INTEGRATED ROUTING SYSTEM FOR WIRELESS MESH NETWORKS

By

Amir Esmailpour

**Bachelor of Computer Science with Honors
Carleton University, Ottawa, Canada, 2000**

A thesis
Presented to Ryerson University
In partial fulfillment of the
Requirements for the degree of
Master of Applied Science
In the Program of
Computer Networks

Toronto, Ontario, Canada, 2006

©Amir Esmailpour 2006

PROPERTY OF
RYERSON UNIVERSITY LIBRARY

UMI Number: EC53490

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI Microform EC53490
Copyright 2009 by ProQuest LLC
All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Author's Declaration

I hereby declare that I am the sole author of this thesis.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Acknowledgements

Hereby I would like to express my gratitude to
Dr. Muhammad Jaseemuddin, for his tremendous guidance and assistance throughout this
research. I also would like to thank
Dr. Bobby Ma for his advice and supports in various stages of my work, fellow student,
Osama Bazan, for his time he gave to provide feedback and recommendations on this
thesis. Finally, I want to give my gratitude to my wife,
Sheiba Feizizadeh; who always supports and encourages me.

Abstract
Amir Esmailpour,
Integrated Routing System for Wireless Mesh Networks
M.A.Sc., Computer Networks, Ryerson University, 2006

Recently Wireless Mesh Network (WMN) has become popular especially for its low cost deployment in the areas of poor network infrastructure and terrain of difficult deployment. Although paths in WMN backbone network are stable, the access link contention can severely constrain the end-to-end throughput. We proposed an integrated routing system for WMN that exploits both paths through the backbone network and through the ad-hoc access network of mobile nodes. The ad-hoc path is considered as an alternative path and is used only when the primary backbone path is severely constrained due to access links contention. We have shown through simulation that alternative path is effective in delivering higher throughput in that situation. We also proposed a scheme for initiating the route discovery of the ad-hoc path.

The main contributions of this thesis are two algorithms that allow the mobile node to evaluate the throughput of the backbone and ad-hoc paths, and to make a decision of taking one path. The proposed algorithms are implemented in OPNET simulator, and network performance is studied under variety of conditions.

Table of Content

Author's Declaration	ii
Acknowledgement	iii
Abstract	iv
Table of contents.....	v
List of Acronyms.....	vii
Keywords.....	vii
List of Figures.....	viii
List of Tables.....	x
List of Equations.....	x
List of Algorithms.....	x

Chapter 1: Introduction 1

Chapter 2: Related Works 4

2.1 Wireless Mesh Networking.....	4
2.2 Wireless and ad-hoc routing protocols.....	4
2.3 WMN routing protocols	5
2.4 Integrated Routing Protocol for WMN	8
2.5 Throughput analysis for WMN	11

Chapter 3: Architecture of WMN 12

3.1 WMN global connectivity and addressing	14
3.2 WMN Routing.....	15
3.3 WMN Structure.....	17
3.3.1 Backbone Network	17
3.3.2 Access Network.....	19
3.3.2.1 Access network including MNs with two interfaces	19
3.3.2.2 Access network including MNs with a single interface	20
3.4 Interference in Backbone and Access networks.....	21
3.5 Throughput Analysis.....	23
3.5.1 Throughput in the mesh network.....	23
3.5.2 Throughput in the ad-hoc access network	25
3.5.3 Summary of Throughput analysis.....	28

Chapter 4: WMN Integrated System Design..... 29

4.1 Access Network	29
4.2 Integrated system	31
4.2.1 Throughput calculations	33
4.2.1.1 Throughput calculation for backbone	33
4.2.1.2 Throughput measurements for ad-hoc path	34
4.2.2 Route Discovery	37
4.2.3 Path Selection: Backbone and Alternative Ad-Hoc paths	40

Chapter 5: Simulation and Results.....	42
5.1 Modeling of Wireless Mesh Network in OPNET	43
5.1.1 Modeling basic WMN network in OPNET	43
5.1.2 AMR and WMR Configuration.....	45
5.1.3 Mobile Node (MN) Configuration	46
5.1.4 Generating MANET traffic	46
5.2 Simulation objectives	47
5.2.1 Problem statement: Contention in WMN	49
5.2.2 Solution: MN finds Alternative ah_path, improves Throughput.....	49
5.2.3 New algorithms for MN to find the alternative path and use it.....	49
5.2.4 Bianchi's saturation throughput	50
5.3 Simulations Results	50
5.3.1 Channel contention in Backbone Path.....	51
5.3.2 Alternative Ad_Hoc Path as a solution for Contention	53
5.3.3 Decision Making Process	55
5.3.3.1 Bianchi's Saturation Throughput	55
5.3.3.2 Simulation scenarios for decision-making path selection	57
 Chapter 6: Conclusions and Future Works	 66
6.1 Conclusions	66
6.2 Future Works.....	67
 Appendix A: Simulation Environment OPNET Model files.....	 68
OPNET Simulation Environment	68
OPNET Architecture.....	68
MANET Model Architecture in OPNET	69
Node Models in MANET	69
MANET node model architecture	71
AODV Model in OPNET.....	72
Configuring AODV in OPNET	72
AODV Model Files	74
Taking results of AODV.....	77
 References	 79

List of Acronyms

AH, ah	Ad-Hoc
AMR	Access Mesh Router
AN, an	Access Network
AODV	Ad Hoc On-demand Distance Vector
AP	Access Point
BB, bb	Backbone
BSS	Basic Service Set
D_AMR	Destination AMR
D_MN	Destination Mobile Node
DSR	Dynamic Source Routing
ETT	Expected Transmission Time
ETX	Expected Transmission count
IAP	Internet Access Point
INT, int	Interface
MANET	Mobile Ad-hoc Network
MIC	Metric of Interference and Channel switching
MN	Mobile Node
OSPF	Open Shortest Path First
QoS	Quality of Service
RREP	Route Reply
RREQ	Route Request
S_AMR	Source AMR
S_MN	Source Mobile Node
WCETT	Weighted Cumulative ETT
WLAN	Wireless Local Area Network
WMN	Wireless Mesh Networks
WMR	Wireless Mesh Router

Keywords

Wireless Mesh Network, ad-hoc network, on-demand routing, Link-state routing, Proactive routing, Reactive routing, Alternative Ad-Hoc Path, backbone path, ad-hoc path, Wireless Mesh Router, Access Mesh Router, Internet Access Point, Mobile Node, Source, Destination, Saturation Throughput.

List of figures

Figure 3.1: Architecture of the proposed WMN.....	13
Figure 3.2: AMR with two interfaces	13
Figure 3.3: MN with two interfaces	14
Figure 3.4: Different 802.11 radios for bb_int and an_int	18
Figure 3.5: Mesh_path with three subsections, compared to ad_path	19
Figure 3.6: MN with one interface	20
Figure 3.7: Channel assignment for WMN.	22
A) Alternating channel in the backbone.	22
B) Alternating channels in the access network	22
C) MNs use a single channel for both communications.....	22
Figure 4.1: Access Network and ad-hoc network is using AODV routing protocol	30
Figure 5.1: Basic WMN scenario	44
Figure 5.2: Configuration parameters for AMR and WMR	45
Figure 5.3: Configuration parameters for Mobile Node (MN)	46
Figure 5.4: WMN scenario including the new source code, indicating bb_path taken by the traffic sent from S_MN to D_MN	51
Figure 5.5: Increasing number of MNs from 1 to 4 results in increasing contention	52
(A) End-to-end Delay	52
(B) Throughput at the destination.	52
(C) Retransmission Attempts.....	53
Figure 5.6: comparing bb_path vs ah_path; high contention in source cluster	54
(A) Throughput at the destination	54
(B) End-to-end delay	54
Figure 5.7: WMN scenario including 6 MNs surrounding AMR for testing Bianchi's Saturation Throughput	56
Figure 5.8: Saturation Throughput based on Bianchi's formula in algorithm 4.1.....	56
Figure 5.9: Figure 5.8 presented in Sample Sum (Cumulative Sum).....	57
Figure 5.10: WMN scenario with 2 MNs including the new source code, indicating bb_path taken by the traffic sent from S_MN to D_MN	60
Figure 5.11: WMN scenario with 4 MNs including the new source code, indicating bb_path taken by the traffic sent from S_MN to D_MN	61
Figure 5.12: Throughput values for 2 scenarios in figures 5.10 and 5.11	62

Figure 5.13: WMN scenario including the new source code with 6 MNs in the source cluster, Indicating ah_path taken by the traffic sent from S_MN to D_MN.....	62
Figure 5.14: Throughput values for scenario in figure 5.17 with 6 MNs and ah_path, and comparison with 2 MNs and 4 MNs of bb_path	63
Figure 5.15: MANET delay values for 2 scenarios in figures 5.10 and 5.11, where number of MNs in source cluster is increased from 2 to 4	65
Figure 5.16: MANET delay values for scenario 6 MNs and ah_path is selected	65
Figure A.1: Three phases of OPNET simulation model.....	68
Figure A.2: MANET model Architecture.....	69
Figure A.3: MANET Node Palette.....	70
Figure A.4: MANET process models in OPNET.....	71
Figure A.5: AODV process model.....	72
Figure A.6: Editing attributes and selecting AODV for Mobile Node (MN)	73
Figure A.7: Editing attributes and selecting AODV for WMRs	74
Figure A.8: Choose statistics to be included in the simulation from DES menu.....	78

List of Tables

Table 3.1: Bianchi's constant values.....	25
Table 5.1: values for Traffic parameters generated from S_MN to D_MN.....	47
Table A.1: List all OPNET source code files changed for WMN model.....	77

