flow graph, $G'(P,L)$, can be classified into the following categories:

Case (i): Non-intersecting and non-contending flows.

Case (ii): Non-intersecting but contending flows.

Case (iii): Intersecting but non-contending flows.

Case (iv): Contending and intersecting flows.

For ease of analysis, we treat these cases in isolation and consider the flow graph to exclusively belong to one of the classifications. The bounds for a generic case where a flow graph is composed of a few of these classifications can be derived by aggregating the bounds derived for each subgraph.

Table 2: Bounds for link-based and component-based channel assignment.

Type	Condition	Link LB	Link UB	Comp LB	Comp UB
NC	N/A	$O(WF)$	$O(WF)$	$O(WF)$	$O(WF)$
C	$\Delta \leq c$	$O(WF)$	$O(WF)$	$O(WF)$	$O(WF)$
C	$\Delta > c$	$O(\frac{WFc}{\Delta})$	$O(W(c+F-\Delta))$	$O(\frac{WFc}{\Delta})$	$O(W(c+F-\Delta))$
I (NC)	N/A	$O(\frac{WF}{\Gamma})$	$O(W(1+F-\Gamma))$	$O(\frac{WF}{\Gamma})$	$O(W(1+F-\Gamma))$
I and C	$\Delta \leq c+\Gamma-1$	$O(\frac{WF}{\Gamma})$	$O(W(1+F-\Gamma))$	$O(\frac{WF}{\Delta+\Gamma})$	$O(W(1+F-\Gamma))$
I and C	$\Delta > c+\Gamma-1$	$O(\frac{WF}{\Gamma})$	$O(W(c+F-\Delta))$	$O(\frac{WF}{\Delta+\Gamma})$	$O(W(c+F-\Delta))$

From observations (i)-(iv), the capacity bounds for link and component-based are of the same order for cases (i)-(iii). The proofs for the first three cases follow from observations (i)-(iv) and are not presented due to lack of space[7]. We present the bounds for link and component-based in Table 2. We now derive the bounds for case (iv).

Case (iv): Contending and Intersecting Flows:

Lower Bound:

The worst case is when all Δ and Γ flows contend and intersect at a single point, and there are several such points in the network. For link-based, consider the case where these Δ flows intersect at some other region in groups of Γ flows. For link-based, the aggregate flow capacity of Δ flows is given by

$$\begin{aligned} \Lambda(\Delta) &= \frac{\Delta}{\Gamma} * \Gamma * O(\frac{W}{\Gamma}) \\ &= \Delta * O(\frac{W}{\Gamma}). \end{aligned} \quad (1)$$

For the Γ intersecting flows, the aggregate flow-capacity is given by

$$\Lambda(\Gamma) = \Gamma * O(\frac{W}{\Gamma}). \quad (2)$$

From equations 1 and 2, the aggregate flow capacity of F flows for link-based is given

[7]In the derivation of bounds for pure contending flows, we have leveraged the property that the flow graph is planar.

by

$$
\begin{aligned}
\Lambda(F) &= (\Delta + \Gamma) * O(\frac{W}{\Gamma}) * \frac{F}{\Delta + \Gamma} \\
&= O(\frac{W * F}{\Gamma}).
\end{aligned}
$$

For component-based, consider the case where the Δ contending flows in each region intersect with one of the existing Γ intersecting flows. Since all these flows operate on a single connected component, by definition of component-based, all flows will operate on the same channel. Thus, the aggregate flow capacity for component-based is given by

$$
\begin{aligned}
\Lambda(F) &= (\Delta + \Gamma) * O(\frac{W}{\Delta + \Gamma}) * \frac{F}{\Delta + \Gamma} \\
&= O(\frac{W * F}{\Delta + \Gamma}).
\end{aligned}
$$

Upper Bound:

The best case occurs when Γ flows intersect in a point, and these Γ flows also contend with each other at some other region. For the Γ intersecting flows, the aggregate flow capacity is $O(W)$ for link and component assignment. For the remaining $\Delta - \Gamma$ contending flows, the maximum achievable aggregate capacity is given by:

$$
\begin{aligned}
\Lambda(\Delta - \Gamma) &= min[(\Delta - \Gamma)O(W), O(cW - W)] \\
&= (\Delta - \Gamma) * O(W) : \Delta \leq c + \Gamma - 1 \quad (3) \\
&= O(cW - W) : \Delta > c + \Gamma - 1 \quad (4)
\end{aligned}
$$

For the remaining $F - \Delta$ flows, the maximum achievable capacity per flow is $O(W)$ for *both* types of channel assignment as they do not intersect with any of these flows. Thus, the aggregate flow capacity for both link-based and component-based is given

Table 3: Worst case competitive ratio of component-based with respect to link-based assignment.

Type	Condition	Link/Component
NC	N/A	$O(1)$
C	N/A	$O(1)$[a]
I (NC)	N/A	$O(1)$
I and C	$\Delta \leq c$, $\Gamma = 2$	$O(\Delta)$
I and C	$\Delta > c$, $\Gamma = 2$	$O(c)$

[a]For planar flow graphs. For non-planar graphs, the competitive ratio is given by $O(\frac{F}{c})$.

by

$$\begin{aligned}
\Lambda(F) &= O(W) + \Lambda(\Delta - \Gamma) + O(W) \times (F - \Delta) \\
&= O(W(1 + F - \Gamma)) : \Delta \leq c + \Gamma - 1 \qquad (5) \\
&= O(W(c + F - \Delta)) : \Delta > c + \Gamma - 1. \qquad (6)
\end{aligned}$$

Competitive Ratio for Component-based to Link-Based:

Thus far, we have analyzed the upper and lower bounds for link-based and component-based. While these are important bounds to study the absolute performance of each of these channel assignment strategies, it is also equally important to identify the worst case competitive ratio with respect to optimal. In this section, we derive the ratio of link-based to component-based for different types. Figure 3 summarizes the competitive ratio of link-based to component-based for all scenarios. From observations (i)-(iv), we notice that the competitive ratio of link to component-based is $O(1)$[8].

For intersecting and contending flows, the worst case scenario for component-based to link-based is when $\Gamma = 2$ and $F - 1$ non-intersecting but contending flows intersect with a single flow. In this case, for component-based channel assignment, all the flows

[8]For planar flow graphs. For non-planar graphs with only contending flows, the competitive ratio is given by $O(\frac{F}{c})$.

will operate on a single channel. The aggregate flow capacity of component-based is given by

$$\begin{aligned}\Lambda(F) &= (F-1)*O(\frac{W}{\Delta}) + O(W - \frac{W}{\Delta}) : \Delta \leq c \\ &= (F-2)*O(\frac{W}{\Delta}) + O(W) \\ &= O(\frac{W*F}{\Delta}). \end{aligned} \tag{7}$$

For the link-based, the $F-1$ contending flows can operate on different channels, and so the aggregate flow capacity for the same scenario is given by

$$\begin{aligned}\Lambda(F) &= (F-1)*O(\frac{W*c}{\Delta}) + O(W) : \Delta \leq c \\ &= (F-1)*O(W) + O(W) \\ &= O(W*F). \end{aligned} \tag{8}$$

From equations 7 and 8, the competitive ratio for link-based to component-based is given by $O(\Delta)$.

When $\Delta > c$, the worst case scenario is the same and the aggregate flow capacity of component-based remains the same. However, the aggregate flow capacity of link-based reduces to

$$\begin{aligned}\Lambda(F) &= (F-1)*O(\frac{W*c}{\Delta}) + O(W) : \Delta \leq c \\ &= (F-1)*O(\frac{W*c}{\Delta}) \\ &= O(\frac{W*F*c}{\Delta}). \end{aligned}$$

Thus, the worst case competitive ratio in this case reduces to $O(c)$.

3.5.1 Insights

- For (i) purely non-contending flows and (ii) purely intersecting flows, flow-based, link-based and component-based all have the same aggregate flow capacity.

- For a combination of intersecting and contending flows, the flow capacity of flow-based and link-based is dictated by the number of intersecting flows and the fraction of contending flows with respect to the number of channels at each node within the flow. The performance of component-based degrades to that of single channel if the F flows form a single connected component. However, the competitive ratio of component-based to link-based is at most $O(min[\Delta, c])$.

- For the contention case, the aggregate flow capacity of flow-based and component-based channel assignment converges to the aggregate flow capacity of link-based channel assignment when each flow contends with $O(\Delta)$ other flows. This happens when:

  1. all flows contend at a single bottleneck region.

  2. if the underlying network graph is planar.

3.6 Realizing the Component-Based Channel Assignment Strategy

We have provided the motivation for a component-based channel assignment in Section 3.4. In this section, we present centralized and distributed approaches for realizing a component-based channel assignment strategy.

3.6.1 Centralized Approach

(i) Overview:

In the previous section, we analyzed that the worst scenario comparing link-based and flow-based approaches occurs when there are both intersecting and contending flows. The key objective of the centralized approach is to minimize the occurrence of this scenario. In this regard, we propose a greedy centralized approach for path selection and channel assignment for a component-based channel assignment strategy. The goal of the path selection phase is to select paths that have *minimal intersecting* paths, given source-destination pairs. From the analytical results in Section 3.5, we observe that channel assignment only addresses flow capacity degradation due to contention in the network and not the case when there are intersections. Once the component set has been determined, channel selection is performed for the different components. This procedure *minimizes the contention* between different components in the underlying flow graph (generated after the path selection phase). In this fashion, the centralized approach identifies the component set efficiently and performs efficient channel assignment on the component set. We now describe the details of the approach with the pseudo-code described in Figure 5.

```
Variables:
i: Node id, c:Number of channels,
f:Flow id, F_j: Flow set at j_th iteration,
cid: Channel id, NU:Number of unassigned flows,
C(cid): Channel Contention Cost in channel cid,
NS(f):Node set for flow f, w(i):Node Weight,
ch(l):Channel assigned to component l,
δ :Node weight increment,
NUC:Number of unassigned components,
UCS:Unassigned Comp. Set,
ACS:Assigned Comp. Set,
PC(l,m):Pairwise contention cost between
components l and m,
TC(l):Total contention cost for component l

Route(f)
INPUT: k pair shortest path tree for all unassigned
(S,D) pairs
OUTPUT: NS(f)
For f = 1 to NU
    For each one of the k shortest paths for flow
    Compute path cost in the path
    Return(path(minimum(k path costs)))
    For each node i ∉ F_j on flow f
      w(i) = w(i) + δ
Return(NS(f))

Assign Channel(f)
INPUT: UCS
OUTPUT: ch(1) ...ch(NUC)
    Do
        For each component m in ACS
        (with channel x)
            PC(l,m) = sum(CF_x(l),CF_x(m))
            TC(l) = TC(l) + PC(l,m)
        l=maximum(TC(l))
        ch(l)=minimum(C(cid))
        Update ACS, UCS; Update C(id)
    While UCS ≠ NULL

Execution Sequence
For each unassigned flow f
    Route (f)
For each unassigned component l
    Assign channel (l)
```

Figure 5: Centralized CBCA Approach

(ii) Path Selection:

The path selection approach is performed in a greedy fashion where, given source-destination pairs, the path with the minimum number of intersections with already computed paths is determined. This is accomplished by the following procedure. For each source destination pair, k shortest paths are determined using a shortest path algorithm. The cost of a path is determined as the sum of the node weights w(i) for all nodes i in the path. *The path with the minimum aggregate weight is chosen as the path for this flow.* Once the path has been established, the weights of all the nodes that constitute this path and do not belong to already formed paths is incremented by a value δ. This is performed to dissuade future flows from choosing nodes that constitute this flow. The overall goal is to minimize the number of intersection points (nodes), so the path where a single node that serves many flows would be preferred over several nodes that serve exactly two flows. For this reason, we only increment the weights of nodes that do not belong to an existing path by δ. A high value of δ causes longer paths to be preferred over intersecting paths[9]. This procedure is repeated for all source destination pairs. In Figure 5, lines 10-16 describe the path selection procedure.

(iii) Channel Assignment:

Once the path selection procedure has been accomplished, the component set for the underlying flow graph is known. The channel selection procedure is performed in a greedy fashion where a component minimizes the contention with previously formed components. Let Assigned Component Set (ACS) refer to the set of components that have already been assigned channels and Unassigned Component Set (UCS) refer to the set of unassigned components. The total contention for a component l is obtained as the sum of its pair-wise contention with all assigned components. We also define a channel contention metric to determine the contention level for each channel. Here, pair-wise contention between components can be defined as the sum of all contending nodes between two components. The channel contention metric for a channel, l, can be defined by the number of nodes already assigned to the channel with

[9] We determine this value of δ empirically to be 3.

which the intended component contends (if it were to operate on that channel). The greedy algorithm works by selecting the component in UCS with the maximum total component contention metric and assigning it to the channel with the least contention metric. This procedure chooses the component with maximum contention with other components in the assigned component set, and assigns it to the channel with the least contention. In Figure 5, lines 20-27 show the channel assignment procedure.

(iv) Component Set Update:

Once a channel has been assigned to a component, the channel contention metric corresponding to the newly assigned component is updated. Also, the assigned component set and the unassigned component set need to be updated. This procedure is repeated for all components in the unassigned component set, UCS. In Figure 5, line 26 shows the modification of channel costs, ACS and UCS.

3.6.2 Distributed Approach

In this section, we present a distributed realization of the greedy component-based centralized approach. In the centralized approach, we perform path selection and identify the different components in the flow graph before efficient channel assignment is performed for different components. In a distributed realization, it is not possible for a node to know the complete list of components before channel selection is performed. Hence, in our distributed approach, channel and route selection are performed in an integrated fashion.

The approach presented in this section enables route computation, maintenance, and termination in a reactive and distributed manner. The approach does not require synchronization between nodes once a route has been established. At a high level, the receiver performs channel selection in an informed fashion based on the contention and channel usage for the different paths between source and destination. We now describe the basic operations in the distributed approach.

During the flow initiation phase between a source and destination, the source transmits a RREQ message on all *active* channels. The list of active channels is determined by passive channel monitoring of the neighboring nodes using a particular channel. Each intermediate node determines whether it already belongs to a certain component and if so piggybacks the active channel, the number of nodes in the component, and the component contention level in each channel, along with the route request message. This information is propagated by each node during the propagation of the RREQ message. The destination node determines the best path and channel for a flow based on the contention level in the component for each intermediate node (if it already belongs to one) and the penalty incurred in switching components. Destination nodes inform the intermediate nodes in the selected path with the channel chosen for the component. Intermediate nodes that already belong to a previous component update their component information and perform a component-level update on all other nodes. The intermediate nodes in the component also perform passive monitoring to determine the contention level in each channel. There are eight phases in the distributed approach:

1. Pre-preparation Process
2. Route Request Broadcast
3. Route Request Update
4. Channel Selection
5. Route Reply Propagation
6. Component Update
7. Route Maintenance
8. Flow Termination

Variables:
i: Node id, s_i:Source id,
d_i:Destination id, c:Number of channels,
ch:Current channel of node i,
x :Commitment Indicator,
$cf(c)$: Number of contending flows on channel
k around node i,
nc:Number of nodes in the component
to which node i belongs,
$CF_1(i) \ldots CF_c(i)$:Number of contending flows in each channel for Component i
$PKT-TYPE$:Packet Type,
$RREQ$:Route Request Packet,
$RREP$:Route Reply Packet,
$RREQ(r)$:Route Request of path r
$UPDATE$:Update Packet,
cc:channel id in the $RREP$ packet,
$active(i)$:List of active channels on node i
cid: Channel id, $comid$:component id,
$PC(i,j)$:Cost between component i and component j,
$TC(i)$:Total cost for component i,
***Transmit** $RREQ(i)$*
Transmit on all active channels $RREQ$ packet with a 4 tuple,
(x,ch,nc,$cf(1)$...$cf(c)$) ,
***Receive** $RREP(i)$*
If(ch != cc) ,
 $ch = cc$,
 Transmit update(i),
If ($x == 0$) $x = 1$
Update with (cid,$comid$,nc)
Transmit update(i)
Transmit update packet with 3-tuple (cid,$comid$,nc)
Receive update(i)
$ch = cc$
Transmit update(i)
Decision process(i)
For each $RREQ(r)$
For each component i in the RREQ packet with channel id x
For each component j in the RREQ packet with channel id y
PC(i,j) = difference(CF_x(i),CF_y(j))
TC(i) = TC(i) + PC(i,j)
$cc(r)$= channel(maximum(TC(i))), $TC(r)$ = maximum(TC(i))
If $RREQ(r) = k$ or timer expired
 cc=channel(maximum($TC(r)$))
Transmit $RREP$(i)
Execution Sequence
If $PKT-TYPE == RREQ$
If $d_i == i$
 Decision process(i)
Else
 Transmit $RREQ$(i)
If $PKT-TYPE == RREP$
 If($s_i == i$)
  Transmit data
 Else
  Receive $RREP$(i)
If $PKT-TYPE == Update$
 Receive update(i)
If $PKT-TYPE == Data$
 Forward Data on channel ch
Do
 Monitor channels and save active channels
 Update component information
  If (flow inactive == True)
   Reset state
   Send update message
While (!(epoch end))

Figure 6: Distributed Component Based Channel Assignment Approach

We now elaborate on the different phases of the approach using the pseudo-code shown in Figure 6.

(i) Pre-preparation Process:

Each node performs a pre-preparation procedure in order to aid in the determination of the component based routing and channel assignment. As part of the process it keeps track of two pieces of information: (i) the number of active channels in the neighborhood, and (ii) the total number of other components on each channel that are in the vicinity of its component. While the number of components locally in the vicinity of the node can be monitored locally, the total number is accumulated through reports from all nodes in its component. Component IDs are used to prevent double-counting of the number of contending components[10].

(ii) Route Request Broadcast:

During the flow initiation procedure, a source node that has data to send, broadcasts route request packets (RREQ) on all *the active channels* in its neighborhood. This procedure prevents unnecessary transmission on all available channels if there are no active neighbors in a particular channel. When the route request is transmitted by the source, it piggybacks the source and destination node identifier with the packet. Apart from this information, the source also specifies the current operational channel (set to $default$ if the source does not belong to an existing component), and the number of components in each channel in its neighborhood.

(iii) Route Request Update:

When an intermediate node receives the route request, it piggybacks the following n-tuple $(x, ch, nc, (cf(1)...cf(k)))$. Here x is the commit flag, which is set to 1 when a node is committed to a channel and 0 otherwise. The current operating channel, ch,

[10]We use the destination ID of the oldest active flow in the component as the component ID.

of node, i, is the operating channel of the component if it already belong to a component. In this case, the number of nodes in the component, nc, is also piggybacked. Otherwise, it is set to the default operating channel. Also, the component contention level in each channel for that particular node, $cf(1) \ldots cf(k)$, is piggybacked by each node. If a node does not belong to any component, this reduces to the local contention level on each channel.

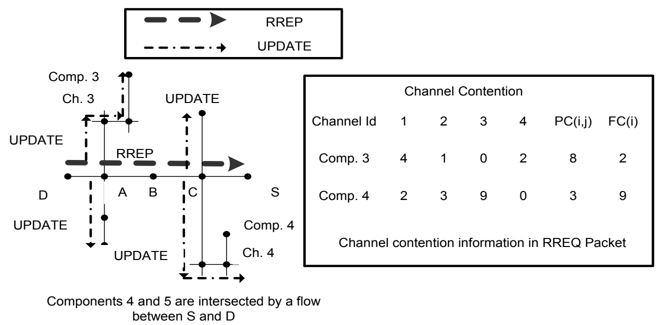


Figure 7: Component Channel Selection and Update Process

(iv) Channel Selection:

The destination waits for at a time corresponding to T_{RREQ} seconds or receipt of k $RREQ$, whichever occurs earlier, before selecting a path and channel for a particular route[11]. The destination chooses the path according to the following order of rules:

- If paths consisting entirely of uncommitted nodes are available, such a path with the minimum ambient congestion at any given channel is selected, and the path assigned to that channel.

- Otherwise, if paths consisting of some committed nodes, but with all on the same channel, are available, such a path with the minimum ambient congestion for the committed channel is selected, and the path assigned to that channel.

[11]In the simulation results, k is set to 3, and T_{RREQ} is set to 5 seconds.

- Otherwise, if only paths consisting of committed nodes, with nodes committed to different channels are available, the path with the minimum number of such channels is selected. Figure 7 illustrates this scenario, where there there is a path in which two nodes are already committed to different channels. Now, the destination needs to choose one of the channels and have all the other nodes in the other component switch their channels to that channel. The destination performs this operation by appropriately considering an overall penalty function associated with each of the components under consideration to switch. For instance, if the different components are say C1 and C2 operating on channels ch1 and ch2, the relative penalties based on the channel contention $C1(cf(ch1)-cf(ch2))$ and $C2(cf(ch2)-cf(ch1))$, referred to as FC_1 and FC_2, are considered. The total number of nodes in each of the components is also taken account as a cost function, PC_1 and PC_2. The overall penalty function for each component is computed as $FC_i + PC_j$, and the component with the smaller penalty function is made to switch. The same logic can be applied for more number of components as well, where the component with the maximum potential penalty is allowed to stay on its channel, and the other channels are made to switch. Figure 7 illustrates the component selection procedure for nodes belonging to two different components.