List of Equations

Equation 3.1: Calculated normalized system throughput.....	24
Equation 3.2: Value of P_s , probability of successful transmission.....	24
Equation 3.3: Probability that there is at least one transmission in a slot time.....	24
Equation 3.4: Calculated normalized system throughput.....	26
Equation 3.5: Calculated normalized system throughput expanded.....	26
Equation 3.6: Delay measurements from	27
Equation 4.1: Calculated value for permissible throughput	35

List of Algorithms

Algorithm 4.1: To calculate Saturation throughput in the backbone based on reference [31].....	34
Algorithm 4.2: To calculate permissible throughput in the ad-hoc based on reference [20]	36
Algorithm 4.3: Decision making process for the mobile node to initiate second route discovery.....	38
Algorithm 4.4: Decision making process for the mobile node path selection.....	40

Chapter 1: Introduction

Recently Wireless Mesh Network (WMN) has become popular especially for its low cost deployment in the areas of poor network infrastructure and terrain of difficult deployment. Wireless routers deployed at fixed locations and connected through wireless links form the backbone of WMN [6]. The backbone consists of Wireless Mesh Routers (WMRs). WMRs could be either at the core of WMN, or at the edge of the Access Network. In latter case they are referred to as Access Mesh Routers (AMRs), and they are head-clusters for a cluster of user devices or Mobile Nodes (MNs). User devices are connected to the network through access links. Typically every AMR in the backbone has two wireless interfaces – a backbone link connected to other wireless routers (*bb_int*), and an access link connected to the user devices (*an_int*). Each MN also has two interfaces; one connected to its cluster's AMR (*an_int*), and the other interface to other MNs in its cluster or in the neighboring clusters (*ah_int*).

Routing in WMNs is a challenging problem because of dealing with the unpredictable behavior of multi-hop wireless links caused by interference, noise, fading, and channel propagation and access characteristics. Most of the routing proposals for WMN use some form of ad-hoc routing with more innovative metrics to reflect wireless link conditions, e.g. ETX [21] or WCETT [23]. In this thesis we propose an integrated routing system for WMN that utilizes both backbone and access links to achieve the benefits of multi-path routing. The WMN backbone is formed by fixed wireless routers; hence its topology does not change frequently. This allows us to employ link-state routing in the backbone. User devices that are connected to different wireless routers in the backbone can also be connected to each other through their access links to form ad-hoc network. Since user devices are not fixed rather they move on their own, the topology of the ad-hoc network undergoes more frequent changes. Hence, we propose on-demand routing in ad-hoc access network. Thus, the proposed integrated routing system for WMNs provides a source with at least two alternative paths – one through the ad-hoc access network called the *ad-hoc path*

(*ah_path*) and the other through the backbone network called the *mesh_path* (*mesh_path*). The *mesh_path* consists of a path through the backbone network between source and destination AMRs (called *bb_path*), the access link between the source MN and its corresponding AMR, and the access link between the destination MN and its corresponding AMR. Since wireless routers in the backbone are fixed and connected to permanent source of power, the backbone portion of *mesh_path* are relatively more stable with no power constraints. In contrast the access paths are relatively less stable with power constraints due to mobility and limited power source of user devices. Hence, we suggest that *mesh_path* is used as a primary path. There are at least three situations when *ah_path* has benefit over the corresponding *mesh_path*. First, the high contention on the access link between the MN and the AMR can significantly reduce throughput of the *mesh_path*. Second; when the excessive handover delay after a MN moves from one AMR to another causes transient *mesh_path* outage. This can be mitigated by setting up alternative *ah_path*. Third, when the *ah_path* has fewer hops than the corresponding *mesh_path*. This typically happens when the source and destination MNs are in the adjacent clusters. For example, two mobile nodes in adjacent clusters if come close to each other such that they can directly communicate, the single hop *ah_path* may become superior to the corresponding *mesh_path*. In the proposed routing system we employ OSPF in the backbone network and AODV in the ad-hoc access network [32].

Proposed routing solution uses throughput as a performance measure to compare the quality of *mesh_path* and *ah_path*. We propose two algorithms by which the MN could evaluate the performance of the *mesh_path* and compare it to a threshold value. First algorithm will check to see if the performance does not satisfy the threshold requirement; the MN will issue a second route request on the ad-hoc interface (*ah_int*), and consequently will find a second route to destination MN (*d_MN*). The second algorithm will allow the MN to compare the throughput values of the two paths found, and select the best path using the knowledge of the quality of both paths.

The throughput measurements are based on the model from Giuseppe Bianchi [31] for the backbone throughput, and based on the model from M. Kazantzidis and Mario Gerla [20] for the ad-hoc paths.

The rest of this thesis is organized in the following order. In chapter 2 we will review related works in various different areas related to this thesis, and present the literature that we have used to support the thesis. In chapter 3 we present an architecture for the Wireless Mesh Network that we are building. In chapter 4 we will explain in detail the design of this architecture, as well as design of 4 algorithms that we use to calculate the throughput values and for the mobile node to be able to evaluate the performance of backbone and ad-hoc paths based on their reported throughput. In chapter 5 we will present the simulation of our design, an implementation of all the algorithms in OPNET modeler as well as the results produced from the simulation. Chapter 6 will cover conclusion and future works. Finally in appendix A, we will present a comprehensive detail explanation of how we used OPNET, including source code changes for all the OPNET files that we used.

Chapter 2: Related Works

2.1 Wireless Mesh Networking

Wireless Mesh Network (WMN) has been a hot area in the wireless research community in the past few years. V. Bahl and his colleagues in Microsoft made a major contribution to WMNs by developing WMN architecture in the test environment and making it available to the research community [2]. D. De Couto of MIT and his colleagues have deployed WMN in real life scenario in their RoofNet project where they have placed many Wireless Mesh Routers (WMRs) in different areas in the city of Cambridge [3]. Roofnet is an experimental deployment of Grid routing protocol [4] where collection of radio-equipped nodes automatically configure a cooperative network in a self-organizing manner. S. Das et al at Stony Brook University have also deployed WMN architecture in a testbed environment [5]. In their development, APs act as a WMR and are equipped with multiple 802.11 MAC interfaces operating in different bandwidth and different channels within each band. A distributed channel assignment algorithm assigns channels to interfaces so that a basic network topology is maintained and throughput is maximized.

F. Akyildiz et al [6] and R. Bruno et al. [7] presented comprehensive reviews of WMN, covering many areas of research activities. One area of particular interest is the architecture of WMNs [6]. The paper clearly identified three different architectures for WMNs, namely Infrastructure/Backbone, Client, and Hybrid WMNs [6]. To this list we wish to add a new architecture, which we call long haul architecture, where a long highway, or street in metropolitan is equipped with a long row of WMRs.

2.2 Wireless and ad-hoc routing protocols

Conventional routing protocols designed for wired networks could not satisfy the unique characteristics of ad-hoc networks, leading to the design of new routing protocols exclusively for ad-hoc networks. Mobile Ad-hoc NETWORK (MANET) is

characterized by mobility of nodes, limited power supply, and unstable routes. These characteristics of MANET introduce continuous topology changes that create enormous amount of overhead, calculations and flooding if conventional protocols were used. Several new routing protocols have been proposed to overcome many shortfalls of the traditional protocols when used for ad-hoc networks.

Vast amount of research have focused on routing for ad-hoc networks in the past decade. Several surveys are available covering and summarizing numerous publications in different areas of routing for ad-hoc networks; [8], [9], [10] just to name a few. Numerous routing protocols propose hierarchical routing, cross-layer designs, clustering and so on. One of the most common ways to characterize those routing protocols is to divide them into reactive and proactive groups. Proactive protocols, such as DSDV [11] and OLSR [12] keep routes in their routing table, and periodically update them. Reactive protocols, such as AODV [13] and DSR [14], on the other hand work on a need-driven basis, where a route discovery is only initiated based on-demand.

Other unpredictable behaviors of wireless networks such as channel fading, medium access contention, and interference motivated a new area of research called Cross-layer routing protocols (explained in section 2.3)

2.3 WMN routing protocols

Wireless medium characteristics affect behaviors of wireless networks in many ways. Such characteristics include channel fading, medium access contention, interference, as well as other physical and MAC layer issues. To this effect, routing protocols for wireless networks, albeit at network layer, should be able to address such lower layer problems. This point leads to the idea of cross-layer design for routing protocols where the lower layer characteristics could be informed to network layer in forms of new metrics that could be incorporated into layer-3 packet header. Reviews of cross-layer designs and proposed metrics are presented in [15] and [16]. Iannone [17]

introduces new metrics for interference and packet success estimation ratios that are communicated among Physical, MAC and Network layers.

To add even more complexity to wireless medium, MANET is characterized by high mobility and low power supply. These features also depend on physical and MAC layer characteristics. In several studies, researchers have shown that traditional routing metrics such as hop-count are not suitable for ad-hoc networks. Introduced by D. De Couto et al. at MIT, the idea of “Shortest Path is not Enough” [18] has become a new paradigm attracting many researchers to introduce several new metrics for ad-hoc routing protocols. They believe any new metrics for MANET or Mesh routing protocols should carry link quality or physical layer information.

The backbone of WMNs is characterized by fixed routers with unlimited power supply and high-speed connections. Consequently, the backbone routes are more stable with higher throughput and bandwidth. However, most ad-hoc routing protocols such as AODV, DSR, OLSR work based on minimum hop count. Minimum hop count does not guarantee best path in most cases, due to the fact that shortest path could involve links with low bandwidth. Therefore researchers looked for routing metrics that take into account link-quality such as throughput, delay etc.

Designing appropriate metric has major impact on the backbone routing. Shortest paths in wireless networks are not necessarily the high throughput paths [18]. ETX is a new metric that measures the Expected Transmission counts of successful packet deliveries is defined in [21], which is effectively used in selecting high throughput paths. ETX is rendered ineffective if WMRs are configured with multiple interfaces, as shown in [22]. Since ETX finds links with low loss rates, in many cases it ignores high bandwidth paths. For example, ETX tends to choose 802.11b as it shows lower loss rate than 802.11a, even though it provides much less bandwidth. Hence, new metrics ETT and WCETT are proposed in [23], which measure Expected Transmission Time and Weighted Cumulative ETT. These metrics can be used to find paths with higher throughput and lower interference. Benchmark for evaluating new WMN metrics is proposed in [24], which is used to show that WCETT only addresses

intra-flow interference and is not isotonic (i.e. cannot guarantee loop-free paths). Therefore it is not a good choice for proactive link state and distance vector protocols. It is only good for on-demand protocols. Reference [24] also proposes a new metric called MIC (Metric of Interference and Channel switching), which favors paths that use less channel time. Hence, it takes into account inter-flow interference as well as intra-flow interference. It is discussed in [25] that hop count is still better than link quality metrics, such as ETX, WCETT etc, for ad-hoc networks because frequent topology changes cause those metrics to re-compute link quality. The repetitive computations introduce significant delay and reduce throughput.

On the same note, Drave et al [25] compared link quality metrics with traditional hop-count. They also concluded that in ad-hoc networks where mobility is high, topology keeps changing. For this reason, any kind of link quality metrics (ETX, ETT, WCETT) would not perform well, since every time topology changes, they should recalculate the link quality metrics with the new topology. All these repetitive calculations will introduce large amount of delay and reduce throughput. All those studies conclude that the ad-hoc nodes act quickly to fast topology changes, therefore hop-count would perform better than link quality metrics in ad-hoc networks.

Link quality metrics are computed in network layer by measuring packet counts [26]. Another approach is to use cross-layer design to compute a metric. In [27] [28] new metrics for interference and packet success estimation ratios are proposed that are communicated across Physical, MAC and Network layers. There are also other studies that show QoS parameters could also be incorporated in the routing by using QoS metrics [29].

In our design, we look at the link quality metrics for two types of paths between a source and destination mobile nodes. Backbone (bb_path) and ad-hoc paths (ah_path) could show very different link qualities with respect to each other. Such differences between characteristics of bb_path and ah_path indicate that in designing

a routing protocol that includes fixed backbone and mobile ad-hoc nodes, we should not try to compare paths from bb and ah with respect to the incomparable metrics.

In WMNs `bb_path` is the main part of a `mesh_path`. `Mesh_path` is defined as the path between source Mobile Node (`s_MN`) and destination Mobile Node (`d_MN`) that goes through the backbone. A secondary path called `ah_path` is defined as the path between `s_MN` and `d_MN`, which goes through the access ad-hoc network. The integrated routing protocol finds mesh and ah paths separately in different times. `bb` and `ah` paths each represents best path within its own network. We use link quality metric ETX in order to find multiple `bb_paths` and to select the best path among those. In the ad-hoc network, we use hop count to find the best path just as in regular AODV. The `mesh_path` is used as the primary route until conditions fall below a threshold for throughput or delay, at which point in time the MN will switch to `ah_path`. It starts a new route discovery via ad-hoc network to find the best path using hop count and use it as a backup route, until the `ah_path` also falls below the threshold. At this point, the MN should compare the throughput or delay of mesh and ah paths, and choose the path with higher throughput or lower delay. This will also guarantee that the MN will switch to ah path when it is moving between clusters and experiences latency, disconnection, or congestion, in all cases that throughput will fall below the threshold and end-to-end delay will rise above its threshold.

2.4 Integrated Routing Protocol for WMN

WMNs were introduced to overcome shortfalls of ad-hoc networks such as power shortage and mobility issues of ad-hoc nodes. Most important of such shortfalls are in routing. As explained in section 2.2, ad-hoc networks could not use traditional routing protocols mainly due to ad-hoc characteristics such as mobility and power constraints. But the good news was that WMNs do not suffer from those constraints. WMNs are characterized by fixed WMRs in the backbone that have unlimited power supply. So, theoretically traditional protocols with some modifications and improvements could be used again. New solutions involving these ideas usually ignore ad-hoc constraints

and try to improve routing performance in the backbone by introducing new metrics to the original protocols (explained in section 2.3)

Research in routing for the backbone has mainly concentrated around new metrics that promise performance improvements. However, we should not ignore the fact that WMN is not all about the backbone. A complete path between source and destination is an end-to-end path comprising of three distinct component sub-paths. Sub-path1 runs from s_MN to s_WMR (source wireless mesh router). Sub-path2 runs between s_WMR and d_WMR (destination wireless mesh router, this is the part that entirely falls into the backbone). Sub-path3 runs between d_WMR and d_MN (Figure 3.6). So the last mile from both ends still falls into the ad-hoc network. We call this path *mesh_path* that runs between s_MN and d_MN via backbone. We believe that most of the challenges facing WMNs, such as increased delay and loss of throughput due to contention fall into the last mile between the WMRs in the backbone and the multihop end users in the access side.

In order to address routing in WMNs, we must clearly distinguish the characteristics of backbone and access, and realize the fundamental differences between the 2 distinguished parts of the network. WMN is comprised of fixed backbone, and mobile ad-hoc access sides. An integrated routing protocol that could address the needs of both networks should be aware of the path characteristics of each network and treat each network according to its own characteristics.

To our knowledge, only a few researches have looked at integrated routing through backbone and access in WMNs. Iannone et al. proposed MeshDV [30], which is a comprehensive routing protocol that takes into consideration both the backbone and the access sides of WMN. MeshDV combines proactive routing for the backbone with a reactive component for the client side. In MeshDV architecture, there is a client manager module that keeps two tables: a Local Client Table (*LCTable*) and a Foreign Client Table (*FCTable*). The *LCTable* holds information on all the clients associated with the WMR (Wireless Mesh Router), similar to MNs in our clusters, and a list of all WMRs that have inquired about these MNs. The

FCTable holds information on all non-local clients, and a pointer to their corresponding WMR. In their solution, WMRs perform all the works, and hold all the information. Mobile nodes are not involved in routing decisions. The backbone is transparent to the mobile node. Like MeshDV, we also consider both backbone and ad-hoc access for routing. However, in our solution the routing and decision-making is distributed between WMRs and MNs. We use OSPF route table instead of FCTable, and do not need to keep routes from non-local clusters in the route table of each WMR. We also use regular AODV cache table instead of LCTable

We propose an Integrated Routing protocol for WMNs that comprises proactive and reactive subcomponents. For the backbone with low or no mobility we propose a proactive protocol, OSPF-v2, and for the access side with higher mobility and limited power supply, we propose a reactive protocol, AODV. Since the `bb_path` includes both ad-hoc and fixed parts, the proposed solution uses AODV for the ad-hoc portions of the `bb_path`, and OSPF for the fixed backbone portion. We use the `bb_path` as the primary path until it becomes unavailable or congested, in which case we further enable the MN to be aware of the new condition, and discover a new path through the ad-hoc network, and be able to switch from `bb_path` to `ah_path` as needed and improve the overall performance of WMN routing.

On another note, most proposed WMN routing solutions to date, improve performance based on link quality solutions to overcome link failure. However they do not address node related issues such as node failure, DoS and cluster-head hotspot congestion issues. Although rare, but a node failure could potentially disconnect the corresponding cluster from the network. Therefore, we believe a comprehensive routing solution should address node failure issues as well. Our proposed solution will also address node related issues by providing an `ah_path` which is completely independent of the WMRs and the backbone, and could be used as a backup to the `mesh_path`, should a WMR fail or become totally unreachable due to DoS.

2.5 Throughput analysis for WMN

In our design, performance analysis is based on throughput measurements. Throughput is mainly selected because of the effects of channel access contention between the MNs when the number of MNs in a cluster increases. We consider two types of throughput measurements; first the throughput in the mesh path including the two access parts and the backbone part. For this part of throughput analyses we consider analytical Saturation Throughput works done by Bianchi [31]. Then in the second part we look at the throughput for ad-hoc path separately. For the ad-hoc throughput, we looked at the methods based on the throughput work done by Gupta and Kumar [19] and Menthos Kazantzidis [20]. Having measured the throughput for each path separately, in chapter 4 we propose an algorithm for the MN to be able to compare the two throughputs and make a decision to which path it should take based on the throughput performance of each path.

Chapter 3: Architecture of WMN

In reference [32] we presented the architecture of WMN. In this chapter we review the architecture. In chapter 4 we show how we design our model of WMN based on this architecture. In chapter 5, we show our OPNET implementation of the design, and finally simulation and results based on this implementation

In this chapter we present architecture for two different solutions. The original solution presented in section 3.3.2.1 is a complete WMN routing system including both backbone and ad-hoc networks. This solution includes MNs with two interfaces. MN uses one interface for `bb_path` communication, and the other interface for `ah_path` communication. While switching to `ah_path`, MN will also switch the channel and use a different channel for `ah_path` communication. The alternative solution presented in section 3.3.2.2 is based on MNs with a single interface and the same channel for both `bb_path` and `ah_path` communications. This solution is presented for implementation purposes. Most of the concepts such as global connectivity, routing, interference, and throughput measurements are the same for both solutions with minor differences explained in each section as required.