(v) Route Reply Propagation:

Once, the path and channel selection procedure has been performed by the destination, the route reply packet is transmitted on the chosen channel (see Figure 7). In addition, a unique component identifier is chosen for the new flow, and all pre-existing components as outlined earlier. The component identifier (with the maximum total penalty) corresponding to which the channel selection was performed, can be used as the new component identifier for all other components in the return path. In addition to that, the destination node also sends the total number of nodes in the newly formed component. This information can be computed from the original *RREQ* packet that was received. The destination node transmits the route reply on the channel information piggybacked on the original *RREQ*. Each intermediate node also performs the same operation.

(vi) Component Update:

As the route reply propagates, the intermediate nodes identify the chosen channel from the packet and updates this information for further transmissions. Further it also performs a component broadcast, where it informs all nodes in the component with the updated information. The component broadcast is a directed broadcast sent by nodes in a previously assigned component, where nodes receive a packet only if they belong to that component. Thus, the overhead of the broadcast mechanism is only limited to the number of nodes in the component. The route reply messages are sent upstream towards the source, and each intermediate node along the path performs a similar procedure. Note that nodes use the old (active) channel to propagate new component information so that nodes that still use the old channel, can update their information and also change channels if necessary.

(vii) Route Maintenance:

Whenever an intermediate node is unable to forward packets to a downstream node (towards the destination), it results in a route error. This triggers a route error message, which is propagated in to the source. The source initiates a new route discovery process as mentioned earlier in this section. Note that such a simple RERR scheme is possible only because of the fact that all nodes in the path are guaranteed to be on the same channel.

(viii) Flow Termination:

Flow termination is accomplished by the maintenance of *soft state*. When a node does not receive any packets from the upstream node in a flow for a threshold period of time T_{flow}, the flow is declared to be terminated. The nodes update their channel, commitment status and the contention values, and return to the default channel if they serve no other flows.

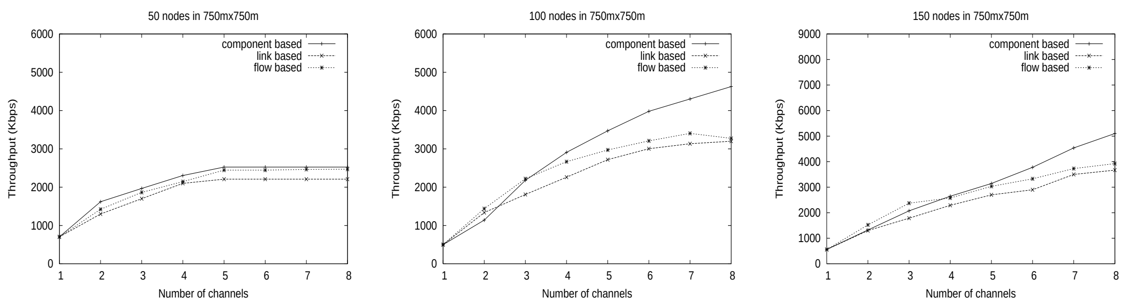
3.7 Performance Evaluation

3.7.1 Simulation Environment

We use $ns2$ for all our simulations. Unless otherwise specified, the simulations are carried out for a $750m \times 750m$ grid with 100 nodes placed randomly. We vary the number of orthogonal channels available from one to eight. We use three different transmission rates, namely, 2, 10, and 54 Mbps to reflect realistic 802.11 a/b/g datarates. By default, we use a 2 Mbps channel. We use constant bit rate traffic over UDP and try to maximize the utilization of the channels (i.e., we increase the traffic rate of each flow till we reach saturation in each scenario). All simulation results are shown over averaging 10 seeds of the topology generated using the random waypoint topology generator provided in $ns2$. We use a constant switching delay of 100 μs. Our focus is on multi-hop scenarios, rather than a single-hop network. We use DSR as the base routing protocol and modify it for certain cases. We simulate the distributed component-based approach described in Section 3.6.2 and approximations of the flow-based (MCRP [13]) and link-based (MMAC [31]) approaches. Since MMAC does not support broadcast inherently, we use pre-computed routes for simulating the link-based scheme. We use aggregate end-to-end throughput and average end-to-end delay to compare the three approaches.

3.7.2 Effect of Density of the Network

First, we study the effect of node density (Figure 8). We vary the number of nodes in a $750m \times 750m$ grid from 50 to 150. From the figure, it can be observed that the relative performance improvement of the component-based approach is significant for intermediate node densities. In a sparse network, there is not much improvement with increasing number of channels due to the presence of cut vertices at which many flows intersect. For sparse networks, the improvement in the component-based is comparable to the flow-based and link-based approaches. The link-based approach has slightly less throughput because of the 20 ms ATIM window overhead [31]. Also, for very dense network, there is a high probability that we have independent routes. Hence, for very high and very low node densities, all three approaches yield similar results.

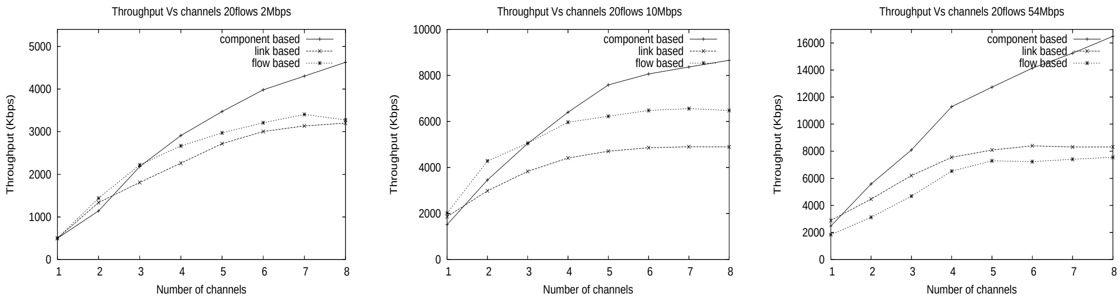


(a) 50 nodes (b) 100 nodes (c) 150 nodes

Figure 8: Effect of density of the network

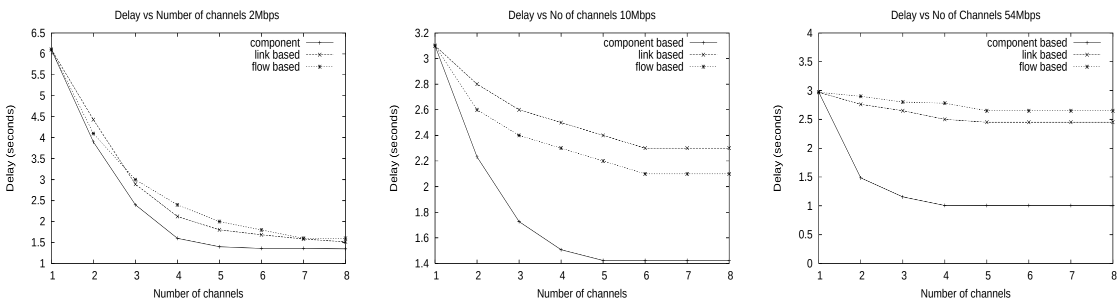
3.7.3 Effect of Channel Rates

Now, we look at the effect of the channel rate on the throughput (Figure 9). We simulate for 2 Mbps, 10 Mbps and 54 Mbps cases to reflect realistic 802.11 data rates. It can be observed that the relative performance improvement of the component-based approach increases with increasing channel data rates. Since the switching nodes (nodes that keep switching between flows) accumulate packets meant for the flow that is inactive, when they switch to a different channel for the new flow, they will not be able to transmit these packets, which leads to a significant number of packet drops for the flow on the new channel. This problem also leads to a large end-to-end delay (Figure 10). We find that, as the rate increases, the end-to-end delay for flow-based and link-based approaches is significantly higher than for the component-based approach due to switching delay and lack of synchronization at the intersection nodes.



(a) 2Mbps (b) 10Mbps (c) 54Mbps

Figure 9: Effect of channel rate

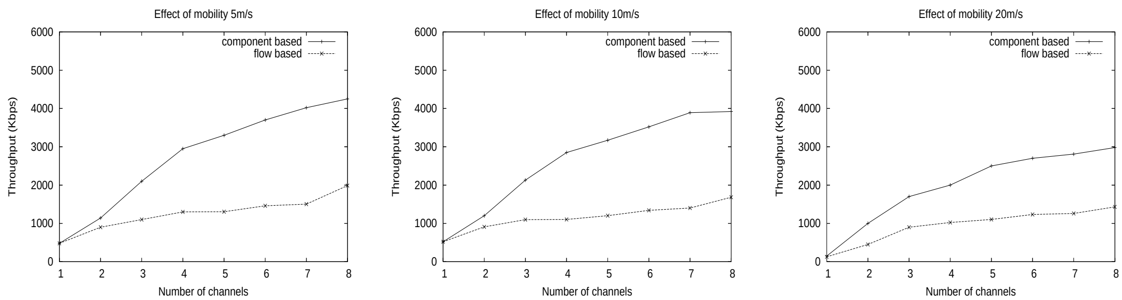


(a) 2Mbps (b) 10Mbps (c) 54Mbps

Figure 10: Average end-to-end delay

3.7.4 Effect of Mobility

We now look at the effect of node mobility on the throughput characteristics (Figure 11). For the component-based approach, in the event of route failures due to mobility, a procedure similar to the route maintenance phase described in Section 3.6.2 is performed. We do not present mobility results for link-based, as handling route failures becomes non-trivial in the case of MMAC. For flow-based, an approach similar to that adopted for component-based is adopted at the granularity of a flow. First, we observe that the throughput is reduced with increasing node speeds for both the flow-based and component-based schemes. This is because of more route failures and a subsequent waste of time for new route computations. Further, the results show that even in the presence of node mobility, the component-based approach yields a higher aggregate throughput when compared to the flow-based approach.

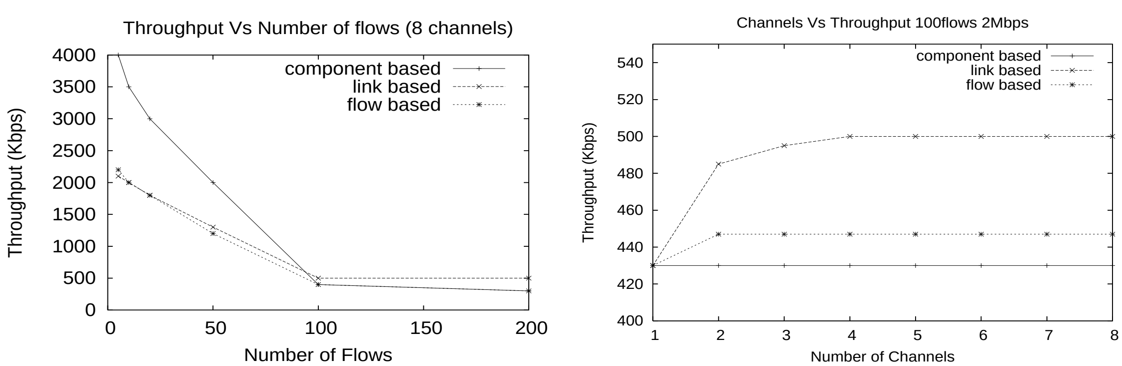


(a) Max speed 5m/s (b) Max speed 10m/s (c) Max speed 20m/s

Figure 11: Effect of mobility

3.7.5 Heavy Load

In Section 3.4, we discussed the impact of a small number of flows for different channel rates. Here, we consider the impact of a varying number of flows (with emphasis on a large number of flows) on all three strategies for a 2 Mbps channel rate. Figure 12 (a) shows the variation of the aggregate throughput, in an eight-channel network, with large number of flows. In all the cases, the aggregate rate of all the flows is kept constant, *i.e.,* 20 flows at 400 Kbps (aggregate of 8 Mbps), 50 flows at 160 Kbps each, and so on. We observe that as the number of flows increases, the aggregate throughput for all three approaches decreases. The general nature of this decreasing



(a) Flows vs Throughput (b) Channels vs Throughput

Figure 12: Effect of Large Number of Flows

trend in aggregate throughput is primarily because of the distributed inefficiencies of the CSMA/CA approach [32]. For a small number of flows, the component-based approach performs better than the link-based and flow-based approaches because of the reasons identified in Section 3.4. However, when the number of flows is larger than 100, the component-based approach yields only one component and effectively utilizes only a single channel. In this case, the link-based approach performs slightly better because, a few links in the same contention region that do not share a common node, can be scheduled in a different channels at the same time. Flows which have a very few hops are the ones that can be most benefit from such scheduling. These flows however cannot be always active as they might still share a node with other flows and hence the intersecting nodes have to switch channels. Thus the overall improvement of link-based approach in such scenarios is not very high. Indeed, we observe from Figure 12 (b) that shows the throughput for 100 flows in the network, we observe that the improvement of link based approach is only with two or three channels. After three channels, the improvement saturates. The flow based approach does not show a huge improvement over CBCA because all links of a single flow have to be on a single channel and with a huge number of flows all links end up using only one or two channels. However, the absolute channel utilization is quite low for these scenarios where there is a large number of flows. For instance, with 100 flows the aggregate throughput observed for the link-based approach is 500 kbps while the total capacity available is 16 Mbps (8 * 2 Mbps/channel), which translates to a very poor channel utilization of only 3.125%. Since it is not desirable to operate the network at such low utilization, the perceived benefit in using link-based and flow-based approaches over component-based is less significant.

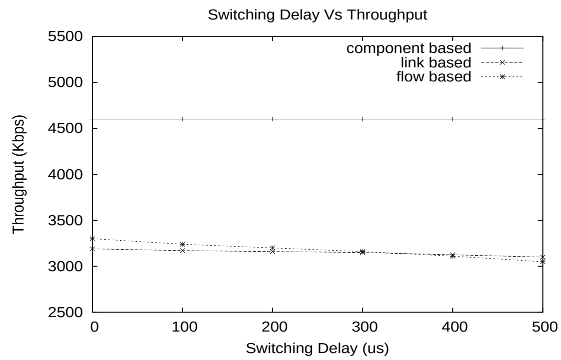
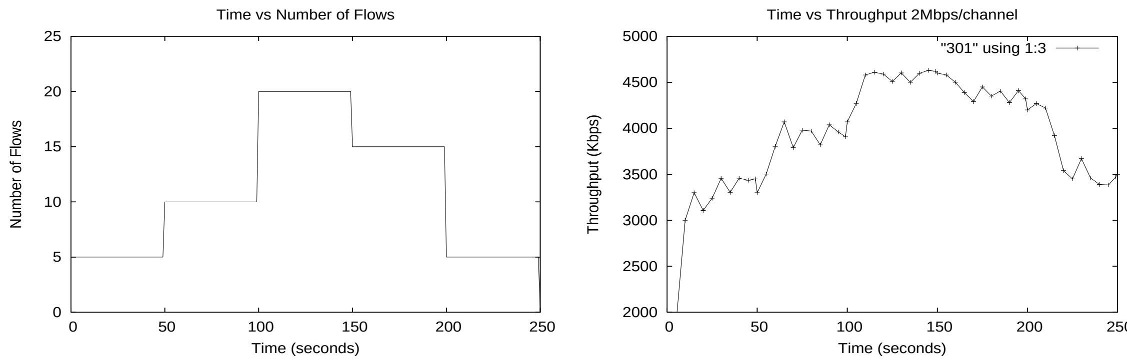


Figure 13: Effect of Switching Delay

3.7.6 Sensitivity to Switching Delay

To study the effect of switching delay, we use simulations for a specific scenario of 20 flows in a 750mx750m grid of 100 nodes (refer Figure 13). Each channel has a capacity of 2Mbps. We vary the switching delay from 0 to 500 μs. As expected, the throughput of CBCA is not affected by switching delay, since this approach prevents any switching during the actual data transfer. All the switching occurs only during the routing phase. The link based approach is expected to be the most affected with switching delay, as the intersecting nodes need to switch very often. However, the MMAC approach that we use as the link based approach masks the effects of the switching delay on throughput using the ATIM window. Every node spends a fraction of time (known as the ATIM window) on a common channel negotiating the channels for the next epoch of time. After the ATIM window, all nodes switch the negotiated channel at the same time and data communication takes place. Although, there is slight drop in throughput for large switching delay as the nodes spend time in switching as soon as the ATIM window is over. The flow based approach is affected by switching delay and throughput drops by as much as 14%.



(a) Number of Flows (b) Throughput

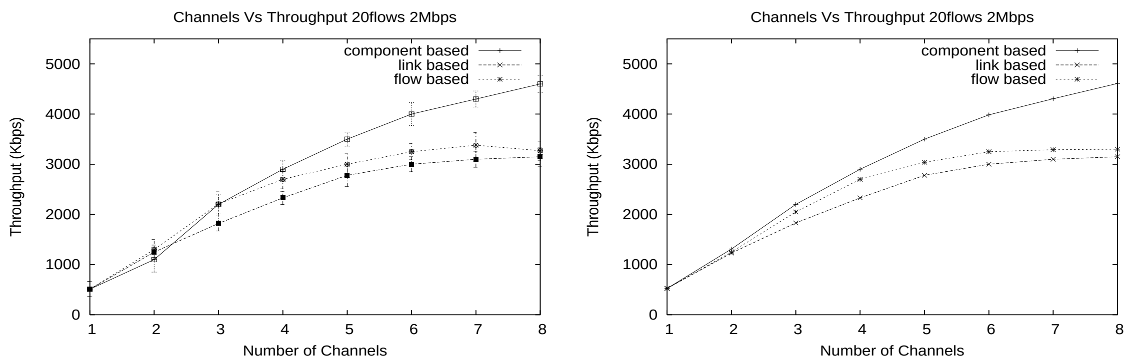
Figure 14: Effect of Dynamic Flows

3.7.7 Instantaneous Capacity

From earlier results in Section 3.4.2, we observed that the aggregate throughput changes with different number of flows. To study the behavior of CBCA with dynamically arriving and departing flows, we simulate a 100 node scenario in a 750mx750m grid with varying number of flows at different time instants and measure the instantaneous throughput. We vary the number of flows every 50 seconds. Figure 14 (a)shows the number of flows active as a function of time. Figure 14 shows the instantaneous throughput for the same scenario. We observe that during the transition period just after the number of flows change, the instantaneous throughput increases or decreases depending on the number of flows. It takes about 5 to 10 seconds for the throughput to reach the full possible value. Although, it takes a finite amount of time for the throughput of new additional flows to stabilize, we note that the throughputs of existing flows are not unduly affected.

3.7.8 Number of Seeds for Averaging

All the graphs (other than instantaneous throughput graphs) presented in this work show average values for simulation runs of 10 seeds. We now present a representative simulation of 30 seeds for a scenario of 20 flows in a 750mx750m grid of 100 nodes with 2Mbps/channel capacity. Figure 15 shows the two aggregate throughput for



(a) 10 Seeds (b) 30 Seeds

Figure 15: Seeds Used for Averaging

both 10 and 30 seeds. We also show error bars for both the graphs. The error bars indicate the standard deviation of the values for each averaged point on the graphs. We observe that some of the discrete jumps in flow based approach are not visible in the graph with larger number of seeds. However, we note that the curves with averaging for 10 seeds are still within a standard deviation of result for 30 seeds.

3.7.9 Greediness of Algorithm

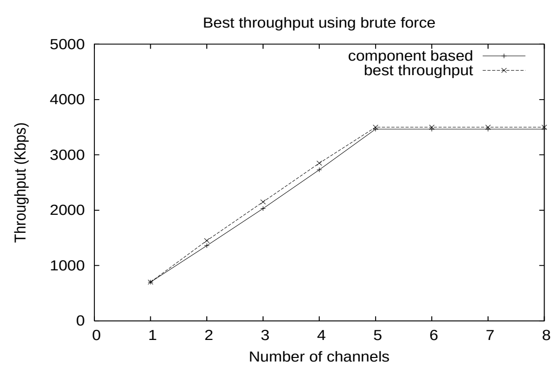


Figure 16: Greediness of CBCA

In Section 3.6, we provided a greedy centralized approach for path-selection and channel assignment in a single-radio multi-channel wireless network using CBCA and later extended the centralized algorithm in to a distributed one. The authors in [27]

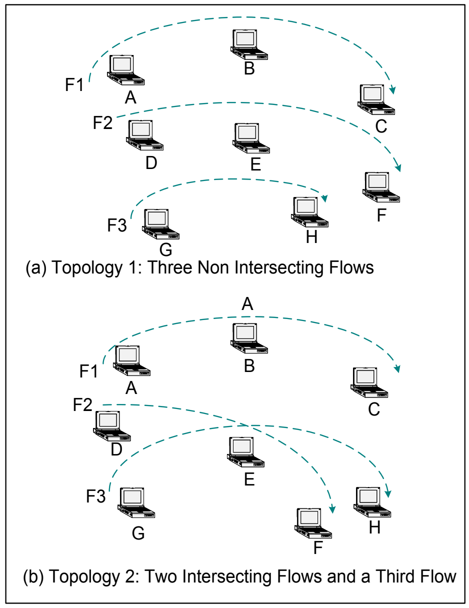


Figure 17: Testbed Scenarios for Comparison of component-based, flow-based and Single Channel

proved that the routing problem in a single-radio multi-channel environment is NP-Hard. Hence we presented a greedy approach. We now present an illustrative example to show the performance of the greedy algorithm. We use a 100 node topology in a 750mx750m grid with multiple 2Mbps capacity channels. We use 5 flows in the network. We compute ten shortest paths for each flow and cycle through all possible 10 shortest paths for each flow and simulate the throughput for the entire network. In Figure 16, we present the throughput of CBCA and compare it with the best throughput of the brute force simulation for each value of number of channels. We observe, that CBCA performs within 95% of the best performance.