The architecture consists of a backbone network of Wireless Mesh Routers (WMRs) and clusters of Mobile Nodes (MNs) that are connected through the backbone, as shown in Figure 3.1. Each MN is connected through an access link to a WMR, which serves as a gateway to the backbone network. WMRs that perform this functionality are called Access Mesh Router (AMR). The difference between a WMR and AMR is that as opposed to AMRs, WMRs are in the core of mesh network and do not have access functionality, nor do they have any MNs connecting to them. Some WMRs in the backbone are connected to and serve as gateways to the Internet for the entire WMN. Those WMRs are called Internet Access Point (IAP)

Figure 3.1: Architecture of the proposed WMN

Each AMR has two 802.11 interfaces; the backbone interface (bb_int) and the access network interface (an_int) as depicted in Figure 3.2. We use different radios for the bb_int and an_int to eliminate interference between the two paths. All bb_ints are equipped with 802.11a radio and connect AMRs to the backbone, whereas an_ints use 802.11b/g and connect AMRs to the MNs in the access network. Both bb_int and an_int are configured in 802.11 ad-hoc mode.

Figure 3.2: AMR with two interfaces

Each MN is also equipped with two interfaces, an access network interface, called an_int; and an ad-hoc interface, called ah_int as shown in Figure 3.3. Each MN

is connected with the backbone via AMR through an_int. It uses ah_int to form the ad-hoc network of mobile nodes, which is discussed in Section 4.1. Both interfaces can be implemented using 802.11b radios configured in ad-hoc mode on different channels.

Figure 3.3: MN with two interfaces

The ah_int of all the mobile nodes in the network are configured on a single channel, called common ad-hoc channel, to form the ad-hoc network.

3.1 WMN global connectivity and addressing

The mesh network is an IP network which is connected to the Internet via IAPs. A WMR may be connected to multiple other WMRs through the bb_int. We assign a different IP subnet address to each of those links. Hence, we create as many sub-interfaces (virtual interfaces) on a bb_int as the number of subnets WMRs is connected with [32]. The an_int forms the access link, which is assigned an IP subnet address. Thus, all the MNs within a specific cluster, which are connected with an AMR through AMR's access link, receive an IP address on that subnet. All those MNs that are connected with the same AMR form this cluster, where the WMR becomes the clusterhead of that cluster, thus called the Access Mesh Router (AMR). When a MN approaches the vicinity of an AMR, it receives the an_int beacon and connects with the AMR. If it moves from the coverage area of the old AMR to the new AMR, then it performs handover and changes its IP address by acquiring a new address on the subnet of the an_int of the new AMR. We allow the connectivity between a MN and its AMR through a multi-hop path composed of mobile nodes

within the same cluster. Hence, a cluster of mobile nodes and the associated AMR forms an ad-hoc network. The mobility at IP level could also be managed by employing a variation of IP mobility solution discussed in [33]. For the purpose of this thesis, we have assigned IP addresses statically to AMRs. Each cluster receives an IP subnet in the range of 172.16.N.X, where N is the cluster number and X is the host or any other MN within that cluster. We have selected the AMRs to be the gateway, and given them the IP address 172.16.N.1.

3.2 WMN Routing

The WMN is composed of backbone and access networks. We propose an integrated routing system for the WMN that takes into account characteristics of both backbone and access networks. The nodes in the backbone are stationary, and routes are stable, so we propose using OSPF, a link state protocol. Some commercial WMRs also use OSPF in the backbone [34]. Generally access is a single hop link between MNs and AMRs; but in our WMN model, we consider: (i) access links could be multi-hop formed by an_int between MNs and AMRs, and (ii) the ah_int of mobile nodes form an ad-hoc network. As explained in section 4.1, the path in the ad-hoc access network between two mobile nodes can be used selectively as an alternative to the primary paths in some situations; hence we propose a reactive protocol, such as AODV for the access network.

Between source (S_MN) and destination (D_MN) there are 2 paths; ah_path and mesh_path. For ah_path we use regular AODV multihop going through the ad hoc network. The mesh_path has 3 components; sub-path1 between S_MN and S_AMR, sub-path2 between S_AMR and D_AMR (also called bb_path) and finally sub-path3 between D_AMR and D_MN. Sub-path1 and sub-path3 although part of the route going through the backbone, but in fact they are access links and use simple AODV to establish that link. For sub-path2, which lies entirely through the backbone, we propose OSPF with some modifications [32]. However this thesis focuses on the access part of this network, therefore for the purpose of this thesis, we mainly focus on AODV.

QoS parameters and link quality metrics such as ETX and WCETT perform well for sub-path2 inside the backbone. Proposed routing protocols for the bb_path uses such metrics to find the multiple paths in the bb and select the best path among those. But the main problem lies in sub-path1 and sub-path3 of the mesh_path; these parts are on-demand, use AODV and experience several limitations such as hotspot congestion, limited bandwidth, channel contention, rate drop and so on. Also there are other possible problems that have not been addressed in any of studies we have seen; such as failure and Denial of Service (DoS). Due to such issues sub-path1 and sub-path3 could affect the mesh_path by lowering throughput and raising delay.

In order to deal with such problems, we propose in this thesis using an alternative Ad-Hoc Path (ah_path) that could be used as a secondary route when mesh_path is unavailable, congested, or does not meet a certain QoS requirements. Since we are using AODV for sub-path1 and sub-path3, these two parts should also be treated as an on-demand routing. They only initiate route discovery if route is needed. The sub-path2 is established using proactive routing, which means AMRs are aware of routes in the backbone, however they do not know about ad-hoc routes in clusters other than their own local cluster.

A routing scenario works as the following: S_MN sends AODV RREQ to S_AMR. S_AMR has the route to the subnet of D_AMRs, but does not know about the exact address of D_MN. However, since D_MN and D_AMR are in the same subnet, it could forward the packets to the D_AMR. At this point D_AMR can keep the packets and refer to the cache table to find the final destination and forward the packet accordingly.

One modification in AODV is that AMRs acting as head clusters periodically send beacons to discover neighbors (all the MNs in their respective clusters), and proactively keep their local cluster's MNs in their AODV cache tables (or IP forwarding table). So when an AMR receives a packet from other AMR, it will find the subnet and forwards the packets to the corresponding AMR, which in turn forwards the packet to the D_MN.

On the access side, MNs use regular AODV, so if the D_MN is located in the same cluster as S_MN, then the route is discovered and packets are sent directly to the D_MN with AMR not intervening at all. And if the AMR receives a RREQ in which D_MN is in the same subnet as the S_AMR, the packet is dropped, assuming that there is a direct connection between the two MNs in the same cluster.

3.3 WMN Structure

The structure of WMN was described at the beginning of this chapter (Fig 3.1). In the following sections we describe different components of the proposed Integrated Routing system for WMNs. WMN consists of 2 major parts; Backbone and Access, and several components including 2 types of nodes and 2 types of paths. Nodes include Wireless Mesh Routers (WMRs) that have 3 subtypes; Access Mesh Router (AMR), Internet Access Points (IAPs) and other WMRs that are in the core of the mesh network and do not connect to any MNs nor to Internet. The second type of WMN nodes are Mobile Nodes (MNs). There are 2 types of paths; Mesh Path (mesh_path) and Ad-hoc path (ah_path). Mesh path has 3 sections; 2 access links, from MNs to the AMRs and a backbone path (bb_path) between the source and destination clusterheads, namely S_AMR and D_AMR.

In section 3.3.1 we present the structure of backbone path, and then in section 3.3.2 we will explain the access network structure.

3.3.1 Backbone Network

The backbone includes several WMRs connecting to each other and creating a mesh network. These routers have three different types. The first type of WMRs are AMRs that connect to access networks as clusterhead. Each clusterhead or AMR has two 802.11 interfaces; bb_int and an_int. We use different radios for the bb_int and an_int to eliminate interference between the two paths. All bb_ints are equipped with

802.11a radio and connect AMRs to the backbone. `an_int` runs 802.11b/g and connect the AMR to the MNs in the access network (Fig 3.4).

The second type of WMRs is the IAPs. These are the only WMRs that have wired Ethernet interfaces as well as wireless interfaces, and connect to Internet to enable the WMN to communicate with the public network. These WMRs are super-power routers that act as gateways for the entire network. The last types of WMRs are neither AMRs nor IAPs. These WMRs are in the middle of the WMN, and act only as forwarding routers to ease the mesh functionality of the WMN.

Figure 3.4: Different 802.11 radios for `bb_int` and `an_int`

WMRs in the backbone are using different channels to communicate with other WMRs in the backbone. This solution was proposed by [30]. In their work they used both interfaces in the backbone. However we use only one interface in the backbone, therefore we have implemented this idea using subinterfaces on each physical interface. Using multiple-NIC cards has been approached by many research groups [35] [36]. In most studies however, one radio (802.11 a or b) is used, separated by multiple channels. They have proposed algorithms for a central and dynamic channel assignment to switch the channel either per packet basis [37] or for longer period of time to eliminate delay caused by numerous calculations [38]. In our case, however we used 2 different radios, and introduce multiple-channel solutions in the backbone and access networks separately.

The link or path component of WMN includes 2 types; `mesh_path` and `ah_path`. `Mesh_path` includes 3 sub-paths; MN-AMR, AMR-AMR, and AMR-MN sub-paths (figure 3.5).

Figure 3.5: Mesh_path including three subsections, 2 ad-hoc portion and one bb_path, ah_path is between source and destination via ad-hoc network

The MN-AMR and AMR-MN links are within the ad-hoc networks and possess all ad-hoc characteristics. The AMR-AMR path is the only path through the backbone path (bb_path) which is between the fixed mesh routers. This path is part of a bigger mesh network and is considered a highly stable path, both due to lack of mobility and stable power source.

3.3.2 Access Network

The access network, as described at the beginning of this chapter, is presented in two slightly different solutions, first solution presented in 3.3.2.1, is based on MNs with two interfaces, using separate channels for each communication. The second solution presented in 3.3.2.2, is based on MNs with a single interface and single channel. The rest of design and implementation in chapters 4 and 5, are mostly based on the second solution.

3.3.2.1 Access network including MNs with two interfaces

The access network includes series of clusters consisting of several MNs, all using 802.11b/g, and separated using alternating channels on the ad-hoc network. Each MN is equipped with two interfaces; access network interface (an_int) that is used to connect to the AMR, and ah-hoc interface (ah_int) used to connect to other MNs in

the ah_path. Both interfaces use 802.11b/g radio; however the ah_int is configured on a separate channel to connect to ah_int of other MNs in the ah_path.

The access network includes the access link of the mesh_path on the upstream of MN to AMR. On the downstream from MN to other MNs, the access network includes ah_path between S_MN to D_MN, which could traverse through multihop of several ad-hoc MNs until it reaches D_MN.

3.3.2.2 Access network including MNs with a single interface

An alternative solution is to have MNs use the same interface to communicate with the AMR and with the neighboring MNs. Mobile nodes with two interfaces could present complexity and high cost. For the purpose of this thesis, and to avoid extra complexity, we have performed our experiments with a single interface as shown in figure 3.6.

Figure 3.6: MN with a single interface

Using 1 or 2 interfaces have advantages and disadvantages. Usually 2 interfaces have several advantages, such as ease in dealing with interference, enabling use of multiple radios, and enabling nodes to connect multiple networks without causing problems. However using 1 interface, we need to switch the channel. Then we have to use channel switching and scheduling algorithms. Using multiple radios, routers have to deal with complicated algorithms for scheduling, radio and channel assignment. For the purpose of this thesis we choose to use a single interface for the MN to connect to backbone and ad-hoc networks.

3.4 Interference in Backbone and Access networks

In WMN routing protocol we must address different types of interferences. First type of interference is the **backbone** interference (Figure 3.7 A) when the WMRs are using different interfaces to communicate with different neighbors. To eliminate this type of interference, reference [39] has introduced a careful channel assignment, such that when a WMR has multiple communications simultaneously using multiple non-interfering channels with several neighbors, such communications will not interfere with each other (Figure 3.7 A).

Second type of interference is **inter-cluster** interference (Figure 3.7 B). This is when a MN's communication with its AMR might interfere with communication of the MN in the neighboring cluster with its own AMR. To eliminate this type of interference, we could use alternating channels for neighboring clusters. In this scheme the neighboring clusters use different channels such that there will not be interference between the access interface of the two neighboring AMRs or between MNs from the neighboring clusters. This will ensure that the clusters are totally independent, and also that the MN is communicating with only one clusterhead at a time, which means that each MN belongs to one cluster only at each specific time. When the MN moves from one cluster to the next, it will have to switch to a different channel, receive beacon from a new clusterhead, acquire a new IP address, and eventually drop the old IP address, and use the new one.

The third type of interference is **intra-cluster** interference. That is if the MN use different interfaces to hold two communications within its own cluster simultaneously, one with the AMR and the other with another MN in its own cluster. Then these two communications should use separate channels to avoid intra-cluster interference. We propose one common channel between all the MNs to be used only for ad-hoc communication (Figure 3.7 C). So MNs will use their cluster-designate channel to communicate with the backbone, and when switch to ad-hoc communication they should switch to common ad-hoc channel. Reviews of different

kinds of interferences for wireless and mobile ad-hoc networks are presented in references [39] and [40].

Figure 3.7: Channel assignment for WMN.
A) Alternating channel in the backbone.
B) Alternating channels in the access network
C) MNs use a single channel for both communications

According to architecture in figure 3.7 (B, C), the MN should use a different channel to communicate with the AMR and with the other MNs. This scheme would suite solution in 3.3.2.1 where the MN uses 2 interfaces to communicate with backbone and ad-hoc networks. However, in the case of alternative solution in 3.3.2.2 where MN uses a single interface, if the MN changes the channel as it moves from bb_path to ah_path, it could cause the traffic to drop. As mentioned in section 2.1, complex channel assignment and scheduling algorithms have been proposed that would allow nodes to switch the channel automatically. However such algorithms and their implementation are not in the scope of this thesis, therefore in our experiment

we have used the same channel for communication between the MN-MN and MN-AMR.

3.5 Throughput Analysis

In this thesis we use throughput as the routing metric to discriminate between the `bb_path` and `ah_path`, and eventually decide to use one over the other. Throughput is mainly selected because of the effects of channel access contention between the MNs when the number of MNs in a cluster increases. We consider two types of throughput measurements; first the throughput in the mesh path including the two access parts and the backbone part. For this part of throughput analyses we consider analytical throughput works done by Bianchi [31]. Then in the second part we look at the throughput for ad-hoc path separately. For the ad-hoc throughput, we use the method proposed by Kazantzidis et al [20]. Once the throughput is measured for each path separately, we propose an algorithm (algorithm 4.4) for the MN to be able to compare the throughput of the two paths and select the path giving higher throughput.

3.5.1 Throughput in the mesh network

The backbone throughput analysis is based on 802.11 DCF performance studies from reference [31]. Bianchi's saturation throughput model is used by many groups in this area of research. We also adopt this model to calculate the cluster throughput at each AMR.

Reference [31] estimated the saturation throughput of a WLAN BSS, calculated in an AP based on an analytical evaluation using a probability function involving a 2-D Markov Chain for the backoff window size as shown in equation 3.1. Based on this model, we conclude that saturation throughput for each cluster could also be calculated in each AMR in the same way that the throughput of BSS is calculated.

$$S = \frac{P_s P_{tr} E[P]}{(1-P_{tr})\delta + P_s P_{tr} T_s + P_{tr} (1-P_s) T_c}$$

Equation 3.1

Calculated normalized system throughput [31]

Where:

P_s is the probability that a transmission occurring on the channel is successful and could be calculated based on equation 3.2.

P_{tr} is the probability that there is at least one transmission in a slot time, and could be calculated based on equation 3.3.

δ is the duration of an empty slot time

$E[P]$ is the average packet payload size

T_s : average time channel is sensed busy while a successful transmission occurs

T_c : average time channel is sensed busy while a collision occurs

Constant values are estimated by reference [31] for T_s , T_c , and δ . Therefore this leaves values of P_s , P_{tr} , and τ to remain unknown. Those values could also be calculated using equations 3.2 and 3.3.

$$P_s = \frac{n\tau(1-\tau)^{n-1}}{P_{tr}} = \frac{n\tau(1-\tau)^{n-1}}{1 - (1-\tau)^n}$$

Equation 3.2

Value of P_s , probability of successful transmission

(Probability that exactly one station transmits on the channel if at least one station transmits)

$$\tau = 2/W+1$$

$$P_{tr} = 1 - (1 - \tau)^n$$

Equation 3.3

Values for P_{tr} (Probability that there is at least one transmission in a slot time) and τ

Where:

τ is the probability that a station transmits in a random time slot

n is the number of stations contending

Based on Bianchi's analytical framework, maximum saturation throughput for a BSS (in our case for a cluster) is measured at the corresponding AP (or AMR). So each AMR will have the information for the throughput of its own cluster, which is propagated to other AMRs, through either the backbone routing control packets or through direct communication between every pair of AMRs. The throughput information could be piggybacked on the OSPF LSAs or in the Hello packets. In this study we will not get into detail of methodology of how OSPF is modified to carry throughput information. Reference [31] makes several assumptions to make its analytical calculation of throughput and constant values are considered for the parameters presented in table 3.1.

W	δ	T_s	T_c	$E[P]$	m
16	1	179.68	174.25	8184	3

Table 3.1: Constant values assumed by Bianchi, used in our implementation

Where:

W is the window size

δ is in second, T_s and T_c are in μsec

$E[P]$ is the average packet payload size

m is number of nodes

Therefore each AMR has the throughput information for each route that it keeps in the route table. So when a RREQ packet comes from a MN, the AMR or any other MN will include the throughput information as well as the route type in the RREP packet. In chapter 5 where the simulation is explained, we will discuss the detail of how AODV route table and RREP packets are modified to include the throughput and route type fields. This type of link quality information included in the routing protocol information is studied in many papers where QoS issues are studied [28] and [29].

3.5.2 Throughput in the ad-hoc access network

The throughput in the ad-hoc network is based on reference [20]. Reference [20] offers a practical method for measuring 802.11-based link throughput and permissible

throughput, and in a source-destination pair of nodes, proposes propagation of this information back to the source as a QoS support mechanism.

In order to measure the permissible throughput in an ad-hoc network we should take into account the medium access contention for each link as well as changes due to mobility. Reference [20] proposes the following Equation for throughput measurements:

$$r(s,d) = (1-u) * \text{Throughput}(s,d)$$

Equation 3.4

Calculated permissible throughput [20]

Where:

$r(s,d)$ is the permissible throughput between source and destination for each link

u is the link utilization.