3.7.10 Testbed Implementation

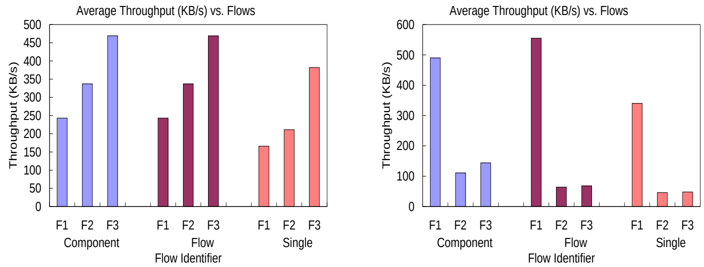
3.7.10.1 Setup

The testbed consists of 8 IBM and DELL laptops. For both scenarios shown in Figure 17, the source and destination nodes are equipped with Lucent Orinoco 802.11b Wi-Fi cards. Three of these laptops have Fedora Core Linux OS, and the remaining five run on Windows XP. We consider two testbed scenarios as shown in Figure 17. For single hop flows, the source and destination nodes are configured to the same $SSID$. For multi-hop flows, we configure two of the Linux laptops as forwarders by enabling IP_V4 forwarding. *The forwarding nodes are equipped with Intel Pro Wireless 2200 802.11 b/g cards.* The routing tables of the source and destination nodes of each flow are configured to allow for host-specific routing. As in the single hop case, the source, destination and the forwarder are all in the same $SSID$. The source nodes for all the flows act as ftp servers and the destination node establishes a ftp connection with the server using winsock utility.

Figure 17 (a) illustrates a topology, where there are three non-intersecting flows, two of which are 2-hop flows. The third flow is a one-hop flow. In this scenario, in the single channel case, all the flows operate on the same channel. Here, both flow and component-based approaches yield the same channel assignment, and each flow is set to a different channel. Figure 17 (b) illustrates a topology, where there are two intersecting flows, and a third non-intersecting flow.

To implement a flow-based approach, we perform periodic switching at the forwarder between the two channels assigned for each flow at intervals of 10s. This time is dependent on the practical switching delay from one channel to another. To determine this switching delay, infinite number of ping messages were transmitted from one of end nodes (D,E,F,G,H) to the forwarder node, E at a constant rate of 10ms. The switching interval was increased from 100ms until the first ICMP message was received. We observed this time to be around 900ms. This 900ms is the practical switching delay, which includes hardware switching, and software updates required to receive ICMP messages. However, for the FTP connection to remain stable, the switching delay had to be much larger, and was determined to be 10 sec.

To implement a component-based approach, we identify the different connected components in the network and assign different channels to them. For single channel



(a) Topology 1 (b) Topology 2

Figure 18: Average Throughput for Component, Flow and Single Channel for the Two Testbed Scenarios

assignment, all the flows operate on the same channel, while for flow-based, each flow is assigned to a different channel and *periodic* switching is performed at intersection nodes. For both topologies, we observe the average throughput for each flow for downloading a 500 MB file. The results are averaged over 5 runs.

3.7.10.2 Results

Figure 18 (a) shows the average throughput in KB/s of three flows using component, flow and single channel assignment for topology 18 (a). In this scenario, component and flow-based assignment yield the same channel assignment and hence the performance of these two approaches are the same. So, for this topology, we compare the component-based throughput with a single channel throughput. The aggregate throughput of all three flows using a component (and flow) based approach is 1049 KB/s, and that of a single channel is 758 KB/s. The improvement in using multiple channels is only 1.4 as opposed to an ideal case of 3. This result corroborates an earlier observation in [5], where they have observed that the different sub-channels in 802.11b overlap to a certain degree. Aside from this observation, the different channels that were used in this scenario also had different background load conditions that varied with time. Also, we had selected the best channel (with the least background load) for the single channel scenario. For both component-based and single channel, the average throughput for a single hop flow is about 1.8x that of two hop flows. This degradation in throughput for multi-hop flows is due to self contention.

Figure 18 (b) shows the average throughput in KB/s of three flows for topology 3.7.10 (b). Here, component and flow-based assignment yield different channel assignments. Here, flows $F2$ and $F3$ are assigned to the same channel in component-based, while they are on different channels in flow-based. $F1$ is on a separate channel in both scenarios. The aggregate throughput for component-based is 745KB/s, while that of flow and single channel are 685KB/s and 431KB/s respectively. The improvement of component-based over flow-based and single channel are 1.1x and 1.7x respectively. However, for flows $F2$ and $F3$, the component-based assignment is 1.95x and 2.6x that of flow and single channel assignment.

3.8 Conclusions

In conclusion, we have considered the channel assignment problem in single-radio multi-channel mobile adhoc networks. Specifically, we have investigated the *granularity of channel assignment decisions* that gives the best trade-off in terms of performance and complexity. We have identified a new granularity for channel assignment that we refer to as *component-based channel assignment* that is simple and has impressive practical benefits. The theoretical performance of the component-based channel assignment strategy does not lag significantly behind the optimal performance, and when coupled with its several practical advantages, it significantly outperforms other strategies under most network conditions. This work resulted in a publication in Mobicom 2006 [34] and an acceptance with major revisions in IEEE Transactions on Mobile Computing [8].

CHAPTER IV

MULTI-RADIO MULTI-CHANNEL AD-HOC NETWORKS: CHANNEL ASSIGNMENT AND ROUTING

4.1 Introduction

In the previous chapter, we looked at the best granularity for channel assignment in a single-radio, multi-channel environment. We now relax the requirement of a single-radio per node and allow individual nodes to be equipped with multiple radios or interfaces. In networks with such multi-radio nodes, a good channel assignment strategy is required to utilize the capacity of all channels efficiently. The problem has been answered at various levels by related works [2, 5, 27]. However, the protocols proposed in some of these works are based on direct extensions of a single radio, single channel adhoc network architecture [5, 27], where a flow between a single source and destination node uses a single path to route all the traffic, while others [2, 38, 24] are based on complex centralized linear programming algorithms that are not practical to implement. This work introduces a new protocol known as Lattice routing, that is shown to effectively combat capacity related issues in such multi-radio, multi-channel environments. It is a completely distributed protocol, and uses local information from neighboring nodes to adapt the paths.

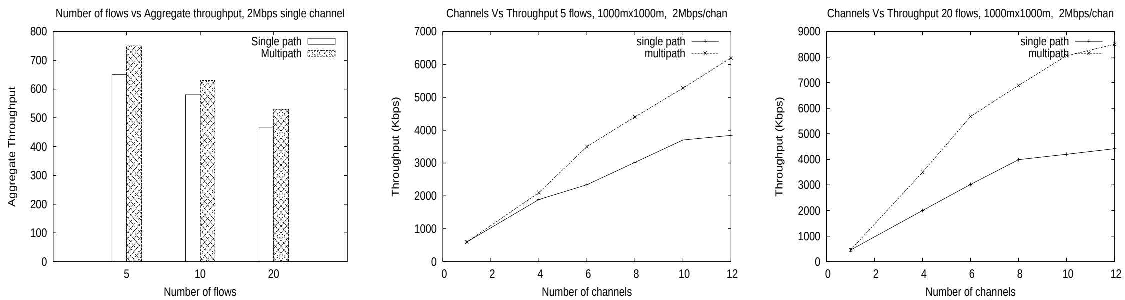
A bottleneck, identified in this work as the *interface insufficiency bottleneck*, arises in such networks. The bottleneck, as the name suggests, is the result of insufficient number of interfaces at the intermediate nodes in a single path for a flow between a source and destination node. This bottleneck reduces the throughput capacity of the network. One way to alleviate this bottleneck, is to use multiple paths (or routes) for each flow. Multi-path routing is not new in adhoc network literature

[15, 22, 37, 20]. However, these works focus on multi-path routing for reasons of reliability, QoS requirements, or load balancing. The current work, on the other hand, focuses on improving aggregate end-to-end throughput in multi-radio multi-channel networks. We call the routing in such networks as 4D routing, because multi-pathing is a two dimensional problem in space, and the multiple radios and channels provide two additional dimensions for routing.

In single channel environments, paths are assigned at the start of every flow, and this is sufficient, because the underlying MAC layer takes care of scheduling the transmissions belonging to the different flows in the network. Given the requirement to form multiple paths for every flow, a fundamental question arises, *"is it sufficient to assign multiple paths at the start of the flow?"* The answer to this question turns out to be *no*. This is because, routing and scheduling of packets cannot be decoupled in multi-radio multi-channel environments. The formation of paths must be an informed decision based on existing traffic conditions, at various stages of the flow.

Based on the above observations, a new protocol called Lattice routing is proposed in this work, that uses multiple paths for every flow (when possible), and is also dynamic in adjusting the various paths based on changing traffic conditions. The protocol works as a cross layer solution for rate control, and routing. It uses a back pressure based algorithm[29] for rate control. The architecture can be realized as a completely distributed protocol, with the individual nodes taking purely local decisions about forwarding data. Each node uses the following principle to route traffic to a destination node: *"serve as much as possible"*. Nodes can forward traffic through any or all of the possible paths, to a destination, to serve the traffic they receive from a previous node.

Extensive ns-2 based simulations show the benefits of the proposed architecture, as opposed to related work on multi-radio, multi-channel adhoc networks. Results indicate, that Lattice routing can adapt to a wide variety of network conditions, such



(a)Single channel performance (b) Throughput for 5 flows (c) Throughput for 20 flows

Figure 19: Aggregate Throughputs for Single Channel and Multi-Radio, Multi-Channel networks

as density of nodes, number of flows etc. Since Lattice routing is proposed as a cross layer solution, between routing and rate control, it can be used with existing off-the-shelf components such as 802.11 a/b/g cards, using pure software modifications. Further, the architecture has an added benefit of security against eavesdropping attacks, because of its inherent multi-path characteristics.

4.2 Motivation

In this section, the proposed Lattice routing architecture will be motivated using three important requirements in multi-radio multi-channel Ad-Hoc networks. These requirements are used to design the protocol, that realizes the proposed architecture.

4.2.1 4D Routing

In a multihop flow from a source node to a destination node, the source and destination nodes have only one function of either transmitting packets or receiving packets, while the forwarding nodes in between have two functions of both transmission and reception. If the forwarding node has a single radio, it cannot transmit and receive data at the same time, because of the half-duplex nature of the radios.

If each node in the flow has two radios, then in a single path (assuming only the top path in Figure 20), the intermediate nodes can utilize one radio each for transmission and reception. But, the source and destination node will have an additional

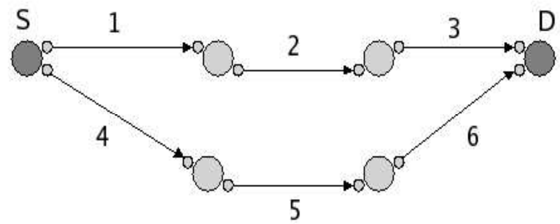


Figure 20: Illustration of 4D routing

radio that cannot be utilized. Thus, the intermediate nodes experience an *"interface insufficiency bottleneck"*. If W is the capacity of a channel, then assuming each link on a different channel, it can be shown that such a single path can achieve a flow throughput of W. The additional radios at the source and destination can be utilized, if an additional path can be formed, as shown in the figure. In this case, the flow using two paths can result in a throughput of $2W$. Thus, for every radio, that a source and a destination node provide for a path, every intermediate node should be able to provide two radios for each path to fully utilize the capacity. We refer to a path as a set of nodes from source to destination, irrespective of the number of radios each node provides for the path. So, if all the nodes in the network have the same number of radios (say r), then a single path between source and destination will have a capacity of rW. However, this will leave $r/2$ radios free at both source and destination node. A second path will achieve the maximum capacity of $2rW$ for the path. We call such a multi-path routing in multi-radio multi-channel networks as 4D routing, because multi-pathing is a two dimensional problem in space, and the multiple radios and channels provide two additional dimensions for routing.

The above discussion was for a single source-destination pair. An adhoc network will likely have a number of such source destination pairs. From the above discussion, it is clear that every source destination needs only two paths to achieve maximum capacity. This requires every intermediate node, in every path, to provide all its radios exclusively for one path, between some source destination pair. However, in the presence of multiple source destination pairs, paths can intersect each other. When paths intersect each other, the nodes at the intersections will not be able to provide all their radios. exclusively to a single path as required. In such a scenario, additional paths can be added for the source destination pairs to add capacity. Thus, multiple paths may be required for every source destination pair to get maximum capacity out of the network.

On similar lines, one might argue that multi-pathing should benefit single radio, single channel environments as well. There are several related works [15, 22, 37, 20], that deal with multi-pathing in single channel adhoc networks. However, in single channel environments, hops belonging to the flow using multiple paths that are close to the source node (or the destination node) contend with each other and only one such hop can operate at any given time. If the nodes in Figure 20 are operating on a single channel, then hops 1, 2, 4 and 5 contend with each other and only one of them can operate at a time. This reduces the benefit of multi-pathing in single channel environments. In fact multi-pathing does not help in the presence of multiple flows either. Figure 19 (a) shows ns-2 based results of aggregate throughput in a single channel adhoc network using both single path and multi-path routing. The results are for a network of size 1000mx1000m with 150 nodes in the network and the channel datarate of 2Mbps. The results show little improvement in aggregate throughput by using multiple paths. All the related works focus on multi-pathing for reasons other than increasing aggregate throughput.

To study the impact of multiple flows, ns-2 simulations are used. Figures 19 (b) and (c) show the performance benefits of 4D routing under different number of flows. As before, the results are for a network of size 1000mx1000m with 150 nodes in the network and a per-channel datarate of 2Mbps. Each node has 4 radios and the number of available channels are changed. The channel assignment strategy is based on the proposed protocol. The 4D routing results are for the proposed architecture.

The single path results are for a related work known as MCR [14]. It is found, that 4D routing gives substantial improvements (50-100%) in aggregate throughput, when compared to a single path routing. When there a large number of flows, the default shortest path routing leads to a number of common nodes in the network. The common nodes have to use the same radios for multiple links, and thus become bottlenecks in the network. 4D routing, on the other hand, distributes the flows across the network. Thus 4D is beneficial even when there are a large number of flows.

4.2.2 Micro-Decoupling

Consider a toy topology of two flows as shown in Figures 21 (a) and (b). In Figure 2 (a), both the flows intersect at a common node while in the other the flows use different intermediate nodes. In either scenario, there are four different links in the network. In a single channel environment, all the four different links in either scenario contend with each other and at any given time, only one of the four links can operate. The underlying MAC layer will ensure, that each of these links get equal opportunity to operate. Thus, the aggregate throughput in either scenario will be identical.

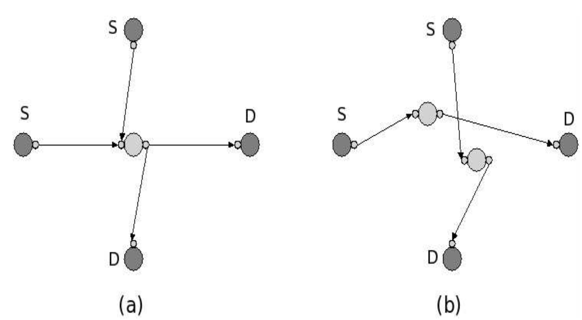


Figure 21: Illustration of Micro-Decoupling

However, in a multi-channel multi-radio environment (assuming two radios per node), the two scenarios will result in different throughput. In the first scenario, the common node in the middle has to serve four different links, but it has only two radios. So, even if the two radios operate on different channels, only two of the links can operate at the same time. The second scenario, on the other hand, will result in higher throughput, because each link can now use a different channel, as every node has at most 2 links (and hence its two radios can be assigned to each of the links). All the four links can be used at the same time. Thus, the routing protocol should be aware of existing traffic patterns and be able to decouple paths, such that throughput is maximized. We call this ability of the routing protocol, to decouple paths, as micro-decoupling. When a new flow is added to the network, the multiple paths that will be chosen for the new flow have to be aware of all the existing paths.

4.2.3 Rapid-Rerouting

Figure 3 (a) shows two non intersecting flows as in previous case. Now, if one of the flows finishes its transmission after some time, the scenario will be as in Figure 22 (b). In a single channel environment, the level of contention on the links belonging to the remaining flow reduces. This results in an increase in throughput, of the remaining flow. In a multi-channel environment, such a situation leads to a scope for an additional path to be formed between the source and destination nodes of the existing flow, as in Figure 22 (c), using the idle node and additional radios at the source and destination node. Thus, capacity for existing flows can be increased, if existing flows are rapidly rerouted, when some flows cease to exist.

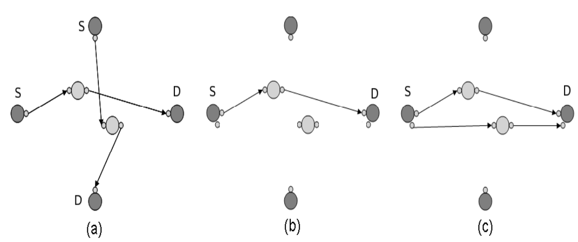


Figure 22: Illustration of Rapid-Rerouting

It can be argued, that such rapid-rerouting can be helpful in any adhoc network. It is possible, to find scenarios in a single channel environment, in which a flow that becomes inactive can allow better new paths for existing flows. These better paths can result in higher throughputs. In a single channel environment however, nodes are the only available resources for rapid-rerouting. In a multi-radio multi-channel environment, nodes, radios and channels are available resources. The available resources increases to the product of the number of nodes, radios and channels. These available resources, present a huge opportunity for dynamic re-allocation.

4.2.4 Interrelationship Among the Requirements

Given the above requirements of 4D routing, micro-decoupling and rapid-rerouting, the natural question that arises, *"Is it possible to use or adapt existing multi-path schemes proposed for single channel Ad Hoc networks ?"* The related works ([15, 22, 37, 20]) do multi-path routing for different reasons like reliability, QoS requirements, or load balancing. None of these meet the current requirements. Further, all these works consider only node-disjoint paths, from source to destination node. However, a node might be serving multiple flows, and thus may not be able to provide two radios, for each path (requirement for multi-pathing as described above). In such a case, the node previous to this particular node can initiate a multi-path (as in Figure 23). This means, that it is not enough to just pick complete node-disjoint paths and use them. This leads, to potentially a large pool of paths, to pick from, and a source routing technique, where the source picks the necessary paths, is prohibitive. Further, owing to the micro-decoupling requirement, channel assignment has to be done, in conjunction to picking the multiple paths. This means, that local decisions have to be made depending on existing traffic conditions at a node. Since the radios can potentially switch across multiple channels, it might be possible to use the same radio, to operate on multiple channels, at different time instants. However, the state-of-art radios have a finite switching delay [34], that makes it necessary to switch the radios as few times as possible.

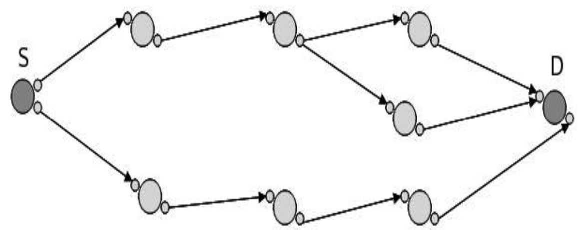


Figure 23: Illustration of Lattice Routing

4.3 Lattice Routing

Based on the above observations, a new protocol ('Lattice Routing') is proposed, in this work that uses 4D paths for every flow (when possible), and is also dynamic in adjusting the various paths, based on changing traffic conditions. The protocol is completely distributed, and has a very low complexity overhead on individual nodes. The core component of the protocol is the control channel. Given C channels that can be used, one channel is assigned as the default control channel. Every node has one of its R radios ($R <= C$) always on the control channel. The control channel prevents the "multi-channel hidden terminal problem" [31]. The control channel is also used to serve data when it is not used for control messages. The other $(R - 1)$ radios can switch between the remaining $(C - 1)$ channels. The radios have a non-negligible switching delay, when they switch from one channel to another. To prevent the impact of switching delay, the protocol uses very few switches. There are two phases of the algorithm: (1) a multi-path route detection phase, which identifies possible multiple paths from source to destination; and (2) a data transmission phase, which selects traffic aware routes determined in the first phase and performs rate control, to allow micro-decoupling and rapid-rerouting.

4.3.1 Multi-Path Route Detection

A distance vector multi-path routing scheme is proposed, in order to provide routing tables, to each node in the path of a flow. At the end of the routing phase, the source

```
Variables:
src_ID: ID of Source node
dst_ID: ID of Destination node
int_ID: ID of intermediate node
cuur_ID: ID of current node
ttl: Time to live
next_hop: ID of next hop in route
hops_to_dst: Shortest number of hops from next_hop
RREQ: Route Request
Fields in RREQ: src_ID, dst_ID, int_ID's, ttl
p: Probability of forwarding RREQ
RREP: Route Reply
Fields in RREP: src_ID, dst_ID, int_ID's
Routing table format:
src_ID dst_ID next_hop hops_to_dst

Functions:
Send_RREQ(dst_ID) {
  Fill src_ID, dst_ID and ttl of
    RREQ packet ;
  Send RREQ on control channel ;
} //Send Route Request Packet

Forward_RREQ() {
  ttl = ttl - 1;
  If ((ttl > 0)&&(random_number < p)) {
   new int_ID = curr_ID;
   Send updated RREQ;
  } else {
   Drop RREQ packet; }
} //Forward Route Request Packet

Send_RREP(RREQ) {
  Fill src_ID, dst_ID, and int_ID's of
   RREP packet using reverse path in RREQ;
  Send RREP on control channel;
} //Send Route Reply Packet

Recv_RREP() {
  If entry for next_hop does not exist {
    Create new routing table entry;
  } else {
    Update routing table entry for hops_to_dst; }
  If (curr_ID != src_ID) {
    Forward RREP to next node in path; } }
} //Receive Route Request Packet

Routing Process
RREQ propagation
Source Node: Send_RREQ(dst_ID)
Intermediate nodes: Forward_RREQ()

RREP propagation
Destination Node: Send_RREP(RREQ)
Intermediate Nodes: Recv_RREP()
Source Node: Recv_RREP()
```

Figure 24: Routing Phase

node of a flow and intermediate nodes in the multiple paths to the destination build routing tables. The routing tables contain entries of possible next hop nodes, to reach the destination, along with the shortest hop length from that next hop. The actual hop length, through a next hop, is not known, as the next hop may decide to send data along any of its known paths. Loops are prevented from occurring in the paths. All the paths identified by the routing phase may not be used, for actually routing traffic. Rather, a subset of traffic aware routed would be used. The distance vector approach also allows intermediate nodes to change routes, for allowing the dynamic rearrangement of paths.

The pseudo code of the multi-path routing is given in Figure 24. All the routing messages are sent on the control channel, so that all nodes in the network can participate in the routing process. When the source node of a flow wants to send data to a destination node, it first broadcasts a route request message (RREQ) on the control channel. The RREQ message has a time to live field (ttl), that can be set to find all possible paths with a hop count of less than or equal to ttl. Each node, upon receiving a RREQ packet, rebroadcasts the packet, with a certain probability, to its neighbors after adding its own node address on the RREQ, if the ttl counter is still valid. A RREQ message will be dropped by a node, if it finds that its address is already part of the message. This prevents routing loops. A single node might however rebroadcast multiple RREQ messages belonging to the same source destination pair. This allows multiple possible paths to be found. However, allowing every node to rebroadcast every RREQ will lead to broadcast storms. Probabilistic dropping of the RREQs, by intermediate nodes, is used to prevent broadcast storms. This is achieved by defining a probability p, with which the intermediate nodes drop RREQ packets they receive. If the nodes are uniformly distributed in the network, the number of RREQs received by a node will increase as its distance (d) from the source node increases. This is because, there will be more nodes at a larger distance. Hence, we choose $p \propto 1/d$.

When the destination node receives the RREQ messages, it sends out route replies (RREP) to each of the RREQs. The RREP is sent, using unicast on the control channel, along the reverse path of the corresponding RREQ message. Routing table entries are created in the nodes, as the RREP propagates back to the source node. A single routing table entry has the following fields: *src_ID*, *dst_ID*, *next_hop*, *hops_to_dst*. The *src_ID* and *dst_ID* are the addresses of the source and destination nodes respectively. The *next_hop* is the address of the next hop to the destination node. The *hops_to_dst* field indicates the shortest hop length to the destination from the next hop. This field will be updated as more RREP messages pass through a node. A new routing table entry is created, for every new next hop node to the destination.

Once the RREP messages reach the source node, the source can start sending data packets. At the end of this routing phase, the source node should ideally have entries corresponding to all possible next hops, to the destination, that satisfy the ttl condition. However, since the RREQ messages are probabilistically dropped, some paths may not be found. Further, the *hops_to_dst* field may not have the shortest hop count to the destination node from the next hop. The memory footprint on each node to maintain the routing tables is not high. This is because, each node has to maintain routing table entries, that contain only next hop information, for a particular destination. The number of entries for a single destination, cannot grow beyond the maximum node degree (number of immediate neighbors) of the network.

4.3.2 Data Transmission Phase

The previous phase allowed the nodes to identify the possible multiple paths, from source to the destination. In this phase, the exact 4D paths are found, and data packets are sent along. The pseudo code for the data transmission phase is given in Figure 25. Three kinds of control messages are designed to aid the data transmission. These messages are used to select traffic aware routes, from the pool of paths, that are identified as a result of the routing process. Further, these messages aid in dynamically re-allocating routes, when traffic conditions change.

- HELLO message - Each node periodically broadcasts its perceived available rates. The perceived rate on a channel is a rough estimate of how much traffic the node can sustain in the channel. This message is a one hop broadcast on control channel. The period of transmitting the HELLO message ($HELLO_{int}$) is defined empirically. A value of 2 seconds was empirically found to be sufficient. The rationale behind such a value of the $HELLO_{int}$, is that it should be small enough to provide recent information about perceived rate and large enough to prevent a huge overhead on the control channel. If a node has a radio

Variables:

(IR_i, c_j): Incoming rate at node i, channel c_j
(OR_i, c_j): Outgoing rate at node i, channel c_j
(AR_i, c_j): Available rate at node i, channel c_j
(NR_i, c_j): New requested rate on channel c_j
R_i: Total number of radios at node i
r_{ik}: One of the R_i radios at node i
C: Total number of channels at node i
$HELLO_{int}$: HELLO Refresh interval
$ALGO_{int}$: Refresh interval for on-the-fly algorithm
(ir_i, f): incoming rate at node i for flow f in previous $ALGO_{int}$
$(or_{i,l}, f)$: outgoing rate from node i to node l for flow f
SORRY Message format: *src*, *dst*, (NR, c_j)

Functions:

Comp((AR_i, c_j)){
  If (some r_{ik} on c_j) {
    (AR_i, c_j) = Totalrate for channel c_j - ((IR_i, c_j)+(OR_i, c_j)+interference);
  } else {
    $(AR_i, c_j) = Average(AR_l, c_j)$;
  } // *l=all neighbors of node i*
} // *Compute available rate at node i,*

Send_HELLO() {
  for all c_j
    comp((AR_i, c_j));
  broadcast available rate for all c_j on control channel
} // *Send HELLO message*

Send_SORRY(l,(NR_i, c_j)) {
  $dst = l$; $src = i$; Fill (NR, c_j) ;
  Send SORRY message to l on control channel;
} // *Send SORRY message*

Recv_SORRY() {
  reduce $(or_{dst,src}, f)$ to NR;
  Accommodate(excess $(or_{dst,src}, f)$) ;
} // *Receive SORRY message*

Accommodate(excess($rate, f$)) {
  lookup entries for f in routing table;
  give probabilities(p_n) to possible all *next_hop* based on *hops_to_dst*;
  While ((excess($rate, f$) > 0) and (radios available at some *next_hop*)) {
    randomly pick a *next_hop* based p_n
    find the best channel (c_{best}) using (AR_i, c_j) and (AR_{next_hop}, c_j);
    find radio of current node (i) and *next_hop*;
    allocate $(or_{i,next\_hop}, f) = min((AR_i, c_{best}), (AR_{next\_hop}, c_{best}))$;
    excess($rate, f$) -= $(or_{i,next\_hop}, f)$; }
  If ((excess($rate, f$) > 0) and i is not source of f)
    Send_SORRY(previous hop,(excess($rate, f$),channel)
} // *Accommodate the excess rate*

Lattice routing process at each node i
After every $HELLO_{int}$: Send_HELLO() ;
After every $ALGO_{int}$:
 For every f through i {
   find excess(ir_i, f) ;
   Accommodate(excess (ir_i, f)) }
Recv_SORRY() if SORRY message is received ;

Figure 25: Data Transmission Phase

on a channel, the perceived rate is estimated, as the difference between the maximum possible rate on the channel and the amount of traffic in the previous $HELLO_{int}$. The amount of traffic in the previous $HELLO_{int}$, can be calculated by allowing the promiscuous mode of the radio. On the other hand, if the node does not have a radio on a channel, the estimate for that channel is calculated, as the average of the perceived rates of all the one hop neighbor nodes. The HELLO messages need not be synchronized with each other, since only a rough estimate is needed.

- SORRY message - This message is a unicast message sent by a node, to its previous node, along a path of a flow. This message requests the previous node, to reduce its rate of transmission of packets. This message is used to create a back-pressure. The use of the SORRY message will become evident in the following paragraphs.

- Channel-change message - This message is used by a node to request a neighbor node to change its channel for a particular radio. This message results in the other node sending an acknowledgment message back. This acknowledgment prevents channel conflicts, when two competing Channel-change messages are received, for the same radio and only one Channel-change message will be honored.

There are two parts in the data transmission phase, and they are explained below:

4.3.2.1 4D Route Selection and Micro-decoupling

Every node knows the current traffic conditions at its neighbors, on each channel, as a result of the HELLO message. The 4D route selection algorithm, shown in Figure 25, is performed by every node once every $ALGO_{int}$. The algorithm states that every node should follow the principle *"serve as much as possible"*. The algorithm allocates the number of packets to be sent to every neighbor, in the next $ALGO_{int}$, and also the channel, and radio to be used. Like the $HELLO_{int}$, the $ALGO_{int}$ is also determined empirically to be 250 milliseconds. The $ALGO_{int}$ should be small enough, to adapt to the received HELLO messages, and large enough, to prevent the intermediate nodes from spending too many computation cycles, to compute the routes. There are several aspects of the algorithm:

- *Randomized next-hop selection:* The source node tries to accommodate the maximum possible datarate, through the possible next hops, determined by the routing phase. Every intermediate node tries to accommodate the data it received in the previous $ALGO_{int}$, for the next $ALGO_{int}$. Thus, the nodes have to select next hop node(s), to forward the packets. The next hops are selected in a randomized fashion. The rationale behind the randomized selection of the next hop nodes, is that it results in load balancing among the different flows in the network. Similar randomized routing solutions have been proposed in the context of sensor networks [21]. Normalized probabilities are given to each of the possible next hops (i), based their number of hops to the destination ($hops\_to\_dst$). Each of the next hop node is given a value v_i given by the following equation:

  $$v_i = 1/hops\_to\_dst$$

  If we define V as,

  $$V = \sum_{\forall nexthops} p_i$$

  Then, the normalized probability for each next hop can be given by,

  $$p_i = v_i/V$$

Thus, a probability value that is inversely proportional to the distance to the destination node, is given to each of the next hop nodes. These probabilities are used to pick the next hop(s), which will be used to forward the packets. The probabilities result in favoring nodes with shorter hop lengths.

- *Channel and radio selection:* Once a node picks a next hop node to forward packets, it has to chose the channel to be used, and the radios that will be used, at the current node, and the next hop node. To prevent the impact of switching delay, radios will not be allowed to switch during an $ALGO_{int}$. If there is free radio available at both the current node and the next hop node, the free radios will be selected, for allocating the packets. The perceived rates of both the nodes are compared, and the best possible mutual channel is selected, for the radios to operate on. Once a channel is selected, the current node sends a Channel-change message to the next hop node on the control channel. Both the nodes will then switch the selected radios, on the selected channel. If one of the nodes does not have a free radio, the least used radio, for that node, is selected and its current channel is selected. The other node will switch its free radio to the channel. If both nodes do not have a free radio, a common channel where both the nodes have a radio is selected. If such a channel cannot be selected, another next hop is picked and the process is repeated. After the radios and channel are picked, the number of packets that can be allocated in the next $ALGO_{int}$ is decided using the perceived rates of both the nodes on the channel. If there are more packets that have to be allocated, another next hop is picked and the process is repeated. The next hop node, the radios used and the channel are preserved for future slots. Only excess rates will be accommodated in future slots (excess rate is new extra rate received by a node that was not already allocated in a previous slot). This process of channel selection should not result in a deadlock where no radio or channel is found. This is guaranteed because of two reasons: (1) every node has a radio always on the control channel. Thus a common channel can always be found between two nodes and, (2) even if the control channel cannot accommodate any traffic, then a SORRY message will be generated and hence the onus of accommodating traffic is transfered to the previous node. A radio on a node becomes free if it does not serve any packet for a few consecutive slots (the simulation results in this paper use a value of 10 slots to determine that a radio is free). Thus, the 4D routes are selected as a result.

- *Back-pressure:* It may not always be possible to accommodate all the datarate that a node receives. In such a scenario, the node will send a SORRY message back to the previous node, asking it to reduce the rate it sends. The previous node on the other hand will try to accommodate this excess rate through another node using the same process. If this node is also not able to accommodate the excess rate, it will in turn send a SORRY message to its own previous node. The process can continue till the source node receives a SORRY message, when it will decide to reduce its outgoing rate. Thus, a simple back-pressure is created to control the maximum rate that can be served along any path. The reason for using the back-pressure algorithm is its simplicity. The routing protocol has to be run on each individual node that forwards packets belonging to various flows in the network. The use of backpressure algorithms for rate control is not new. The seminal work of Tassiulas et al [29] introduced the concept of backpressure under a control theoretic framework. Also, in [29], the authors describe a multicast bandwidth scheduling scheme using back pressure. As a result of this back-pressure, micro-decoupling of routes is performed and suitable routes are used for forwarding traffic at every intermediate node.

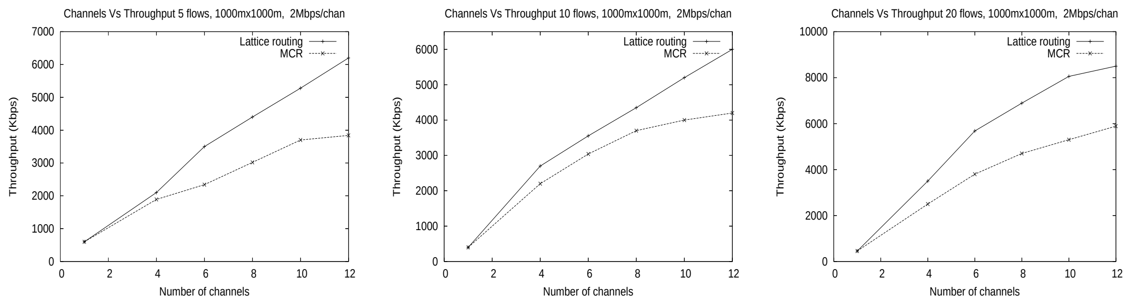
4.3.2.2 Rapid-Rerouting Using Forward Pressure:

It was motivated in Section 4.2, that rapid-rerouting of traffic can make use of new paths as and when they are possible. This will result in better incremental throughput, when possible. The back-pressure results in the source settling for a maximum possible datarate. Periodically, the source node tries to push more data than in previous slots to see if new accommodation is possible. If new paths become available, they can be now utilized.

As a result of performing the above algorithm, every node comes up with a packet allocation strategy, for the next $ALGO_{int}$, during which the node tries to send all the allocated packets. If all the packets are not sent in the slot, they will be carried over for accommodation during the next pass of the algorithm. This ends the discussion of the Lattice routing algorithm. The next section discusses the performance results of the algorithm.

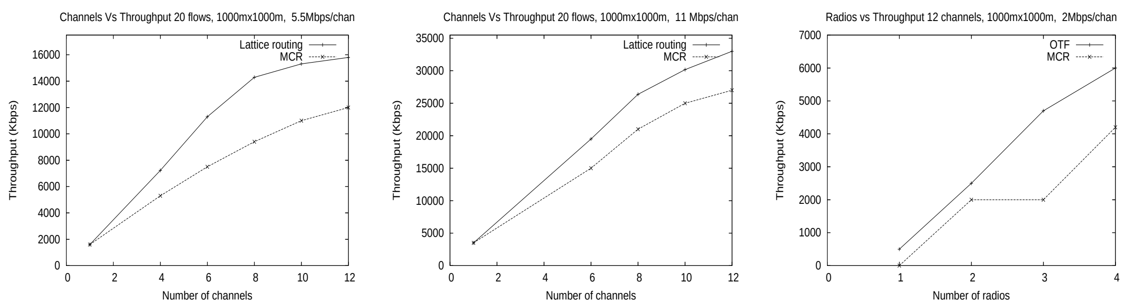
4.4 Performance Evaluation

This section provides the performance evaluation of the proposed Lattice routing multi-pathing architecture. First, comparisons with Lattice routing architecture are provided for the Multi-Channel Routing Protocol (MCR) [14], a related work on multi-radio, multi-channel Ad Hoc Networks. MCR is a fully distributed protocol that uses hybrid channel allocation. Given a set of radios at each node, MCR allocates a few radios as receiving radios that constantly listen on predetermined channels and are used only to receive data. The other radios are used for transmitting data. When a node wants to send data to a neighboring node, it changes the channel of a transmitting radio to one of the channels of the neighboring radio. The protocol uses a single path for every flow. Results are also provided for specific properties of Lattice routing architecture, like time taken for dynamic rearrangement, and effect of number of nodes.



(a) Throughput for 5 flows (b) Throughput for 10 flows (c) Throughput for 20 flows

Figure 26: Average Throughput (Kbps) vs. No. of Channels for Varying Number of Flows for Single path and Multi-path routing



(a) 5.5Mbps/channel (b) 11Mbps/channel (c) Effect of radios

Figure 27: Effect of Datarate and Number of Radios

The simulations were performed using the ns-2 simulator. In all simulations, the size of the network is kept at 1000mx1000m. The nodes have an omni-directional range of 250m. Unless otherwise specified, the number of nodes in the network is set to 150. 802.11 DCF is used as the MAC layer. RTS/CTS are disabled. By default, a 2Mbps per channel datarate is used. But other datarates of 5.5 and 11Mbps are also studied. Unless otherwise specified, the number of radios is set to be 4. All traffic is constant bit rate traffic over UDP. The traffic rates of the flows in the network are kept sufficiently high to saturate the network, i.e., no more increase in aggregate throughput can be achieved by further increasing the flow rates. The performance metrics used to evaluate the protocols are aggregate end-to-end throughput, average end-to-end delay, and delay jitter. Each simulation scenario is repeated for 10 random seeds and results are averaged. The simulations are each performed for 200 seconds.

Source destination pairs are randomly selected. The average shortest path hop length is around 4-5. The parameters used for Lattice routing are as follows: $HELLO_{int}$, the refresh interval between two successive HELLO packets is set to be 2 seconds. $ALGO_{int}$, the refresh interval between two successive instances of performing the Lattice routing algorithm is set to be 250ms. The different nodes in the network are not synchronized with each other.

4.4.1 Effect of Number of Flows

Figures 26 (a), (b) and (c) show the aggregate throughput comparisons of Lattice routing and MCR for varying number of flows. It can be observed that the relative performance improvement of Lattice routing is significantly better than MCR. There are two primary reasons for the better performance of Lattice routing. The first reason is the increase in capacity because of multi-pathing. The second reason is that under heavy load i.e., large number of flows, MCR, which uses single path routing results in nodes, that are common to multiple flows. These common nodes result in a capacity bottleneck. Lattice routing, on the other hand, tries to balance the traffic across nodes. Hence common nodes do not form a significant capacity bottleneck.