$\text{Throughput}(s,d)$ is the throughput measured for the link (s-d)

$$\text{Throughput}_{\text{packet}} = \frac{S}{t_q + (t_s + t_{CA} + t_{\text{overhead}}) * R + \sum_{r=1}^R B_T}$$

Equation 3.5

Measured link throughput [20]

Where:

t_q : MAC queuing time

t_s : transmission time of S bits

t_{CA} : Collision Avoidance phase time

t_{overhead} : Control Overhead time (like: RTS, CTS, ACK Header)

R : Number of necessary transmissions

$\sum B_T$: sum of Backoff time for r (for $r = 1 \dots R$)

Measurement windows are used in order to increase statistical robustness of the measurements. A window of 16 or 32 packets is sufficient to produce fast and noise tolerant measurements. In previous studies, it is observed that the high

variances per packet measurements are aggregated on a window of 32 packets [20]. However for our studies we use window size of 16, as suggested by references [31]

In order to use the packet window, reference [20] considers packet delay measurements carefully. Specially, queuing delays should be factored out. When a packet has waited in the queue, its delay can be calculated using equation 3.6 (A). If it has not waited in any queue, the delay could be calculated using 3.6 (B).

<p>(A) Delay = Now – T_{previous ACK}, when the corresponding ACK is received (B) Delay = Now - T_{arrival}</p>
--

Equation 3.6

Equation 3.5 is used to calculate the throughput of ad-hoc networks at each MN. Then propagation techniques could be used to relay the throughput information back to the source such as S_MN which requests for a route via RREQ packet. The throughput information could be piggybacked into RREP packet for AODV without making major changes in the fundamental of AODV protocol.

The implementation of equation 3.5 is based on an algorithm presented by reference [20]. This algorithm will be presented in chapter 4 (algorithm 4.2). They calculate throughput based on measurements of delay. From equation 3.5 it is observed that throughput calculation is dependent on the packet size and delay caused by this packet size. Therefore they consider a constant $c = 1200 \mu\text{sec}$ for the delay caused by the packet size (presented as t_{overhead} in equation 3.5) in order to eliminate the packet size dependence of this equation.

The measurement method for ah_path throughput calculation presented in reference [20] differs from the bb_path throughput calculation using equation 3.2 [31]. In reference [31] Bianchi's method uses an analytical scheme to calculate the throughput for the entire cluster. This is a conservative method that makes estimation rather than an actual value. Although a simple and fair estimate of throughput in the last portions of the mesh_path (namely subpath1 and subpath3), but Bianchi's method

could not be used for the ad-hoc path, where we are not concerned with the whole cluster anymore, and there are many paths to destination on the ah_path. For ad-hoc network, the method in reference [20] is selected since this is a real-time measurement and unlike the cluster throughput, the contention is not the only factor affecting the measurements.

In regular AODV only the routes that the MN uses are recorded. In our new design, we keep two best routes, one from the mesh network, and the other from the ad-hoc network. The two routes are distinguished by the route type. The throughput is added to the route table of AODV along with hop count metric to be used in building routes. Route type is used to separate the mesh_path routes from the ah_path routes. The sequence number is also used just like the regular AODV to check freshness of the route.

3.5.3 Summary of Throughput analysis

Based on sections 3.5.1 and 3.5.2, throughput values are measured in the AMR and MN for bb_path and ah_path accordingly. Then the throughput value of bb_path is reported back to the MN via RREP, and MN takes the minimum throughput of all 3 parts of mesh_path, and assigns it to mesh_path in the route table. At this point the MN has two routes to the destination, one from the mesh_path, and the other from ah_path. For each path, the throughput value is recorded in the route table along with hop count and other metrics. In chapter 4, we will present algorithm (4.4) by which the MN compares the throughput values for both routes and based on the comparison, it makes a decision which path to take.

Chapter 4: WMN Integrated System Design

In this chapter we explain the design of the proposed Integrated Routing system for WMNs. The design is based on the WMN architecture proposed in [32] and reviewed in Chapter 3. We will discuss the contribution of this thesis in designing ad-hoc network composed of AMR clusters in section 4.1 and integrated routing system including the route selection algorithm in section 4.2.

4.1 Access Network

Consider paths between a source mobile node (S_MN) and a destination mobile node (D_MN) as in Figure 3.1. As explained in chapter 3, there are essentially two types of end-to-end paths; a path that traverses the backbone, called *mesh_path*, and the other that does not go through the backbone, called *ah_path*. The *mesh_path* has 3 segments: *sub-path1* between S_MN and S_AMR, *sub-path2* between S_AMR and D_AMR and finally *sub-path3* between D_AMR and D_MN (figure 4.1). Among the sub-paths only *sub-path2* is entirely composed of links within the backbone, whereas sub-paths 1 and 3, called access network paths (*an_paths*), are composed of access links, potentially multi-hop, formed within the clusters of S_AMR and D_AMR. We could use any ad-hoc routing protocol, such as AODV, to set up *an_paths*.

Generally, the paths within the backbone (*bb_paths*, between AMRs) are more stable than *an_paths* because the AMRs are stationary nodes and the links among them are formed by directional antennas [6]. Dynamic link quality metrics such as ETX and WCETT can be used in the backbone routing to perform multi-path routing within the backbone. However, the *an_paths* are the unstable segments of the *mesh_path* due to channel contention, rate drops caused by increasing distance between a MN and the AMR, and instability due to node mobility. Hence, the *an_paths* could constrain the quality of a *mesh_path*, for example, by lowering throughput or raising delay.

We propose *mesh_path* as the primary path used between source and destination MNs because of its tendency to traverse stable backbone links. We, however propose using ad-hoc path (*ah_path*) as an alternative path when *mesh_path*

is severely constrained under the following situations. First, when an an_path become bottleneck causing reduced throughput of the mesh_path. Second, while the excessive handover delay after a MN moves from one cluster to another causes transient an_path outage. This can be mitigated by setting up alternative ah_path. Third, when the ah_path has fewer hops than the corresponding bb_path. This typically happens when the source and destination MNs are in the adjacent clusters. For instance, two mobile nodes in adjacent clusters if come close to each other such that they can directly communicate through their ah_ints, the single hop ah_path may become superior to the corresponding mesh_path.

Any ad-hoc routing protocol can be used to establish an_paths. For example, table-driven proactive protocol can be used because an_paths are part of primary paths that are used for most of the time. On the other hand the ah_path is a secondary path that should be set up only when required. Hence, for the ah_path set up we propose using on-demand ad-hoc routing protocol, such as AODV, which only initiates route discovery if route is needed. On-demand routing protocols, such as AODV, perform route discovery when a new route is needed for packet forwarding or when an existing route is refreshed in the routing cache. The route discovery process typically involves flooding of discovery packets inside the network, e.g. flooding of RREQ packets in AODV. Since routes are not discovered or refreshed periodically in on-demand routing, it incurs less flooding overhead, which is suitable for WMN ad-hoc access network. (Figure 4.1)

Figure 4.1: Access network as well as ad-hoc network is using AODV routing protocol

4.2 Integrated system

There are three key issues in designing the integrated routing system. First, which node should decide about using either a primary or alternative route? The route selection decision can be either made by the AMR or the MN itself. In either case, the MN establishes the `ah_path`. Hence, if the AMR makes the decision, then the information about the `ah_path` has to be transferred to the AMR, which necessitates discovering full `ah_path` prior to making the decision. If the MN makes the decision, then it can delay discovering the `ah_path` until after making the first decision, which is whether or not to initiate a second route discovery.

The MN can make the route selection in two steps. In the first step it decides about initiating the route discovery based on the quality of the available `mesh_path`. Then, it can decide about using the primary or the alternative path after the full `ah_path` discovery and having knowledge of quality of the `ah_path`. Then the MN performs route selection.

Second key issue in design is when to initiate the route discovery process for the `ah_path`? The `ah_path` route discovery is an expensive process; hence we argue that it should be initiated only when there is a good chance of using the `ah_path`. For instance when the throughput of backbone drops far below that of ad-hoc path (In our simulation we have considered the case where the difference in the provided throughputs are greater than 25%) We propose an algorithm for initiating route discovery in AODV, which is invoked by the mobile nodes (Algorithm 4.3). The source MN broadcasts AODV RREQ for the destination setting the AODV `RREQ_TTL = x`, where x is the number of hops the MN is away from the AMR. When the AMR receives the RREQ from the source node, it checks the destination IP address. If the destination is in the local cluster, the AMR sends regular AODV RREP if it finds the route in its AODV cache. If the destination is not in the local cluster, the AMR looks up the backbone routing table. If it finds a route to the destination, it

returns RREP with the number of hops in the mesh_path. If the number of hops is less than a threshold value, then there is a chance of discovering a shorter ah_path. For example, the number of hops along the mesh_path between two adjacent mobile nodes in different clusters is at least three, while a single hop ah_path can be discovered through the ah_int. When the number of hops is more than three, then it means that the MNs are not in the adjacent clusters and the mesh_path should be taken since the backbone is designed for this purpose.

The third important issue in design is how to decide between the quality of the mesh_path and ah_path? Dynamic link quality metrics such as ETX and WCETT are effective measures of the throughput of backbone routes [21] and [23]. However, they are not as effective in ad-hoc network [22]. A careful estimate of RTT of the ah_path can be used as a measure of its throughput. In our analysis, each AMR uses Bianchi's model to calculate its saturation throughput for its local cluster. Therefore each AMR has throughput information of its own cluster, which could be transferred to other AMRs through backhaul transmission via piggybacking with control messages, or creating a special protocol for transmitting the throughput information.