4.4.2 Effect of Channel Rates

Figures 27 (a), and (b) show the aggregate throughput for varying channel rates. The channel rates analyzed are 2Mbps, 5.5Mbps and 11Mbps, reflecting realistic 802.11 datarates. It is observed, that the absolute improvement of both the schemes reduces with increasing datarates. In particular, Lattice routing gives a maximum improvement of almost 15 times single channel throughput for 2Mbps channels, while it gives only about 9.5 improvement for 11Mbps channels. This is because, of the $sifs$ and $difs$ intervals used by the 802.11 standard. These values are constant and their impact on throughput is higher at higher datarates, because of low transmission times. In all cases however, Lattice routing gives better results when compared to

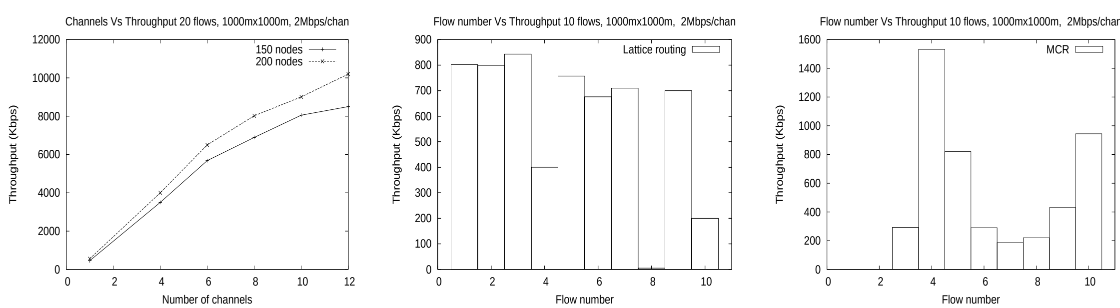
MCR.

4.4.3 Effect of number of radios

Figure 27 (c) shows the effect of number of radios on the throughput capacity. We keep the number of flows to a constant 10 and the number of channels used is 12. We observe that Lattice routing scales well with increasing number of radios. However, MCR works only for even number of radios (this is because MCR gives equal number of radios for transmission and reception). Lattice routing always performs better than MCR.

4.4.4 Effect of Number of Nodes

Figure 28 (a) shows the throughput of Lattice routing for varying number of nodes in the network. The number of flows in the network is fixed at 20. It can be observed that the absolute improvement of Lattice routing is higher for larger number of nodes in the network for the same total number of flows. This is because with larger number of nodes more paths become available for use for the different flows, thus increasing the absolute improvement.

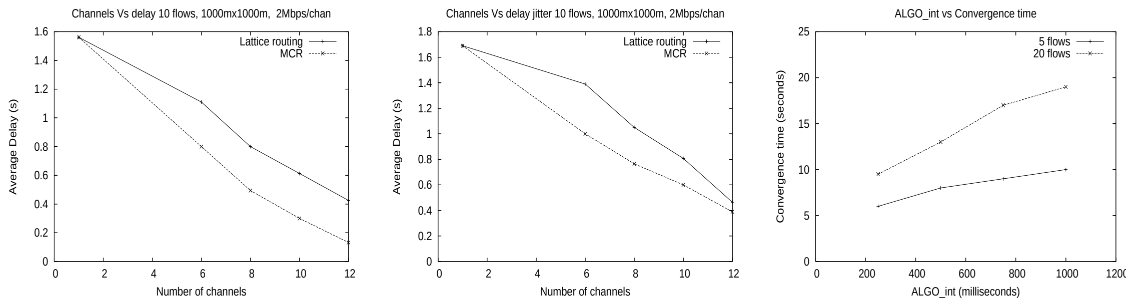


(a) Effect of nodes (b) Per-flow thr Lattice (c) Per flow thr MCR

Figure 28: Effect of Number of Nodes and Fairness Results

4.4.5 Fairness of Flows

Figure 28 (b) and (c) show the per flow throughput for a single case of 12 channels and 10 flows in the network. It is observed that Lattice routing shows a balanced



(a) Average end-to-end delay (b) Delay jitter (c) Convergence time

Figure 29: Time Results

throughput across flows. On the other hand, MCR shows uneven throughput across flows. Lattice routing balances the load because of the inherent multi-path nature. Multiple paths in single channel networks are used to load balance the traffic. It is observed, that this load balancing effect of multi-path routing carries over to multiple channels as well.

4.4.6 End-to-end Delay

Figure 29 (a) shows the average end-to-end delay per packet for Lattice routing and MCR for 10 flows, and 2Mbps channels. Delay is calculated only for those packets that reach the destination nodes. The results indicate that MCR shows a lower end-to-end delay when compared to Lattice routing. Packets experience an initial delay in Lattice routing scheme because rates are allocated in slots of $ALGO_{int}$. Considering one 5 hop path from a source node to destination node, the source node allocates the data in the first slot, the next node allocates the data in second slot and so on. This leads to an initial delay that carries on to other packets of the flow. Hence, Lattice routing experiences a higher end-to-end delay, although this delay is significantly less than that of single channel networks.

4.4.7 Delay Jitter

Figure 29 (b) shows the end-to-end delay jitter of the packets for Lattice routing and MCR. Delay jitter is defined as the standard deviation of end-to-end delay. As in the previous case delay jitter is calculated only for those packets that reach the destination nodes. The results show that delay jitter performance, like the average end-to-end delay, of MCR is lower than that of Lattice routing. In MCR all flows use a single path from source to destination. However in Lattice routing several multiple paths are used for each source destination pair. The multiple paths could potentially be of varying lengths. This leads to varying characteristics of each paths. This results in a higher delay jitter. However, as in the case of end-to-end delay results, the delay jitter of Lattice routing is still significantly lower than that of single channel case.

4.4.8 Convergence Time

Figure 29 (c) shows the convergence times for varying $ALGO_{int}$ for different number of flows. Convergence time is defined as the time taken to converge on a channel assignment over all the nodes in the network when a new flow is added. As described before, $ALGO_{int}$ is the refresh interval, after which the Lattice routing algorithm is performed at each node. A lower convergence time is beneficial as it reduces the switching time and also to maintain consistent throughput for the flows. The results indicate, that lower the $ALGO_{int}$, lower is the convergence time. Further, the presence of a large number of flows leads to higher convergence time. This is because, when there are a large number of flows, many nodes participate in forwarding the traffic, and hence they take more time to converge to a fixed channel assignment. However, traffic is not disrupted during the convergence process.

4.5 Conclusions

This work introduces a new routing protocol for multi-radio, multi-channel Ad-Hoc networks known as 'Lattice routing' that uses 4D routes for the flows and that also dynamically adjusts paths as a result of changing traffic conditions. Extensive ns-2 based simulations show the benefits of the proposed protocol in terms of applicability to various network conditions. Since the protocol is proposed as a solution between the rate control and routing layers, it can be implemented as a software only solution using off-the-shelf hardware.

CHAPTER V

MULTI-RADIO MULTI-CHANNEL LINKS: EFFECTIVE DATA-RATE AGGREGATION

5.1 Introduction

Thus far, we addressed network level problems in using multiple orthogonal channels in an adhoc network. We now shift our focus to the practical link level issue of actual usage of multiple radios on different orthogonal channels. We use IEEE 802.11 WLAN devices (Wi-fi) and pose the following question: *Can devices use the multiple orthogonal channels in wi-fi networks simultaneously to realize a high data-rate wireless link and hence cater to applications requiring high bandwidths?* In other words, given that there are 3 orthogonal wi-fi channels in the 2.4GHz band and 12 orthogonal wi-fi channels in the 5.2GHz band, can a pair of devices each equipped with 15 wi-fi radios use all the available orthogonal channels to achieve a high data-rate wi-fi link?

We believe that such high data-rate wireless links will have use in the multi-radio adhoc networks that we discussed in previous chapters and also in greenfield environments where co-existence with pre-deployed networks is not a concern. Examples of such greenfield networks include enterprise network deployments and wireless backhauls for wireless mesh networks. Furthermore, even in environments that have prior wi-fi deployments, a solution that is fully backward compatible with normal wi-fi links and opportunistically provides high data-rate communication capabilities will indeed be desirable. We term such a set-up with multiple wi-fi radios mounted on a single device as a *wi-fi array*.

To the best of our knowledge, no efforts have been undertaken in related research to characterize achievable data-rates when using wi-fi arrays with all possible orthogonal channels in the 5.2GHz and 2.4GHz spectrum. Hence, we first experimentally determine the achievable data-rates using off-the-shelf (OTS) wi-fi radios. We use Microtik R52 miniPCI cards mounted on Routerboard IA/MP8 8-slot miniPCI-to-PCI adapters for our experimental set-up. Surprisingly, we find that while the expected application layer data-rate with a wi-fi array that uses 15 orthogonal channels (12 'a' and 3 'g') is approximately 600Mbps, *the observed performance is a mere 91Mbps*. We delve into this observation and identify two phenomena, both pertaining to the close physical proximity of the radios on the wi-fi array that together cause the performance degradation. Specifically, we find that *out-of-band (OOB) emission of energy* at a transmitting radio is strong enough at short distances (<1m) that it can trigger carrier sensing at a *nearby* radio operating on an orthogonal channel, and also corrupt the reception of packets at the other radio if it were receiving. Secondly, we find that *filter inefficiencies,* when two radios in close proximity are operating on orthogonal channels, also increases effective bit error rates further lowering performance.

We then present *Glia*[1] a practical software only solution that coarsely coordinates the different radios on a wi-fi array and in the process delivers the aggregate data-rate expected from the array. Glia uses a combination of medium access, scheduling, framing and channel management mechanisms that allow the radios on the wi-fi array to overcome the aforementioned problems. Perhaps, more importantly, we realize Glia as a software module that works with any *off-the-shelf wi-fi radios, thus requiring no changes to the hardware or firmware* of the radios themselves. Using experimental evaluation, we demonstrate that Glia, with a 15 radio wi-fi array (12 'a' radios and 3 'g' radios) achieves approximately 600Mbps[2].

[1]Glia, Greek for 'glue', is a solution that effectively glues together wi-fi radios.

[2]While we don't perform extensive tests of Glia with 802.11n due to current bus speed limits in our experimental set-up, we do show a proof-of-concept that Glia works with 802.11n as well. Thus, a full set-up with Glia and 802.11n in the 2.4GHz and 5.2GHz bands could achieve over 1Gbps in data-rate.

Note that there are several approaches to achieve high datarate wireless communication. Some of these techniques include *channel bonding* [45], *using higher frequency ranges of the spectrum* [35, 43], *wideband techniques* [50], *directional antennas*, *MIMO and adaptive array communication* [45], *radio bonding* [46, 40] and *advanced PHY layer techniques* [7, 17]. However, there are a few fundamental differences, and hence advantages, to the Glia approach to achieving high data rates: (i) Unlike all of the above solutions, Glia is a pure software based solution that works with *off the shelf radios*. We believe that this is a significant advantage when it comes to deployability and time to availability of the solution. (ii) wi-fi is by far the most ubiquitous wireless technology deployed in data networks today, and Glia is built a top wi-fi, and perhaps equally importantly is fully backward compatible with legacy wi-fi devices. (iii) Finally, to the best of our knowledge, despite the promise of high data rate wireless communication that other solutions offer, Glia is the first demonstrated experimental working solution that offers upwards of 600Mbps in data rate. We delve into other specific differences between Glia and the aforementioned alternatives later in the Chapter.

The contributions of this work are three-fold:

- We experimentally study the performance of a 15 radio wi-fi array and characterize the data-rate performance achievable using OTS radios as being a mere 91Mbps. We then identify the reasons behind the lower than expected performance.

- We present Glia, a software only solution effectively exercising a wi-fi array that coarsely coordinates the different radios on a wi-fi array to achieve the expected aggregate performance.

- We prototype Glia and demonstrate in a real experimental set-up that close to 600Mbps data-rate is achieved using only OTS wi-fi radios.

The rest of the chapter is organized as follows: In § 5.2,we describe our setup of wifi-arrays and present the results of default testing of the setup. We also analyze the reasons behind poor performance in the default 802.11 operation. In § 5.3, we explore the core principles of our solution, Glia. In § 5.4, we present the software architecture of Glia and explain how each component of the solution works. In § 5.5, we present the performance evaluation of Glia using an implementation on the wifi-array testbed and also using ns2 based simulations. In § 2.4, we present the related work in this field, and finally we conclude the work in § 5.6.

5.2 *Baseline Performance and Motivation*

5.2.1 Testbed Setup

First, we explain the setup used for experimentation. Two Intel core-2 based Dell Optiplex GX 520 desktops, running Ubuntu Linux (version 8.04, kernel 2.6.24), and equipped with 12 WLAN radios each, act as source and destination wifi-arrays. Since all the arguments we present in the work are relevant only within a single band, we restrict the scope of the experimental set-up to only 12 radios belonging to the 802.11a 5.2GHz band. However, we revisit a complete set-up with 15 radios (12 a and 3 g) in the performance evaluation section. Atheros chipset (AR5413) based Microtik R52 802.11a/b/g miniPCI cards are used as WLAN radios. The cards are mounted on two Routerboard [48] IA/MP8 8-slot miniPCI-to-PCI adapters, each housing 6 cards. The open source Madwifi [47] driver is used for the WLAN cards. The 12 radios together occupy all the 12 available channels in the 5.2GHz spectrum. For the baseline experimentation, the Iperf application is used for generating UDP traffic. The source and destination wifi arrays are placed 10 meters apart. The RTS/CTS of the 802.11MAC protocol is turned off for all experiments. Figure 30 shows a photograph of one of the two wifi-arrays with 12 radios, while Figure 31 shows a schematic of the 12 radio wifi-array testbed.

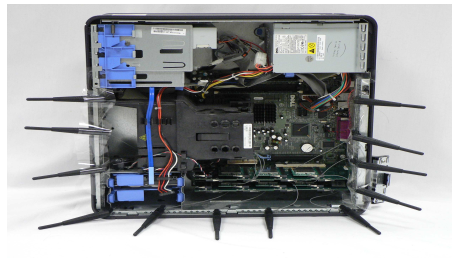


Figure 30: 12 radio wifi-array

5.2.2 Baseline Experimentation

In this Section, we present results of the baseline experimentation using the testbed.First, the individual per-channel data-rate is observed to be around 40Mbps [3], by running only one UDP iperf session across each channel at a time. The 12 channels used by the radios are supposed to be 'orthogonal', i.e, communication on one channel should not affect communication on any of the other channels. Thus the expected aggregate throughput, when all the 12 radios are operated simultaneously, should be around 480Mbps (40* 12). However, when simultaneous UDP iperf sessions are setup on each of the 12 channels, the observed aggregate throughput is only 70Mbps. Figure 33 shows the variation of aggregate throughput as a function of the number of simultaneous links active at the same time. Thus *only 15% of the ideal aggregate throughput capacity* is observed when off-the-shelf radios are used as-is for the wifi-arrays (OTS Wifi).

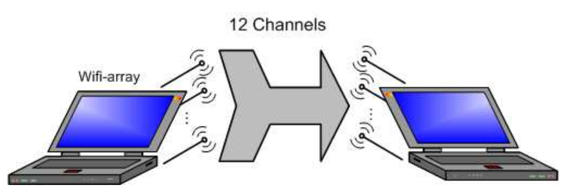
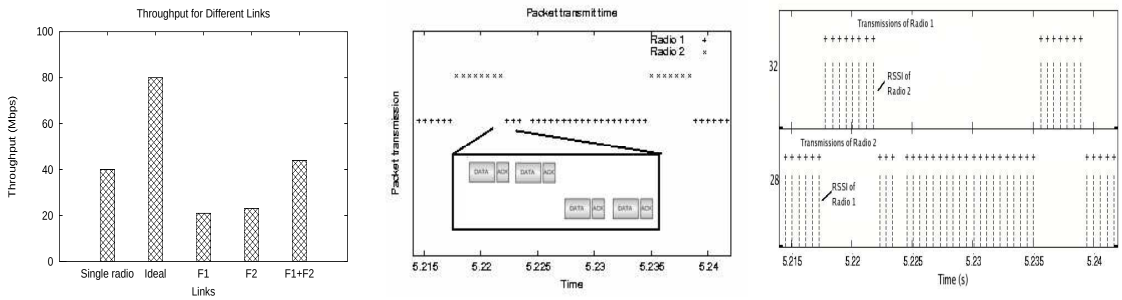


Figure 31: Schematic of 12-radio wifi-array Testbed

[3]Note that the throughput we mention here is the actual application-level achievable throughput from the 802.11 links and not the raw datarates that the 802.11 standard specifies.



(a) Throughputs (b) Wireshark Analysis (c) Packet Error Rates

Figure 32: Experimentation with Collocated Tx/Tx

5.2.3 Analysis

In the previous section we observed that using all 12 channels at the same time using collocated radios gives a lower than expected throughput performance. However, in practice WLAN network deployments do use orthogonal channels simultaneously. The key differentiating property of our experimental set-up when compared to such typical WLAN network deployments is the physical proximity between the radios using the orthogonal channels. To verify that this factor is indeed the reason for the poor performance we use a simple two channel experiment. Two adjacent channels in the 802.11a band (5.18GHz and 5.2GHz) are used for analysis. Figure 34 (a) shows the topology of the experiment, where two links operating on adjacent channels are setup. In this setup the two transmit radios are kept far apart (similarly the two receive radios are also kept far apart). However, the two transmit radios (similarly the two receive radios) are within transmit range of each other. The difference between this setup and a wifi-array setup with two radios is the absence of proximity between the radios. When the two links are active at the same time, the aggregate throughput is observed to be 78Mbps which is close to the ideal aggregate throughput of two channels. This points the reason for poor performance of the wifi-array setup to the proximity of the radios at the transmit and receive nodes.

To understand what exactly happens at each of the transmit and receive wifi-arrays, we experiment with a three node (A, B and C) setup, where node A has two

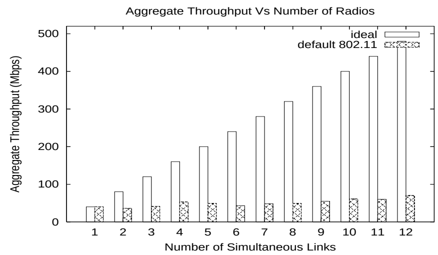


Figure 33: Throughput vs number of radios

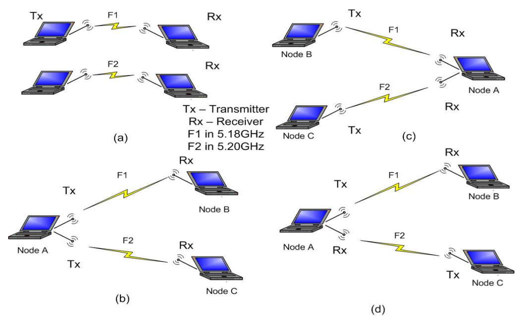


Figure 34: (a) Non proximal radios (b) Collocated Tx/Tx; (c) Collocated Rx/Rx; (d) Collocated Tx/Rx

radios while nodes B and C have only one radio each. Nodes B and C are placed far apart. The two radios at node A connect to either of nodes B or C on adjacent channels (5.18GHz and 5.2GHz). Depending on the direction of DATA flow in each of the two links, there are three possible scenarios, as studied below:

5.2.3.1 Collocated Tx/Tx

In this scenario, both the radios of node A are used for transmission(Tx) of DATA packets (refer Figure 34 (b), while nodes B and C act as receivers. Figure 32 (a) shows the ideal throughput of the two radio setup, and the observed individual and aggregate throughputs. We refer to the two links as F1 and F2. While the expected aggregate throughput is 80Mbps, the observed throughput is only 44Mbps. Thus, single link throughput is what is observed in-spite of the fact that two links on orthogonal channels are active at the same time. A deeper inspection, using the Wireshark packet analyzer shows that in fact only one link is active at any given time. Figure 32(b) is a visualization of the wireshark dump, which shows the times at which packets are sent across the two links. The figure also shows a zoomed version of a part of the visualization. It is clearly seen at any given time only packets belonging to one link are sent. This phenomenon occurs in-spite of the two radios of node A operating on orthogonal channels.

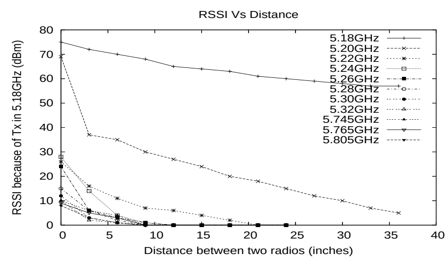
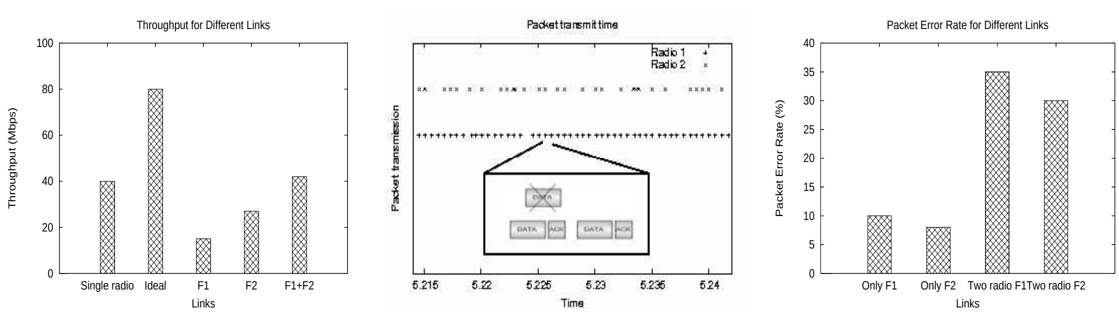


Figure 35: RSSI vs Distance



(a) Throughputs (b) Wireshark Analysis (c) Packet Error Rates

Figure 36: Experimentation with Collocated Rx/Rx

To verify the above phenomenon we investigate the RSSI (Received Signal Strength Indicator) values at both the radios of node A. The RSSI is used by 802.11 radios to perform physical carrier sense and is available readily as a hardware register on the physical device. Figure 32 (c) shows the RSSI at each radio of node A, when the other radio is transmitting DATA packets. It is observed that each radio records a finite RSSI when the other radio is transmitting. This RSSI triggers carrier sensing at either radio and prevents it from transmitting a packet when the other radio is transmitting. Thus even though the two channels are technically orthogonal to each other, there is some power leakage from a transmitting radio on one channel to the other. This leakage power is termed as Out-Of-Band (OOB) emission, and has been discussed in related literature [39].

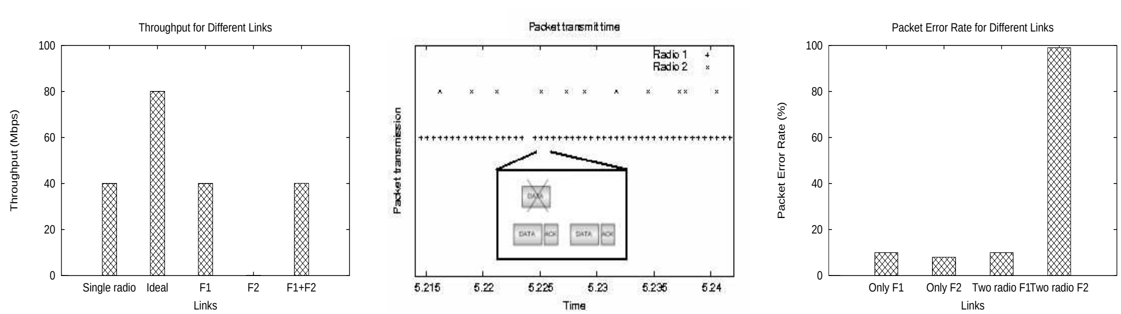
Delving further, we characterize this OOB by studying RSSI values using different channels and different distances between the two collocated radios of node A. Figure 35 shows the variation of RSSI observed on one radio as a function of distance from the second radio, when the second radio is transmitting packets. In the figure we note that when the two radios are placed very close to each other, even channels that are as far as 5.18GHz and 5.805GHz (channels at extreme ends of the 802.11a spectrum) can be affected because of OOB emissions. *This power leakage can however be anticipated and the physical carrier sense mechanism can be suitably modified to account for the OOB.*

5.2.3.2 Collocated Rx/Rx

In this scenario, both the radios of node A are used for receiving(Rx) DATA packets (refer Figure 34 (c)). As in the previous scenario, Figure 36 (a) shows the ideal and observed throughputs of the setup. The observed aggregate throughput of the two links is 45Mbps. Again single link performance is what is observed. To investigate further, we perform Wireshark analysis of the two links. Figure 36 (b) shows a visualization of the times at which packets are sent on each link. While in the previous scenario, it was observed that only one link was active at any given time, in this scenario, packets are sometimes sent on either link at the same time. However, the aggregate throughput is low. To dig deeper, we zoom into the visualization and observe that some packets on either links do not start exactly at the same time, but overlap each other. In this case the reverse direction ACK from one of the radios overlaps with the DATA reception on the other. The ACK for the other DATA packet is not sent back, indicating a packet error. This reverse direction ACK will cause errors on the other DATA packet reception because of OOB emission. Further, we analyze the packet error rates of the received DATA packets [4]. Figure 36 (c) shows the packet error rates on each of the two radios, of node A, while under individual and simultaneous operation. The packet error rates are significantly higher under simultaneous operation confirming the earlier hypothesis that reverse direction ACKs can corrupt DATA reception. This phenomenon of ACKs corrupting DATA occurs irrespective of the channels used by the two radios (as long as the two channels are within the same band), albeit to varying degrees. *Thus it can be concluded that ACKs corrupt DATA.*

Table 4: Packet Error Rates and Aggregate Throughput for Different Locations and Different sets of Adjacent channels used

Channels/Location	PER	Thrpt(Mbps)
5.18, 5.20/ A	0.01, 0.1	75.6
5.18, 5.20/ B	0.32, 0.24	56.7
5.24, 5.26/ A	0.5, 0.1	58.0
5.24, 5.26/ B	0.2, 0.21	62.8



(a) Throughputs (b) Wireshark Analysis (c) Packet Error Rates

Figure 37: Experimentation with Collocated Tx/Rx

From the above observation, turning OFF 802.11 ACKs should result in ideal aggregation of the two links (assuming no background interference). However, a second phenomenon is observed when adjacent channels are used for the two Rx radios. Packet errors are observed in the reception of DATA packets in either radios. The packet error rate, and hence the aggregate throughput is different on the two radios, and varies depending on the location of node A. Even small differences in location

[4]The packet error rates can be figured from a hardware register on the physical WLAN device

can lead to a widely varying observed throughputs. The aggregate throughput is also affected by the adjacent channels being used by the two radios for the same location. However, the throughput remains fairly constant for a considerable amount of time (in the order of a few hours). Table 4 shows variation of aggregate throughput of the two radios and packet error rates with location and adjacent channels used. 802.11 ACKs are disabled for these experiments. The two different locations studied (1 and 2) are only 3 inches apart. As explained in [41] this phenomenon occurs because of the imperfect filter operations at the receive radios. The power from a transmission on a neighboring channel can be filtered along with the legitimate power on the current channel at a receiving radio. This extra power acts as interference and causes CRC errors resulting in packets being dropped at the receive radio. The effect of the extra power is observed only when the channel gains for the adjacent channel is high enough. The channel gains for the receive power can vary depending on location, time and channels being used.

5.2.3.3 Collocated Tx/Rx

In this scenario (refer Figure 34 (d)), one radio of node A transmits packets (link F1) while the other radio receives packet (link F2). The throughput results in Figure 37 (a) indicate that while F1 gets a high throughput of 38Mbps, F2 gets only 1 Mbps. Wireshark analysis shows that while DATA packets are present in both the links, very few packets of F2 are ACKed. The reason for this phenomenon can also be attributed to the OOB emissions from the transmit radio of node A, which make it almost impossible for the other radio to decode its received DATA packets. Figure 37 (c) shows the unusually high packet error rates for F2, when both the radios are operating simultaneously. Thus, it can be concluded that it is not possible to simultaneously transmit and receive using collocated radios. *Since DATA transmission on one radio corrupts DATA reception on a collocated radio, simultaneous DATA transmission and* *reception through collocated radios is never possible.*

5.3 Design Elements in Glia

In this section we present two broad design elements that allow aggregation of throughput capacities of multiple orthogonal channels. These design elements are based on the insights derived from the previous section. In § 5.4, we propose a software-only solution, known as Glia, using these two principles. The first principle, act-as-one, coarsely bonds individual radios and creates a single logical radio, that can use all the available channels at the same time. The second principle, exploit-the-many, allows the right radio-channel association for both the transmitting and receiving wifi-array, such that maximum aggregate throughput can be achieved. The two principles are explained in detail below.

5.3.1 Act-as-One

This design element facilitates multiple radios in a wifi-array to act as one single radio occupying all the channels at the same time. The key concept is to use coarse synchronization of the radios and allow near simultaneous transmission of data packets on the collocated radios.

5.3.1.1 Mutually exclusive Rx/Tx

In §s 5.2.3.2 and 5.2.3.3, we identify that transmission of a packet on a radio will render reception of any packet on a collocated radio useless. Hence it is essential to ensure that simultaneous transmissions and receptions of packets never occur in a wifi-array. However, it is possible to either simultaneously transmit from all collocated radios or simultaneously receive on all collocated radios. We now present a scheduler that achieves this behavior. If a single wifi-array interacts with multiple other wifi-arrays at the same time, it becomes difficult to schedule packets to/from those wifi-arrays (on different channels) such that unnecessary triggering of carrier sense and packet corruption is prevented. Hence, the scheduler allows a wifi-array to interact with only one other wifi-array at any given time.

5.3.1.2 Adaptive Carrier Sensing

In § 5.2.3.1, we identify that OOB emissions from one radio can trigger unnecessary carrier sensing at a collocated radio, thus preventing packet transmission on a radio if a collocated radio is already transmitting another packet. It is possible to estimate the effect of OOB power from a collocated radio. This estimate can be used to prevent an unnecessary carrier sensing, if the OOB from a collocated radio is anticipated. The default carrier sensing mechanism, of identifying if the received power is less than a threshold, can be thus replaced with a more intelligent adaptive carrier sense (ACS) mechanism. The new adaptive carrier sense mechanism will remove the estimated effect of a transmission from the received power before determining if the received power is greater than some threshold, to identify a legitimate carrier. If there are multiple collocated radios transmitting at the same time, the aggregate effect of all the radios by summing the estimated powers of each transmission should be used for the adaptive carrier sense.

Received power is measured at a radio using RSSI [5]. For atheros cards, the RSSI is equal to $10log(SNR)$, where SNR is the Signal to Noise ratio, and is usually reported as an integral value in dBm. Thus, it is not possible to determine the accurate power received, given an RSSI reading. Further, if there are two components to some received power value, a higher power component can mask the lower value. For example, if two components of powers are 15dBm and 20dBm, the aggregate of the two is only 20dBm. It is possible that power from a collocated transmission mask the power of a legitimate background carrier and as a result the legitimate background carrier may not be detected by adaptive carrier sensing. Thus, there are two ranges

[5] The reporting of RSSI is vendor specific. This fact poses a limitation, on our solution, of having to use cards from the same vendor. Further work is needed to combine cards from different vendors

of received power of a legitimate background carrier: 1) a region where a legitimate background carrier can definitely be identified, and 2) a region where the legitimate carrier can be masked by collocated transmissions and hence not be detected.

5.3.1.3 Coarse Synchronization

It is not always possible to identify a legitimate background carrier on a channel if collocated radios are transmitting some packet. It is possible to get complete information about a channel only if all collocated radios are idle. We propose a coarse synchronization across all radios in a wifi-array, where all radios in a transmit wifi-array start sending packets at the same time after physical carrier sense of their respective channels. Each radio sends one packet at a time and waits for an acknowledgment (ACK) from the corresponding radio at the receive wifi-array. An epoch is a time period during which a wifi-array sends packets on different radios and waits for ACKs. ACKs are sent by the receiver radios only after all packets in the epoch have been transmitted. This prevents ACKs corrupting receptions. If a particular radio of the transmit wifi-array senses its channel to be busy, it will not send any packet during that epoch. If some of the packets are not received during an epoch, they are retransmitted during the next epoch. The retransmission can happen through a different radio than the one in which the packet was sent around the first time. For providing fairness across all nodes occupying the channels, the transmit wifi-array performs a random backoff, similar to the random backoff in 802.11 MAC. There is however, only a single backoff for all radios. This ensures the coarse synchronization across all the radios.

This simple model of synchronization has three issues: a) It is not possible to perfectly synchronize all radios to send packets at the same time. There are several possible sources of delay along a packet path in the network stack. These delays are compensated as explained in § 5.4.1.2. b) Since there is a single backoff for all channels, it is possible to be unfair across users. If there are multiple users on a particular channel, packets belonging to the different users may collide with each other. An unsuccessful ACK will indicate such a loss of packet. Ideally in such a scenario, the transmitters should backoff for a larger time on that particular channel during the next packet transmission. However, since all radios in a wifi-array have a single backoff, it is not possible to have a larger backoff for a particular radio. In this case, compensation is provided by not sending any packet in some epoch. c) A radio does not send any packet, during an epoch, if the corresponding channel is busy at the start of the epoch. However, it is not always possible to know if the channel becomes free before the epoch duration. This is because collocated radio transmission can mask the channel. This might be unfair to the wifi-array as other users in the channel might get access to the channel for a longer time than the wifi-array. However, this unfairness is allowed for the particular radio of the wifi-array.

5.3.1.4 Framing

While using a wifi-array, channel conditions may vary across different channels being used. Depending on the channel conditions, different rates of data transmission may be required for different radios,to ensure successful reception of the packets. However, different rates imply different transmit times for packets with same length. So when only one packet is sent across a radio in a single epoch, a slower radio will prevent faster radios from transmitting new packets. Thus a slow radio in a wifi-array can pull down the aggregate throughput achievable out of the wifi-array. However, if different radios, with different rates, use different packet sizes, such that the transmit time for any packet is the same, such wastage can be avoided. All packets from higher layer are joined to form a single byte stream. Suitable sized frames are created from this stream and given to individual radios. This variable size framing is also used to compensate the delays in packet transmission across radios. Given the link layer focus of Glia, we haven't focused on how different higher layer protocols will behave as a result of variable sized frames. A detailed analysis of how Glia interacts with higher layers will be part of our future work.

Table 5: RSSI and Aggregate Throughput for Different Combinations of Radio-channel Association for a 2 radio wifi-array

Combination #	Receive RSSI	Throughput(Mbps)
1	34, 36	70.1
2	31, 40	65.2
3	41, 38	78.2
4	33, 31	60.1

5.3.2 Exploit-the-Many

This design element exploits the presence of diversity of radios at source and destination wifi-arrays to maximize the achievable aggregate throughput. In § 5.2.3.2, we identified that imperfect filtering at the receiver radios leads to packet errors during reception of packets, when adjacent (yet orthogonal) channels are used simultaneously. This error rate depends on location of the radio antenna and even a small difference in location can lead to a huge improvement in aggregate throughput. However, the error rate does not change drastically during short intervals of time. The error rate observed has some correlation with the RSSI observed at a particular receiver radio, when both the channels are simultaneously being used. The higher the RSSI at the receiver, the higher the throughput of the radio. If there are n radios at each of the source and destination wifi-arrays, there are $n! * n!$ combinations to assign n channels to the different radios. It is possible to find a combination that gives the best aggregate throughput. For example, consider a 2 radio wifi-array. Table 5 shows the RSSI readings for the two receiver radios, and their corresponding aggregate throughputs (no carrier sense and no ACK), for the 4 different combinations of radio-channel associations, when both the transmit radios are simultaneously transmitting packets. Combination 3 performs the best, in terms of the aggregate throughput. Combination 3 also has the highest aggregate RSSI. Hence, the sum of RSSI of all receive radios, when all the transmit radios are active, is used as metric to determine the best combination.

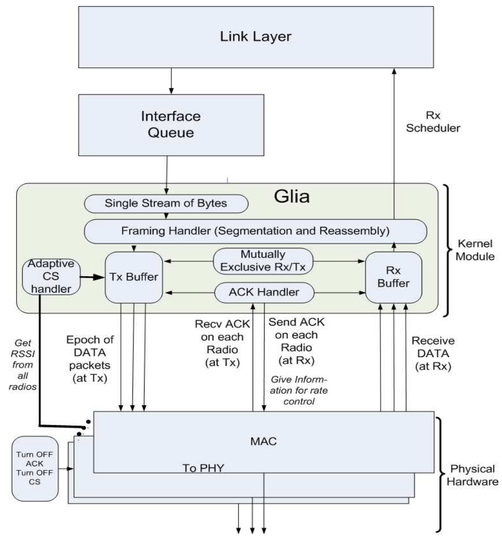


Figure 38: Software Architecture of Glia

5.4 Software Architecture

In this section, we present the details of how each principle, identified in § 5.3 can be implemented in a real system. We develop Glia as a software module that works with any off-the-shelf wi-fi radio, thus requiring no changes to the hardware or firmware of the radios themselves.

5.4.1 2.5 Layer Operation

We propose Glia as a 2.5 layer solution between the link layer and the medium access layer. Figure 38 shows the software architecture of Glia in the network stack. The correct operation of the solution requires the following from the 802.11 MAC: 1) The default carrier sense mechanism has to be turned OFF. Glia relies on adaptive carrier sense mechanism. Real-time RSSI values from the hardware are needed by the Glia layer. 2) The default 802.11 ACK mechanism has to be turned OFF. As discussed earlier, the default ACK is replaced with a delayed ACK, to compensate for the indeterministic delays in the network stack. We now present the various components of Glia in detail.

```
INPUT:
isIdle = Variable indicating if all radios are idle
recvAddr = Address of the receiver
snoopAddr[i] = Src and Dst addresses of opportunistic snooped
   packets, i = 1 to k, k = Total number of addresses
OUTPUT:
isSend = Variable requesting to send packets to recvAddr
ALGORITHM
If (isIdle == 1) {
   if (recvAddr != snoopAddr[i] ∀ i = 1 to k)
     isSend = 1;
   else isSend = 0; }
```

Figure 39: Pseudo Code for Mutually Exclusive Rx/Tx

5.4.1.1 Mutually Exclusive Rx/Tx

The mutually exclusive Rx/Tx scheduler is required to prevent simultaneous transmissions and receptions. The pseudo code of this component is shown in Figure 39. There are two main functionalities of the scheduler: a) The first functionality prevents transmission of DATA packets on any radio if some of the radios of the wifi-array are receiving packets. b) The second functionality prevents transmission of packets to a wifi-array that is already in conversation with a third wifi-array. To achieve this, all wifi-arrays opportunistically snoop on packets that they hear. These packets need not be destined to a snooping wifi-array. However, the addresses on the snooped packets help the scheduler in determining if the intended destination is busy with some other communication, in which case packets should not be sent to the receiver. If it is not possible to snoop packets of an intended receiver (this is possible if the local node is out of reception range of the transmission but within the carrier sense range of the transmission), the wifi-array will depend on adaptive carrier sense to determine if a particular channel is free. However, if the receiver wifi-array is busy with some other interaction, it will not send any ACKs.

```
DEFINITION: epoch = a period of time when radios in a
    wifi-array send out packets.
INPUT: RSSI[i] = Current RSSI of radio i, i = 1 to n
CSthresh = RSSI threshold for default Carrier Sense
aCSthresh[i] = RSSI threshold, for radio i, for ACS,
    using estimated RSSI of collocated transmissions
OUTPUT: oPkt[i] = Packet of suitable size to send on radio i
isSendPkt[i] = 1 if oPkt[i] should be sent in this epoch,
    0 otherwise
VARIABLES: isFree[i] = 1 if channel i is free
ALGORITHM:
for (i = 1 to n)
  if (RSSI[i] < CSthresh) isFree[i] = 1
for (i = 1 to n) {
  if (RSSI[i] < aCSthresh[i]) {
    create oPkt[i] of suitable size
    send oPkt[i] on radio i } }
```

Figure 40: Pseudo Code for Coarse Synchronization

5.4.1.2 *Coarse Synchronization*

The coarse synchronization mechanism is used to send packets through all radios of a wifi-array, within an epoch. The pseudo code for this component is shown in Figure 40. A single backoff is used for all radios. The traditional carrier sense (CS) mechanism is replaced with the adaptive carrier sense (ACS) mechanism. RSSI values are estimated for all combinations of active collocated radios. These estimated RSSI values are used with the current RSSI to perform the adaptive carrier sense as explained in 5.3.1.2. Before sending out any packet in a epoch, ACS is performed on all radios to figure out, if their channels are free. All radios with free channels will send out packets in the current epoch. ACKs are sent by the receiving wifi-array on each radio to indicate the successful reception of the packet. The ACKs are sent using basic rate (defined in 802.11 PHY) to improve reliability. The ACKs are sent after the last packet in the epoch is received. Lost packets are retransmitted in the next epoch, possibly through a different radio (than the first time). ACKs are handled by an ACK handler as shown in Figure 38

Since perfect synchronization of all radios is not possible, the delays that occur as a result of various bottlenecks along the network stack have to be compensated. Since Glia is a 2.5 layer solution, only the delays that are caused below the link layer have to be addressed. The delays can be split into two parts:1) a constant deterministic delay (α) and 2) a variable delay (β), that is not fully deterministic. The deterministic delay occurs because of the system bus bottleneck. This delay can be as high as 11μs per packet if a PCI bus is used for the mounting the radios [6]. An X4 PCIexpress bus can reduce this delay to around 2μs. The α delay is compensated by variable packet size. The first packet is sent out with default packet size. Each successive radio is given a packet that is smaller than the previous one, such that the difference in packet size accounts for the deterministic delay. The goal of the compensation is to have the end times of all packets to be coarsely synchronized. This prevents the reverse direction ACK, on some radio, from corrupting DATA reception on a different radio. The β delay occurs because of system inefficiencies. A range of the β delay is precomputed and this delay is compensated by having an ACK timeout of maximum β after the last packet is sent out. Also before handing out the packet to the radio, a second ACS is performed to determine if no new packet has started transmission during the time between the two ACSs. If the second ACS indicates the presence of a some new background carrier on a particular channel, the channel is not used for this epoch. It is possible that the second ACS does not catch a legitimate carrier, because of masking. In such a scenario, collision will occur at the receiver, and an ACK will not be generated.