The design of an integrated routing protocol for WMN involves two major components; first is Route Discovery process in which MN finds the routes through both mesh_path and ah_path. In this section the MN evaluates performance of the mesh_path and decides whether to use this path or to discover a second alternative backup path through ad-hoc network. The second component is to evaluate and compare the route through the mesh_path and ah_path and to decide when the backup path should be used. These components and the corresponding algorithms for each component are discussed in different sections under 4.2.

4.2.1 Throughput calculations

Throughput is calculated in two separate parts; first is the throughput calculation for mesh_path based on reference [31] and then throughput measurements based on reference [20]. In the following two sections we present these two methods separately.

4.2.1.1 Throughput calculation for backbone

Each AMR calculates the throughput for its corresponding cluster based on equation 3.1. Each AMR holds all the routes for all cluster nodes in its route table. For this part each route has its throughput value as well as hop count included in the route table and RREP packet Route table entries include throughput and hop count values as well. The throughput for each route is calculated based on the minimum throughput of all the links between source and destination. Throughput calculation is based on the saturation throughput model of reference [31].

Considering the constant estimated values given by Bianchi for T_s and T_c (table 3.1), leaves us with only two unknown values; P_s which is probability that a transmission on a channel is successful and P_{tr} which is probability that there is at least one transmission occurring. These two probabilities are only dependent on the number of nodes in each cluster and are calculated based on equations 3.2 and 3.3.

A counter, n , is kept by each AMR, and incremented each time a node joins this cluster, and decremented each time a node leaves the cluster. Using n we keep track of the number of active nodes in each cluster. Once n is measured, using Equations 3.2 and 3.3, values of P_{tr} and P_s are calculated for a window size of 16. Algorithm 4.1 shows our implementation of Bianchi's saturation throughput for each cluster. This is performed at each AMR separately, and throughput values are added to available routes entries to all MNs within this cluster.

Algorithm 4.1:

```
n++; //constant value considered
W = 16,  $\delta = 1\mu s$ ,  $T_s = 179.68$  and  $T_c = 174.25$ ; //constant values by Bianchi
E[P] = 8184; //Packet Payload (bits)
 $\tau = 2/W + 1$ ;
 $m = 1 - \tau$ ;
 $P_{tr} = 1 - m^n$ ;
 $P_s = [n * \tau * m^{(n-1)}] / P_{tr}$ ;
 $T_{put} = P_{tr} * P_s * E[P] / (1 - P_{tr}) * \delta + P_{tr} * P_s * T_s + P_{tr} * (1 - P_s) * T_c$ ;
```

Note that the Bianchi throughput is calculated for the whole cluster. Then each AMR sends its calculated saturation throughput value for each route over to the neighboring WMRs included in the update packets when proactive protocol is used or to RREP packet when AODV is used. Each WMR along the way checks the throughput for each route received from AMRs, compares it to its link throughput and updates the throughput value of that route with minimum of the two, and sends it to the next WMR as the throughput of that specific route. Finally S_AMR takes the throughput for the destination route compares it with its own link throughput and records the minimum of the two in its route table as the designated throughput of the destination route. Upon receiving RREQ from S_MN, S_AMR checks the route table for the destination, and includes the route along with the throughput in the RREP message, and flags the route as bb_route by setting the route type (rte_type) as bb in the RREP message.

4.2.1.2 Throughput measurements for ad-hoc path

In the access network, MN calculates the throughput value for the ah_path using the model in reference [20] and discussed in section 3.5.2. In the implementation of this scheme, we must consider the effect of the packet size on the throughput measurement and use a measurement window. Different packet sizes are an important source of noise to the measurements. For instance, at lower loads, applications with smaller packets traveling a path would report significantly lower throughputs than paths traversed by larger packets. In order to limit the dependence to packet size a constant c is deducted from delay measurements.

At higher loads, the throughput will be lower, and not affected by the packet size as much, due to the longer collision avoidance and backoff windows. Therefore, an estimated value for c results in fairly good measurements for throughput. For our studies we go by the estimate of reference [20] and set $c = 1200 \mu\text{sec}$.

The idle time and window duration are calculated to produce the link utilization factor and the permissible throughput measurement as shown in equation 4.1. This equation basically shows how equation 3.4 is implemented in their algorithm. Idle time divided by window duration is used to measure link utilization $(1-u)$, and throughput measured is the link throughput.

$\text{Permissible throughput} = (\text{Idle time in window} / \text{window duration}) * \text{throughput measured}$
--

Equation 4.1

Calculated value for permissible throughput

In the implementation of delay, reference [31] uses equation 3.6, and provides a network layer algorithm for delay measurements. They use the following variables in the design of the algorithm. Their implementation is presented in algorithm 4.2.

LastPacketTimestamp[WINDOW_SIZE] : keeps timestamp when a packet leaves the network layer
LastPacketPayload[WINDOW_SIZE]: keeps the useful bits of the packets
VQ: number of packets sent, incremented each time a packet is sent
NTF_ptr: number of notifications received, incremented each time an Ack is received
PrevNTFTimestamp: timestamp of the last notification
SampleDelays[RECEIVERS]: sum of delays measured for packets to RECEIVER
SampleBits[RECEIVERS]: sum of bits for the packets
Window_duration: duration of the current window and *Idle_time*

Algorithm 4.2: *//this algorithm is taken directly from reference [20]*

```

VQ++;                                     //when packet is sent
{
LastPacketPayload[VQ] = <packet bits>;
LastPacketTimestamp[VQ] = now;

    If (VQ - NTF_ptr == 1)                //Queue was empty
    {
        Idle_time += (now-PrevNTFTimestamp);    //Channel has been idle
        Window_duration += (now-PrevNTFTimestamp); // since last notification
    }
}

NTF_ptr++;                               //when notification is received
{
    If (ACK received)
        SampleBits[Notif.source] += LastPacketPayload[NTF_ptr];
                                                //packet is waiting in queue
    {
        If (LastPacketTimestamp[NTF_ptr] < prevNTFTimestamp);
                                                //packet is waiting in queue
            Delay = (Now - prevNTFTimestamp - C);
        Else
            Delay = (Now-LastPacketTimestamp[NTF_ptr]-C);
    }
    SampleDelays[Notif.Source] += Delay - C;
    Window_duration += Delay;
}

THROUGHPUT MEASUREMENT()
{
    Link_Throughput= SampleBits[R]/SampleDelays[R]; //For All Measured Receivers R:
    If ( Test_Measurement() )
    {
        PermissibleT[R] = (idle_time/Window_duration * Link_Throughput);
        Update_Tables();
    }
    Reset_Variables();
}

```

As mentioned in section 3.5.2, due to fundamental differences between the backbone and access network, and the way their throughput is calculated, for the purpose of throughput comparison we will use the OPNET-provided throughput calculation for ad-hoc network for implementation in chapter 5.

At this point, after having calculated the ad-hoc throughput and having received the backbone throughput from AMR, MN will keep both mesh_path and ah_path in the route table with their corresponding throughput as well as hop-count, and distinguished by rte_type value. Then MN compares the throughput values of

both paths and uses the best path to forward the packets. To select the best path among the `mesh_path` and `ah_path`, MN will use algorithm 4.4 to decide which path to use. Details of route discovery and path selection of the integrated system including the two algorithms that the MN uses to make its routing decisions are explained in sections 4.2.2 and 4.2.3.

4.2.2 Route Discovery

Source node (`S_MN`) broadcasts an AODV RREQ for destination (`D_MN`). At the first stage, `S_MN` sets the AODV RREQ_TTL = 1. Only nodes that are a single hop away from `S_MN` receive this RREQ including the clusterhead mesh router.

This RREQ could be captured by either another MN or by a WMR. The MN could be in the local cluster or in a remote cluster. WMR could be the local clusterhead (AMR) or any other WMR along the way.

Algorithm 4.3 explains how the route discovery is implemented. We define a function for initiating `second_route_discovery` which is called every time hop count or throughput falls below a threshold amount. First the MN broadcasts a RREQ, and waits to receive a RREP. Upon receiving RREP, MN checks to see if RREP is from an AMR or another MN. If it is from an AMR, then it should call route discovery function. This function checks the hop count and throughput of the RREP, and if they fall below threshold, it initiates the `second_route_discovery` by sending a second RREQ, otherwise it will enter the RREP into the route table. If the RREP is received from another MN, then it has to check whether the next hop of that MN is an AMR. In any case, the MN still calls the function `second route discovery`. The difference is that if there was an AMR along the way, then the route type would be entered in the route table as backbone.

Algorithm 4.3:

#define function second_rte_discovery

set hc0 = 3; *//pre-set hop-count threshold*

set Tput0 = 0;

broadcast RREQ (ttl = 1);

upon receiving RREP; *//RREP includes hc, rte_type and Tput of bb_route*

If(route provider ip address == gw ip address) *//route is provided by AMR*

Call function second_rte_discovery on ah_int

else If (route provider ip address != gw ip address) *//route is provided
//by a member MN*

if (NH == AMR)

Call function second_rte_discovery on ah_int

else

for (1 to hc)

if (rte_type == bb)

Call function second_rte_discovery (on ah_int)

Enter route as bb_path

elseif (rte_type == ah)

Call function second_rte_discovery (on ah_int)

Enter route as ah_path

endif

endif

endif

end

#function second_rte_discovery

check hc

if (hc < hc0) | (Tput < Tput0)

then initiate route discovery via ad-hoc by broadcasting RREQ (ttl = hc)

else

accept the route and enter in the route table

end

When a RREQ is received by a MN, first the IP address of D_MN is checked. If the D_MN is in the same subnet, it means that it is in the local cluster. In this case a regular AODV procedure can be used to resolve the route discovery. If the D_MN is not local but the route to D_MN is available, a RREP is sent to S_MN including the D_MN IP address, its hop count, and the throughput of the route.