Individual radios are allowed to have their own rate control algorithm. However, since the default 802.11 ACK is turned OFF, the driver, which performs the rate control, does not have access to the successful packet delivery information. Instead, the ACK handler sends this information to the driver. After every packet transmission

[6] Assuming a packet size of 1500bytes and PCI bus speed of 133MBps

on a radio, the corresponding driver is given the information whether the transmission was successful or not. This information will be used by the driver to perform rate control.

```
INPUT: rate[i] = Datarate for radio i, i = 1 to n
iPkt[id] = Higher layer packets, id = packet number
OUTPUT: oPkt[i] = Glia Packet for radio i
GLIA PACKET FORMAT:
|Header|Segment|Header|Segment|...|Header|Segment|
Segment = segment of bytes of some iPkt[j]
Where, Header = (pnum, length, offset, more)
pnum = j, input packet number
length = length of iPkt[j] bytes used in this segment
offset = Location of the first byte of Segment in the iPkt[j]
more = 0 if this segment contains the last byte of iPkt[j],
  1 otherwise
VARIABLES USED: pSize[i] = size of Glia packet on radio i
tTime[i] = transmission time for Glia packet on radio i
ALGORITHM:
1  Convert all iPkt[id] into one single byte stream
2  find k such that rate[k] is maximum ∀ i
3  pSize[k] = MTU
4  Choose pSize[i] ∀ i != k such that tTime[i] = tTime[k]
5  Take (pSize[i] - Header size) bytes from byte stream,
    add Headers and create Glia packet
```

Figure 41: Pseudo Code for Framing

5.4.1.3 Framing

This component is used to send packets of different sizes through different radios within an epoch. Variable sizes may be required for accommodating multiple rates or for compensating the α delay. MTU is used for the fastest radio (to accommodate different rates) or the first packet sent out (to compensate the α delay). For accommodating variable rates, packet sizes are determined by ensuring constant packet transmit times. For compensating the α delay, successive packets are given increasingly smaller sizes. Figure 41 shows the pseudo code of the framing component for accommodating different rates. All packets from the higher layer are first combined to form a single byte stream. The packet size is determined for each radio and the corresponding number of bytes are given to the respective radio. The newly formed packets are termed as Glia packets. In order to aid in the re-assembly of the higher layer packets, from the Glia packet, a four tuple header is used for each segment of a unique higher layer packet. The packet format and the descriptions of the four tuple are shown in Figure 41. If a packet has to be retransmitted (because of packet loss), a new packet size may be required. In such a situation, the Glia packet may be further fragmented to make smaller Glia packets, or new segments may be added to make a larger Glia packet.

5.4.1.4 Radio-Channel Association

Radio-channel association involves the exploitation of diversity, possible because of the presence of a potentially large number of combinations ($n! * n!$ for an n radio wifi-array) channel assignments to the source and destination wifi-arrays. As discussed in § 5.3.2, the RSSI measurements at receive radio can be used to estimate the best possible combination. Even though the search space is very large, a significantly smaller number of experiments are sufficient to make the RSSI measurements. The fact that simultaneous transmission using only adjacent channels affect the achievable throughput on any channel, is used to reduce the number of experiments required to make the RSSI measurements. At the transmit wifi-array, three radios are simultaneously activated (we refer to simultaneous activation as sending DATA packets on all three radios at the same time after turning OFF CS and ACKs) using adjacent channels. The RSSI measurement is made for the middle channel on each of the n receive radios. This single experiment will give n RSSI readings for a particular combination of channel (the middle channel), transmit radio (radio at transmit wifi-array using the middle channel), and the receive radio. Changing the three channels of activation and the transmit radio for the central channel lead to a total of n^2 experiments. From these experiments it is possible to determine all the required RSSI values to compute the metric used to determine the best combination. The metric we use is the sum of

RSSI readings, and this simple metric is found to provide a good combination that shows a high achievable aggregate throughput. We use a simple brute force search algorithm. A sophisticated algorithm will be part of our future work. The entire set of experiments can be automated. Once a suitable radio-channel association is selected, it can be used as long as the RSSI values at the receiver do not change significantly. The RSSI values can change if channel conditions have changed, because of mobility or time of operation.

5.4.2 Case Studies

While Glia is primarily designed for multi-radio wifi-arrays, it allows other background 802.11 traffic to co-exist. Further, Glia also allows wifi-arrays to communicate with legacy 802.11 devices. We consider four case studies, depending on the type of nodes present in the network and explain how Glia works in each scenario.

5.4.2.1 Single wifi-array link:

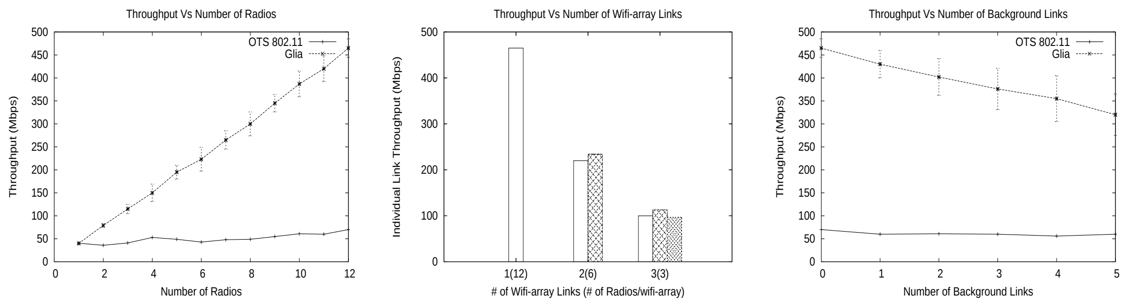
In this scenario a wifi-array A wants to talk to another wifi-array B. There are no other interfering sources. At node A, Glia gets packets from the higher layer, puts them all in a single byte stream, creates packets of variable sizes for different radios and hands over packets to the corresponding radios. Since there is no other transmission in the vicinity, every radio will sense the channel to be free. Each radio will send the Glia packets during the epoch. At the end of the epoch, The receiver node sends ACKs on each radio, if the corresponding packet was received successfully. If some packets are lost during the epoch, they are re-transmitted during the next epoch. Re-transmission might take place on a new radio.

5.4.2.2 Contending wifi-array links:

In this scenario several wifi-arrays contend with each other to transmit packets. Since Glia uses a single backoff for all radios in the wifi-array, and because all the radios are virtually glued together, there should ideally be a single virtual channel with multiple contending nodes (as in a single channel 802.11 network). However, since there is only a coarse synchronization across radios, and there is a finite delay between the start of packets on each radio, different wifi-arrays might take over control of different channels during an epoch. This will result in an epoch, where each of the transmit wifi-arrays use a subset of all the available channels. If the destination nodes of each of the transmit wifi-arrays is different, then Glia will essentially result in a situation with multiple links operating at the same time, with each link operating on a subset of the channels. However, consider a scenario where two wifi-arrays A and B want to talk to the same destination wifi-array C. Since the wifi-arrays A and B have different random backoffs they will likely start at a different time instants and hence only one of them takes over all the channels. On the other hand it is also possible that before all radios of the node that starts first start its transmission, the other node might start its own transmission. In this case, there are two possible situations. If the second node opportunistically snoops the packets of the first node, the mutually exclusive Rx/Tx scheduler will not allow the second node to talk to C. However, if opportunistic snooping is not possible, both nodes will go ahead and send packets on different channels. Node C will only ACK packets belonging to the first wifi-array and ignore all packets from the second wifi-array.

5.4.2.3 Contending background legacy 802.11 links:

In this scenario, there are background 802.11 transmissions on some of the channels that are being used by a pair of wifi-arrays. Because of the random backoff, the channels with the background traffic will be shared between the corresponding radios of the wifi-arrays and the background 802.11 traffic. As discussed in the § 5.4.1.2, the radios of the transmit wifi-array will not use a channel if it is already being used by some other traffic. However, it is possible that OOB emissions mask the background carrier, and ACS fails. In such a situation, packets will collide, on the particular channel, at the receiver radios. This will result in a lost packet. The lost packets will be retransmitted at a later time.



(a) Effect of # of radios (b) Multiple wifi-arrays (c) Effect of background

Figure 42: Evaluation Results Part 1

5.4.2.4 Contending foreground legacy 802.11 links:

In this scenario, a wifi-array A has to interact with both another wifi-array B and a single-radio node C. The mutually exclusive Rx/Tx scheduler will ensure that only one of links (A with B or A with C) is active at any given time. There are four possible scenarios depending on the direction of communication between the A-B and A-C pairs. If the wifi-array A is transmitting to both B and C, then A will either transmit an epoch of packets to B or transmit a single packet to C. Now consider the scenario when A wants to send packets to B and A has to receive from C. In this case, when C is sending some packet to A, the scheduler will ensure that no packet is transmitted from A. Similarly when A is transmitting packets, C will simply backoff because it sees a packet in its channel. The third scenario is when B sends to A and A sends to C. When B sends an epoch of packets to A, the scheduler will not let any packet from A to C. Similarly when A is sending a packet to C, B will opportunistically hear the packet and refrain from sending the epoch to A. The fourth scenario is when both B and C try to send packets to the wifi array A. In this case, the wifi-array B might not be able to opportunistically snoop C's packets, if they are out of transmit range of each other. C and B might be able to detect the other with carrier sense (physical or adaptive) else packets will collide on the channels and will be simply re-transmitted.

5.5 Performance Evaluation

In this section, we present the performance evaluation of Glia on the 12 radio wifi-array testbed. We implement it as a software application, which hooks with the open source madwifi driver. The source-code of madwifi is modified to accept user-input values to any hardware register of the Atheros chipset (for each of the 12 radios), through the iwpriv command. The current RSSI of the chipset is mapped to a /proc file that can be accessed in real-time. The default CS of the chipset is turned OFF using the transmit-stomping feature. Traffic stomping works by telling the card to interrupt any reception of any data packet and shift to transmit mode when there are packets to send. The 802.11 ACK is turned OFF by setting the noACK parameter of the 802.11e QoS specification. All the other elements of Glia are implemented as user space C code. Traffic is generated using UDP datagrams. Unnecessary processes in the Linux OS (Example: X server) are turned OFF to reduce the indeterministic delays. ACS for a radio is performed using current RSSI value and pre-estimated RSSI values for OOB emissions. Although we don't implement the framing mechanism, we study the impact of variable frame sizes for different rates (§ 5.5.6). Radio-channel association is performed as an offline process by first performing the individual experiments, as described in § 5.4.1.4. The channel associations are computed offline and fed manually to the individual radios at source and destination wifi-arrays. All experiments are performed in an urban office environment. There are no background users on any of the 5.2GHz channels. However, there are users in the 2.4GHz spectrum. Unless otherwise specified, the results are provided for experiments using the 5.2GHz band. Unless otherwise mentioned, all results are obtained as a result of 10 experimental runs.

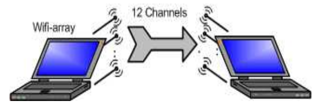


Figure 43: Single wifi-array Link

5.5.1 Single wifi-array Link

We first study the effect of number of radios on the throughput capacity of a single wifi-array link in an isolated environment (Figure 43). Each radio operates on a different 'a' channel. Figure 42(a) compares performance Glia with off-the-shelf (OTS) 802.11 operation. The OTS performance suffers for reasons identified in § 5.2. However, Glia shows expected linear behavior of throughput, indicating the fact that all the channels are effectively being used. With all 12 radios, Glia is able to provide a throughput of about 465 Mbps very close to the ideal 480Mbps.

5.5.2 Multiple Contending wifi-array links

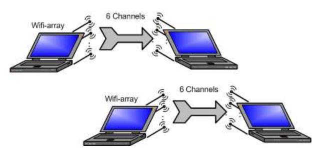


Figure 44: Multiple Wifi-Array Links

Here, we show how Glia operates in the presence of multiple wifi-array links (Figure 44). Due to the lack of enough equipment, we use lesser number of radios for each wifi-array, when experimenting with multiple wifi-array links. We use independent source destination pairs for each link. Figure 42(b) shows the individual link throughputs for different number of links. It can be observed that, in each scenario, all the wifi-array links get similar throughputs. In fact the links share the available bandwidth of all the channels they operate on. The single backoff across all the radios of a wifi-array ensures that the wifi-array acts as a single logical radio. It is however possible that different links use different sets of channels at the same time. However, on the average, each link gets approximately the same throughput.

5.5.3 Contending background 802.11 links

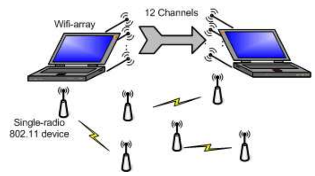


Figure 45: Glia Link with Background Traffic

Next, we study the fairness properties of Glia when there is legitimate background traffic on some of the channels (Figure 45. Multiple background single-radio 802.11 links are added to different channels used by a 12 radio wifi-array link. While Figure 42(c) shows how the number of background links affects the 12 radio throughput, Figure 46 shows the aggregate throughput of the background links. Results of Glia are compared with a OTS 802.11 wifi-array. The throughput of the wifi-array is much higher for Glia, as expected. While a Glia wifi-array tries to share any channel with a background link present on the channel, an OTS 802.11 wifi-array uses very little of any channel. Hence, in the case of OTS 802.11, background links get more time to transmit and as a result experience higher throughput.

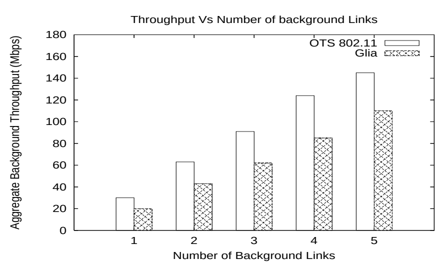


Figure 46: Aggregate of background Traffic

5.5.4 Contending foreground 802.11 links

Now, we study how Glia can coexist with other legitimate foreground 802.11 traffic. (Figure 47). Multiple single radio clients, each on a different channel, are added to the setup of a wifi-array link. A single wifi-array acts as the source for all the single

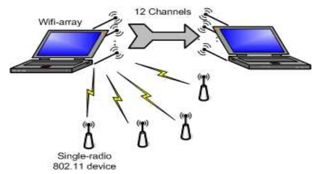
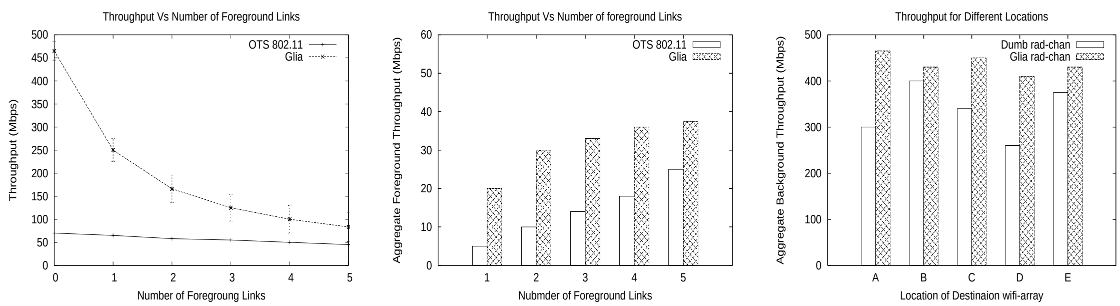


Figure 47: Glia Link with Foreground Traffic

radio clients as well as the other wifi-array. Due to lack of space, we do not study other situations of traffic directions. While Figure 48(b) shows how the number of foreground links affects the 12 radio throughput, Figure 48(a) shows the aggregate throughput of the foreground links. It can be seen that the throughput of the wifi-array link falls drastically with addition of new single-radio links. This is because the transmit wifi-array can send traffic to only one other destination at a given time. When the transmit wifi-array is talking to a single radio node, only one of the multiple radios is active.



(a) Effect of foreground (b) Aggregate of foreground (c) Effect of Location

Figure 48: Evaluation Results Part 2

5.5.5 Radio-Channel Association

As discussed in § 5.3.2, the radio-channel association plays an important role in achieving the best throughput out of a wifi-array. Further, the radio-channel association depends on the physical location of the source and destination wifi-arrays. In this experiment, using a single 12 radio wifi-array link, the location of the source wifi-array is fixed and the location of the destination is changed within the transmit range of the source. These different locations are all within the urban office environment.

Glia's radio-channel association is compared with a dumb association in Figure 48(c). The dumb association just assigns channels to the radios in a sequential order. The results indicate that the throughput achieved with Glia is always higher than the dumb association. What is interesting to note is that a dumb association can lead to throughput that is only about 60% of the maximum achievable throughput.

5.5.6 Effect of Different Datarates

In § 5.3.1.4, a framing algorithm is proposed for using different rates at different radios of a Glia link. Instead of actually implementing the variable packet size algorithm, we study the effect of the variable frames size by manually setting the frame size for different rates. We study the effect of framing in a simpler 2 radio wifi-array setup. Table 6 shows aggregate throughput achieved by the two radios for constant frame size for both radios and variable frame sizes. When using a constant frame size, a slower radio will pull down the aggregate throughput as only one packet is sent on a channel during an epoch. However, with variable frame sizes the transmission time for packets in either channels is the same, thus increasing the aggregate throughput.

Table 6: Aggregate Throughput for Different Datarates (Mbps)

Rates	Const Pkt	Var Pkt
54, 6	10	43.1
54, 48	72	75.1
36, 12	18	37.8
12, 6	12	16

5.5.7 Glia in 2.4GHz band

Thus far we provided results of Glia operation in the 5.2GHz band. In this section we provide results of Glia operation in 2.4GHz band. The 2.4GHz band is a relatively congested band with lots of users. We show how Glia can aggregate the limited available bandwidth in this band. There are only three orthogonal channels that can be used in the 2.4GHz band (channels 1,6, and 11). Table 7 compares the aggregate throughput achieved by Glia with a default 802.11 implementation on a three radio setup using all the three channels. We run the experiments at two different times when the background traffic conditions are different. Glia can aggregate the available throughput at any given time.

Table 7: Glia in 2.4GHz Band

Scenario	Aggregate Throughput (Mbps)
802.11 (12:00pm)	20
Glia (12:00pm)	55
802.11 (12:00am)	34
Glia (12:00am)	95

5.5.8 Glia in dual band operation

Since Glia uses independent radios for each channel, we can use it in all the 15 available orthogonal channels in the 2.4GHz and 5.2GHz unlicensed bands at the same time. Further, a transmission in the 2.4GHz band will not cause OOB emissions in the 5.2GHz band and vice versa. Hence we can run Glia on a 15 radio node independently, as two sets of Glia links, one in each band. Figure 49 shows the throughput vs number of channels used in such a 15 radio Glia link. These experiments were carried out at 12:00 am in the night when the 2.4GHz band is relatively free. Glia can show an aggregate throughput of 567Mbps with all the 15 radios.

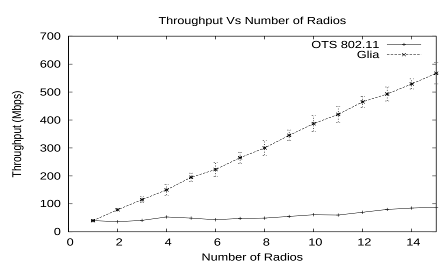


Figure 49: Glia in dual band operation

Table 8: Aggregate Throughput for collocated 802.11n radios (in Mbps)

Scenario/Channels	Aggregate
Ideal Two-Radio	192
Default Two-Radio	81
Partial Glia	132

5.5.9 Glia in 802.11n context

802.11n is latest standard in the 802.11 suite of protocols. It offers higher throughputs among other benefits, by utilizing a variety of technologies like MIMO Multiple Input Multiple Output) antennas, spatial multiplexing and a wider bandwidth (40MHz operation). Although, we present Glia in the context of 802.11 a/g, the design elements of Glia are valid even in the context of 802.11n. To study the impact of Glia in an 802.11n setting, we equip one of the wifi-arrays (Linux desktop) with two miniPCI 802.11n radios based on the Atheros 9160 chipset. The chipsets use the open source ath9k driver. Because the ath9k driver is still in a development stage, we were able to use the cards only in the client mode. Hence, we use two other 802.11n access points (a Linksys WRT600n and a Netgear WNR834) for the other ends of the wifi-array, connected via Gigabit Ethernet to two other Linux machines. The topology used is similar to the Tx/Tx topology of Figure 34(b). We use the 2.4GHz spectrum for experimentation. Since the 2.4GHz band has only about 60MHz of available spectrum, one of the two radios of the wifi-array can use a 40MHz channel and the other radio can use the remaining 20MHz channel. Table 8 shows the ideal and observed throughput when the two radios of the wifi-array transmit DATA packets simultaneously. It is observed that even 802.11n performs poorly in a wifi-array. We were able to disable CS of the 802.11n cards but not able to disable the 802.11 ACKs. Hence we were unable to fully implement Glia in the 802.11n context. However, disabling carrier sense, does show benefits in the aggregate throughput (confirming the OOB emission effect). The reason for not achieving the ideal throughput is that reverse direction ACKs collide with legitimate DATA packets. While this straw-man implementation shows the relevance of Glia in an 802.11n, we believe a full blown implementation can provide even higher aggregate throughputs. Ideally it should be possible to achieve about an aggregate throughput of around 1.2Gbps using all the 15 channels in the two bands.

5.5.10 TCP performance with Glia

Thus far, we have not explicitly considered the use of TCP (Transmission Control Protocol) as the transport layer protocol for traffic that is sent over Glia. However, there are some important implications of using Glia with TCP based traffic. Glia, as we have presented it, does not explicitly provide in-order packet delivery. Since packets are served opportunistically on the different links on an array, it is possible that packets arrive out-of-order at the receiving end depending upon the bandwidths and delays along the different links in an array. However, TCP *interprets* sustained out-of-order packet delivery as a sign of network congestion (it infers a loss on the third DUPACK for a particular sequence number) and will cut down the rate at which a connection is operating. Fortunately, a simple extension to Glia that explicitly does re-sequencing at the receiving end will address this above problem. We defer further investigation of such techniques and an in-depth study of the impact of Glia on other higher layer protocols for future research.

5.6 Conclusions

In this work, we identify the practical issues of aggregating throughput of multiple orthogonal channels using multiple off-the-shelf radios in a wifi-array. We analyze the reasons for poor performance in such wifi-arrays using a combination of wireless packet trace analysis, and spectrum analysis. We present a practical software only solution, known as Glia, that can achieve close to theoretical aggregation. We evaluate our solution with an implementation on a 12 radio wifi-array testbed. A Mobility analysis will be part of our future work.

CHAPTER VI

CONCLUSION AND FUTURE WORK

In this thesis, we addressed the core issues that impede the simultaneous use of multiple orthogonal channel in wireless adhoc networks. We identified open challenges that span from a single multichannel wireless link to the challenges that impact the entire network. In particular, we considered two important network wide problems in multi-channel adhoc networks involving channel assignment and routing. At the link level, we identified and solved the practical challenges associated with using multiple orthogonal channels for providing a high data-rate wireless link across two devices. In the following we summarize the main contributions of this thesis research.

6.1 Main Contributions

- We considered the channel assignment problem in single-radio multi-channel mobile adhoc networks and investigated the *granularity of channel assignment decisions* that gives the best trade-off in terms of performance and complexity. We identified a new granularity for channel assignment that we refer to as component-based channel assignment that is simple and has impressive practical benefits.

- We performed theoretical analysis of the component-based channel assignment strategy and compared it with flow and link-based strategies. We proved that the component-based strategy does not lag significantly behind the optimal performance, and when coupled with its several practical advantages, it significantly outperforms other strategies under most network conditions.

- We evaluated the benefits of the component-based channel assignment using both ns2 simulations and a strawman prototype. We showed how component based strategy performs the best because of the lack of switching and scheduling overheads.

- We considered the routing problem in multi-radio multi-channel adhoc networks. We identified the interface insufficiency bottleneck in such networks that limits the throughput performance. We proposed a 4D routing protocol, known as 'lattice routing' that performs multipath routing to overcome this bottleneck. The protocol also dynamically adjusts paths as a result of changing traffic conditions.

- We performed extensive ns2 simulations to show the benefits of the proposed 'lattice routing' protocol in terms of applicability to various network conditions. We compared the performance with other state-of-art routing protocols for such networks and showed the benefits of 'lattice routing'.

- We then considered the link-level challenges in realizing a multi-radio multi-channel wireless link. In particular, we identify practical challenges involved in aggregating throughput of multiple orthogonal channels using multiple off-the-shelf wi-fi radios. We analyzed the reasons for poor performance in such wifi-arrays using a combination of wireless packet trace analysis, and spectrum analysis.

- We proposed a practical software-only solution, known as 'Glia', for achieving close to theoretical aggregation in a multi-channel wireless link. We propose Glia as 2.5 network layer solution that sits between the network and medium access layers of the protocol stack and hence can be easily used for any off-the-shelf wi-fi radios.

- We fully implemented Glia in a first-of-its kind 15-radio testbed, and showed real throughput aggregation close to the theoretical maximum. We also studied the performance characteristics of Glia and showed how Glia can inter-operate with other legacy wi-fi standards.

6.2 Future Work

There are a number of challenges for using multiple orthogonal channels in wireless networks. We have identified several challenges at the network and link level that are unique to this environment. However, there are several open problems that a specific to multichannel wireless networks. Also there are several open problems that have opened up as a result of this research. We highlight some of the important problems in the following:

- The transport layer of the network stack provides end-to-end reliability, and flow-control for connections in networks. While we have addressed the issues of network and link layer, the multi-channel usage in wireless networks poses a unique problem to the transport layer. The transport layer provides an in-order delivery of packets from the source to the destination. However, when multiple channels are used simultaneously using multiple radios, packets are delivered opportunistically depending on existing conditions of the channels. As a result packets may arrive out of order and this will severely impact TCP's performance, even though multiple channels provide a high data rate wireless link. This is because, TCP *interprets* sustained out-of-order packet delivery as a sign of network congestion (it infers a loss on the third DUPACK for a particular sequence number) and will cut down the rate at which a connection is operating. A new transport layer solution for multi-radio multi-channel wireless network that handles these out-of-order packets is an important challenge.

- In Glia, we identified a greedy algorithm for the radio-channel association for improving the performance of wifi arrays. However, this solution depends on a exhaustive search of all the different possible combinations of radio-channel associations. While a one time association is sufficient for static scenarios, a mobile scenario or a rapidly changing channel environment would demand this association process to be performed very often. Identifying the core performance characteristics of a radio-channel selection would lead to a faster association algorithm.

- 802.11n is an emerging standard for indoor wireless access and can provide higher capacity per channel using advanced physical layer techniques such as the use of smart antennas. As we saw in Section 5, 802.11n also has the same issues with using multiple orthogonal channels simultaneously. Extending the principles developed in Glia for 802.11g/a to 802.11n will be a worthwhile endeavor providing even higher throughputs over a wireless link. Further, the principles developed in Glia can also be extended to other types of wireless networks such as Wimax, LTE, etc.

- In this thesis, we presented Glia as a solution for achieving a high datarate wireless link. Extending the testbed to implement an adhoc wireless network would identify practical implementation issues in using multiple radios in such environments.

- While we used multiple off-the-shelf components for building the multi-radio wifi array, it is possible to absorb the principles developed in this thesis in to a single device solution that incorporates multiple radios. A single device solution will have added benefits in terms of reduced physical size, integration of medium access logic that is currently split across multiple channels etc.

In conclusion, there are a number of practical problems with using multiple orthogonal channels in wireless networks. The wireless spectrum is a scarce resource and it is certainly worthwhile investigating these issues to further our understanding so that real devices can enjoy high datarates in such constrained environments.

CHAPTER VII

PUBLICATIONS

Conference Papers

1. **S. Kakumanu** and R. Sivakumar, "Glia: A Practical Solution for Effective High Datarate Wifi-Arrays, " *ACM International Conference on Mobile Computing and Networking (MOBICOM)* , Sep. 2009.

2. R. Vedantham, **S. Kakumanu**, S. Lakshmanan, and R. Sivakumar, "Component Based Channel Assignment in Single Radio, Multi-channel Ad Hoc Networks," *ACM International Conference on Mobile Computing and Networking (MOBICOM)* , Sep. 2006.

3. **S. Kakumanu**, S. Lakshmanan, Y. Jeong and R. Sivakumar, "Enhancing IPTV using Sensor Networks," Demonstration at *ACM International Conference on Mobile Computing and Networking (MOBICOM)* , Sep. 2007.

4. C.-L. Tsao, **S. Kakumanu**, and R. Sivakumar, "SmartVNC: An Effective Remote Computing Solution for Smartphones," *ACM International Conference on Mobile Computing and Networking (MOBICOM)* , Sep. 2011.

5. Y. Jeong, **S. Kakumanu**, C.-L. Tsao, and R. Sivakumar, "Improving VoIP Call Capacity over IEEE 802.11 Networks," *IEEE International Conference on Broadband Communications, Networks, and Systems (BROADNETS)*, Sep. 2007

6. Z. Zhuang, **S. Kakumanu**, Y. Jeong, R. Sivakumar, and A. Velayutham, "On the Impact of Mobile Hosts in Peer-to-Peer Data Networks," *International*

Conference on Distributed Computing Systems (ICDCS), Jun. 2009.

7. Y. Jeong, S. Lakshmanan, **S. Kakumanu**, and R. Sivakumar, "Cue-based Networking using Wireless Sensor Networks: A Video-over-IP Application," *IEEE Communications Society Conference on Sensor, Mesh and Ad hoc Communications and Networks (SECON)*, Jun. 2008.

8. C.-L. Tsao, **S. Kakumanu**, and R. Sivakumar, "A Solution for Rapid Mobilization of Enterprise Applications," submitted to *ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT)* , Dec. 2011.

Journal Papers

1. **S. Kakumanu**, R. Vedantham, S. Lakshmanan and R. Sivakumar, "Component Based Channel Assignment (CBCA): A New Channel Assignment Strategy for Multi-Channel Single-Interface Ad-hoc Networks," *revised version submitted to IEEE Transactions on Mobile Computing*, Dec. 2010.

2. **S. Kakumanu**, S. Eidenbenz and R. Sivakumar, "Lattice Routing: A 4D Routing Scheme for Multiradio Multichannel Adhoc Networks," *Elsevier Adhoc networks journal*, Jan. 2011.

3. Y. Jeong, **S. Kakumanu**, C.-L. Tsao, and R. Sivakumar, "VoIP over Wi-Fi Networks: Performance Analysis and Acceleration Algorithms," *Springer Mobile Networks and Applications Journal (MONET)*, Aug. 2009.

4. Z. Zhuang, **S. Kakumanu**, Y. Jeong, R. Sivakumar, and A. Velayutham, "Mobile Hosts Participating in Peer-to-Peer Data Networks: Challenges and Solutions," to appear in *Wireless Networks*, 2010.

5. Y. Jeong, S. Lakshmanan, **S. Kakumanu**, and R. Sivakumar, "Cue-Based Networking," *ACM/Springer Wireless Networks Journal (WINET)*, Apr. 2011

REFERENCES

[1] Adya, A., Bahl, P., Padhye, J., Wolman, A., and Zhou, L., "A multi-radio unification protocol for IEEE 802.11 wireless networks," in *IEEE International Conference on Broadband Networks (BROADNETS)*, Sept. 2004.

[2] Alicherry, M., Bhatia, R., and Li, L. E., "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Sept. 2005.

[3] Bahl, P., Chandra, R., and Dunagan, J., "SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks," in *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Sept. 2004.

[4] Cheng, C.-M., Hsiao, P.-H., Kung, H. T., and Vlah, D., "Adjacent channel interference in dual-radio 802.11a nodes and its impact on multi-hop networking," in *IEEE Global Communications Conference (GLOBECOM)*, Dec. 2006.

[5] Draves, R., Padhye, J., and Zill, B., "Routing in multi-radio, multi-hop wireless mesh networks," in *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Sept. 2004.

[6] Gupta, P. and P.R.Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, pp. 388–404, Mar. 2000.

[7] Halperin, D., Anderson, T., and Wetherall, D., "Taking the sting out of carrier sense: interference cancellation for wireless lans," in *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Sept. 2008.

[8] Kakumanu, S., Vedantham, R., Lakshmanan, S., and Sivakumar, R., "Component based channel assignment (cbca): A new channel assignment strategy for multi-channel single-interface ad-hoc networks," *accepted with major revisions in IEEE Transactions on Mobile Computing.*

[9] Kar, K., Luo, X., and Sarkar, S., "Throughput-optimal scheduling in multichannel access point networks under infrequent channel measurements," in *IEEE International Conference on Computer Communications (INFOCOM)*, May 2007.

[10] Kodialam, M. and Nandagopal, T., "Characterizing achievable rates in multi-hop wireless mesh networks with orthogonal channels," *IEEE/ACM Transcations on Networking*, vol. 13, pp. 868–880, Aug. 2005.

[11] Kodialam, M. and Nandagopal, T., "Characterizing the capacity region in multi-radio and multi-channel mesh networks," in *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Sept. 2005.

[12] Kyasanur, P. and Vaidya, N., "Capacity of multi-channel wireless networks: Impact of number of channels and interfaces," in *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Sept. 2005.

[13] Kyasanur, P. and Vaidya, N., "A routing protocol for utilizing multiple channels in multi-hop wireless networks with a single transceiver," in *International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QSHINE)*, Aug. 2005.

[14] Kyasanur, P. and Vaidya, N., "Routing and link-layer protocols for multi-channel multi-interface ad hoc wireless networks," *Mobile Computing and Communications Review*, vol. 10, pp. 31–43, Jan. 2006.

[15] Lee, S.-J. and Gerla, M., "Smr: Split multipath routing and maximally disjoint paths in ad hoc networks," in *IEEE International Conference on Communications (ICC)*, June 2001.

[16] Liese, S., Wu, D., and Mohapatra, P., "Experimental characterization of an 802.11b wireless mesh network," in *IEEE International Wireless Communications and Mobile Computing Conference (WCMC)*, July 2006.

[17] Lin, K. C.-J., Kushman, N., and Katabi, D., "Ziptx: Harnessing partial packets in 802.11 networks," in *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Sept. 2008.

[18] Lin, X. and Rasool, S., "Distributed joint channel-assignment, scheduling and routing algorithm for multi-channel ad hoc wireless networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, May 2007.

[19] Liu, Y., Xiong, Y., Yang, Y., Xu, P., and Zhang, Q., "An experimental study on multi-channel multi-radio multi-hop wireless networks," in *IEEE Global Communications Conference (GLOBECOM)*, Dec. 2005.

[20] Marina, M. K. and Das, S. R., "On-demand multipath distance vector routing for ad hoc networks," in *IEEE International Conference on Network Protocols (ICNP)*, Nov. 2001.

[21] Nassr, M., Jun, J., Eidenbenz, S., Hansson, A., and Mielke, A., "Scalable and reliable sensor network routing: Performance study from field deployment," in *IEEE International Conference on Computer Communications (INFOCOM)*, May 2007.

[22] Nsipuri, A. and Das, S. R., "On-demand multipath routing for mobile ad hoc networks," in *IEEE International Conference on Computer Communication Networks (ICCCN)*, Oct. 1999.

[23] Patra, R., Nedevschi, S., Surana, S., Sheth, A., Subramanian, L., and Brewer, E., "WiLDNet: Design and Implementation of High Performance WiFi Based Long Distance Networks," in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, April 2007.

[24] Rad, A. H. M. and Wong, V., "Joint channel allocation, interface assignment and mac design for multi-channel wireless mesh networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, May 2007.

[25] Rahul, H., Kushman, N., Katabi, D., Sodini, C., and Edalat, F., "Learning to share: narrowband-friendly wideband networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, 2008.

[26] Raman, B. and Chebrolu, K., "Design and evaluation of a new mac protocol for long-distance 802.11 mesh networks," in *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Sept. 2005.

[27] Raniwala, A. and Chiueh, T., "Architecture and algorithms for an ieee 802.11-based multi-channel wireless mesh network," in *IEEE International Conference on Computer Communications (INFOCOM)*, Mar. 2005.

[28] Robinson, J., Papagiannaki, K., Diot, C., Guo, X., and Krishnamurthy, L., "Experimenting with a multi-radio mesh networking testbed," in *IEEE International Workshop on Wireless Network Measurements (WiNMee)*, Aug. 2005.

[29] Sarkar, S. and Tassiulas, L., "Back pressure based multicast scheduling for fair bandwidth allocationu," in *IEEE International Conference on Computer Communications (INFOCOM)*, Apr. 2001.

[30] Shacham, N. and King, P., "Architectures and performance of multichannel multihop packet radio networks," *IEEE Journal on Selected Area in Communications*, vol. 5, pp. 1013–1025, July 1987.

[31] So, J. and Vaidya, N. H., "Multi-channel MAC for ad hoc networks: Handling multi-channel hidden terminals using a single transceiver," in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, May 2004.

[32] Sundaresan, K., Hsieh, H.-Y., and Sivakumar, R., "IEEE 802.11 over multi-hop wireless networks: Problems and new perspectives," *Ad Hoc Networks Journal*, vol. 2, Apr. 2004.

[33] Tam, W.-H. and Tseng, Y.-C., "Joint multi-channel link layer and multi-path routing design for wireless mesh networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, May 2007.

[34] Vedantham, R., Kakumanu, S., Lakshmanan, S., and Sivakumar, R., "Component based channel assignment in single radio, multi-channel ad hoc networks," in *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Sept. 2006.

[35] Weiss, M., Huchard, M., Stohr, A., Charbonnier, B., Fedderwitz, S., and Jager, D., "60-ghz photonic millimeter-wave link for short- to medium-range wireless transmission up to 12.5 gb/s," *Lightwave Technology, Journal of*, August 2008.

[36] Wu, S.-L., Lin, C.-Y., Tseng, Y.-C., and Sheu, J.-L., "A new multi-channel MAC protocol with on-demand channel assignment for multi-hop mobile ad hoc networks," in *IEEE International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN)*, Dec. 2000.

[37] Ye, Z., Krishnamurthy, S. V., and Tripathy, S. K., "A framework for reliable routing in mobile ad hoc networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, June 2001.

[38] Zhang, J., Wu, H., Zhang, Q., and Li, B., "Joint routing and scheduling in multiradio multi-channel multi-hop wireless networks," in *IEEE International Conference on Broadband Networks (BROADNETS)*, Oct. 2005.

[39] Zhu, J., Waltho, A., Yang, X., and Guo, X., "Multi-radio coexistence: Challenges and opportunities," in *IEEE International Conference on Computer Communication Networks (ICCCN)*, Aug. 2007.

[40] "Buffalo nfiniti wireless dual band ethernet converter, http://www.buffalotech.com/home/," June 2009.

[41] "The effects of adjacent channel rejection and adjacent channel interference on 802.11 wlan performance, available online at http://whitepapers.techrepublic.com.com/abstract.aspx?- docid=70499," June 2011.

[42] "Enabling new ultra portable form factors through multi-radio integration, avalable online at download.intel.com/pressroom/kits/events/idfspr_2008/ day0_krishnamurthysoumyanath_en.pdf," June 2011.

[43] "Gigabit wireless, available online at http://nicta.com.au/research/projects/wireless_gige," June 2011.

[44] "Ieee 802.11 vht study group, available online at http://www.ieee802.org/11/reports/vht_update.htm," June 2011.

[45] "Ieee 802.11n standard, http://en.wikipedia.org/wiki/802.11n," June 2011.

[46] "Linksys dual band ap, http://www.linksysbycisco.com/us/en/products/wrt400n," June 2011.

[47] "Madwifi, available online at http://madwifi-project.org/wiki," June 2011.

[48] "Routerboard, available online at www.routerboard.com/," June 2011.

[49] "Ruckus wireless, http://www.ruckuswireless.com/," June 2011.

[50] "Ultra-wideband, http://en.wikipedia.org/wiki/ultra-wideband," June 2011.

[51] "Xirrus 16 radio wifi-array, http://www.xirrus.com/products/arrays-80211abg+n.php," June 2011.

VITA

Sandeep Kakumanu was born in Guntur, India and brought up in Chennai, India. Since fall 2004, he has been a PhD student in the school of Electrical and Computer Engineering at Georgia Institute of Technology. His research interests are in the areas of wireless sensor and actor networks, wireless mesh networks, and network protocols for wireless ad-hoc networks.

Prior to his doctoral studies, he received his M.S. in Electrical and Computer Engineering from Georgia Institute of Technology in 2007, and B.Tech. in Electrical Engineering from Indian Institute of Technology, Madras in 2004. He has publications in Elsevier Ad Hoc Networks, Springer Wireless Networks, Springer Monet, ACM MOBICOM, IEEE SECON, IEEE BROADNETS, and IEEE ICDCS among other leading journals and conferences. He has reviewed several papers in leading conferences and journals (MOBICOM 2005-2011, INFOCOM 2005-2010, MOBIHOC 2005-2009, ToMC 2006-2010, Computer Networks 2006-2011). He also has filed two US patent applications and two provisional patents. His industrial experiences include summer internships at the Los Alamos National Laboratory and Asankya Inc.