When an AMR receives the RREQ from S_MN it checks the D_MN IP address, if the D_MN is in the local cluster, AMR uses the AODV cache and replies with a RREP, including the IP address of the destination, just as in regular AODV. If the D_MN is not in the local cluster, the AMR looks up the routing table. If it finds a route to the destination, it returns RREP with the number of hops.

A new field is added to RREP packet format for `route_type`. `Route_type` can have values “b” (for `bb_path`) or “a” (for `ah_path`). RREPs from backbone are marked as `bb_route`, whereas RREPs from other MNs are marked as `ah_route`. Also a new column is added to AODV route table as `route_type`. Any route returned by mesh router is entered in the route table as `bb_route` or `ah_route` depending on where it comes from. Once a RREP is sent by `D_MN`, it is tagged as `ROUTE_TYPE_AH`. At any stage, if it passes by an AMR or WMR, its route type will change to `ROUTE_TYPE_BB` and will remain BB until it reaches the `S_MN`. Therefore if a RREP is tagged `ROUTE_TYPE_AH` once it reaches `S_MN`, that means this route lies entirely within ad-hoc path, and there is no backbone router on the way of this path.

Upon receiving RREP from AMR, MN has to decide whether route provided by the AMR satisfies certain set of threshold requirements. If the required metrics fall below thresholds, then the MN should start a new route discovery by sending a second RREQ using AODV expanding ring search, and finds a secondary route through ah path and uses it as a backup route.

A set of threshold values for the throughput is defined within the routing information. Upon receiving RREP from AMR, `S_MN` compares the throughput value collected from the `bb_path` to the pre-set conditions, and decides whether to use the route provided by mesh router, or to initiate a new AODV RREQ with longer TTL (Algorithm 4.3).

`S_MN` initiates route discovery by broadcasting RREQ to ad-hoc nodes, and searches for a backup route via ad-hoc multi-hop path. Upon receiving RREP from ad-hoc network, `S_MN` enters a route in the route table as `ad-hoc rte_type`.

4.2.3 Path Selection: Backbone and Alternative Ad-Hoc paths

Having performed a second route discovery, S_MN has two routes to the destination, one through the mesh backbone, and the other through the access ad-hoc network, and has to decide when to use the primary bb_path, and when to switch to ah_path. Algorithm 4.4 below is a network level implementation of decision-making process by the MN.

This algorithm is only executed if there are 2 routes to choose from. It checks the route table, if there is no route then it calls algorithm 4.3 to find the routes. If there are 2 routes available, and it wants to decide which one to take, then it checks the throughput provided by the two routes. Then it finds a threshold value called “d”. the value of d is calculated by subtracting the two throughputs, dividing them by the ad-hoc throughput and multiplying by 100. This equation will find out the difference in the two throughput values by percentage. Different network set up could try different values for d depending on how reliable is the backbone route. For the purpose of this thesis, we assumed that we will take ad-hoc path, only if ad-hoc provides a higher throughput by 25%.

Algorithm 4.4:

```
route required
check route table
if no route available
    then start algorithm 4.3
else
    check throughput fields of RREQ1 and RREQ2
    set throughput = thr_0 //threshold throughput
    get bb_path throughput = Tput_bb
    get ah_path throughput = Tput_ah

     $d = (Tput\_ah - Tput\_bb) / Tput\_ah * 100$ 

    if (d > 25)
        then ah_path → status = active
    elseif (d ≤ 25)
        then bb_path → status = active
    endif
endif
end
```

According to algorithms 4.3 and 4.4, the MN uses `bb_path` until throughput falls below the minimum requirement (threshold). Then once the MN is notified of this information, MN starts a second route discovery (Algorithm 4.3), finds the `ah_path` and starts using this path if necessary (Algorithm 4.4).

MN keeps using the `ah_path` until the `ah_path` throughput also falls below the threshold, then the MN starts comparing the throughput values of the two paths again, and selects the best path to be used. Therefore in case the `ah_path` throughput also drops, the MN should be able to run algorithm 4.4 again and switch back to `bb_path`, however this part of the algorithm has not been developed in this study yet.

These algorithms ensure that the MN will switch to `ah` path whenever throughput will fall below the threshold level. Such cases could happen when MN is moving between clusters and there is latency, disconnection, or congestion.

Chapter 5: Simulation and Results

In this chapter we will present the simulation of our WMN in OPNET Modeler 11.5 PL1. Section 5.1 includes details of the WMN model created in OPNET along with detail of different node and process models that we used, as well as traffic generated and performance measures selected for this model. In section 5.2 we will establish the simulation objectives. Also in cases that we have presented alternative solutions for design such as use of one interface versus two interfaces, we will outline clearly which case is implemented and why those solutions are selected for implementation over others. Then in section 5.3 we will explain in detail the results for throughput and end-to-end delay analysis of WMN, stating the problem arising from channel contention between MNs in their respective clusters. Also in the last section we present the implementation of algorithms developed in chapter 4 in OPNET and results for performance improvements by these new algorithms.

In appendix A, we have included details of our simulation environment including all the OPNET node and process models used in this experiment, along with a complete explanation of all the changes done in the source code of OPNET files being used. We have used different OPNET modules including WLAN and MANET modules in order to implement our designed WMN. Within each module there are several node and process models as well as other supporting C and header files. OPNET uses these files to implement various components of the related networks including nodes, packets, and protocols and so on. In order to make any changes in these components, or implement new algorithms, some of these files and their associated components should be manipulated and the entire code should be re-compiled. Then new simulations based on the changed files could be saved as new project, and the new simulation could be run which reflects the new changes and algorithms.

The important point taken from appendix A is that our implementation runs between several modules including WLAN and MANET modules. Source code for

several files from each module is changed. We have worked on delay and throughput performance measurements. Algorithms 4.3 and 4.4 are implemented in MANET module since all the files related to ad-hoc networks; including AODV files are located in this module. However performance measurements are performed in WLAN module. OPNET needs some major changes in the source code in order to make connection between the two modules. In some cases making these connections are out of the scope of this thesis. In such cases we have made measurements in one module, and have made reasonable estimations and manually entered those values in the other module. All the details about these changes are presented in appendix A.

5.1 Modeling of Wireless Mesh Network in OPNET

In this section we will explain the Wireless Mesh Network (WMN) model created using AODV and MANET modules in OPNET. WMN is deployed based on figure 3.1 in chapter 3. First we will show the basic model created in OPNET following some results collected. Then we will show the modified models based on the changes made in the source code of OPNET and the results from the new models.

We simulated a small WMN in OPNET modeler 11.5 PL1 by creating a mesh of wireless routers in the backbone and clusters of mobile nodes attached to each wireless router. We used 802.11a for backbone, and 802.11b/g for access network.

5.1.1 Modeling basic WMN network in OPNET

A campus network is deployed over a square geographical area of range 10000×10000 m as shown in the figure 5.1.

WMN is deployed using eight wireless mesh routers (WMRs) creating WMN backbone, comprising of two rows of routers. The lower set of routers includes Access Mesh Routers (AMRs) which connect Mobile Nodes (MNs) to the backbone. The upper set of routers are middle WMRs that participate in the backbone, however

do not have any MNs connecting to them for access purposes. First AMR in the lower row is named S_AMR, which depicts AMR corresponding to the S_MN cluster, and the last AMR is D_AMR which shows the AMR corresponding to the D_MN cluster.

Each AMR is surrounded by MNs comprising a cluster for the corresponding AMR. For each cluster, we start the simulation with one MN, and then increase number of MNs to see the effect of increased traffic and channel contention. All the nodes in the network are configured to work under ad hoc mode. At this stage MNs are fixed. In future work we are planning to include mobility among the access network.

Figure 5.1: Basic WMN scenario

Each AMR is equipped with two interfaces; one for the backbone running 802.11a and the other for access network running 802.11b/g according to figure 3.4 in chapter 3. At this stage, our MNs have a single interface running 802.11b/g to connect to

both AMR and ad-hoc network. The assumption is that MNs use same interface and same radio frequency to connect to both AMR and other MNs. This assumption is reasonable considering that all nodes are in the ad-hoc mode and capable of connecting to more than one peer at the same time. Since the backbone is on 802.11a, bb traffic would not interfere with MN-MN and MN-AMR traffic. Different cases for interference were discussed in chapter 3.4.

5.1.2 AMR and WMR Configuration

Various configuration parameters were set for each AMR as well as WMRs as shown in figure 5.2.

Figure 5.2: Configuration parameters for AMR and WMR.

Configuration parameters for WMRs include name, model, ad-hoc protocol parameters, IP parameters including interface parameters and so on. Figure 5.2 shows that each AMR and WMR is configured with two interfaces (IF0 and IF1). Each interface is on a different subnet. One interface is used for backbone, and the other for

access network. Ad-hoc routing protocol is activated on the router, and AODV is selected. AODV is also activated on the interface separately. This is required because, other non-ad-hoc protocols are also possible on these interfaces.

5.1.3 Mobile Node (MN) Configuration

Various configuration parameters were set for each MN as shown in figure 5.3.

Figure 5.3: Configuration parameters for Mobile Node (MN)

Configuration parameters for MNs include name, model, ad-hoc protocol parameters, IP parameters including interface parameters and so on. Figure 5.3 shows that MN has also AODV set as ad-hoc protocol.

5.1.4 Generating MANET traffic

MANET traffic is generated from a source MN (S_MN) to a destination MN (D_MN) according to the specifications values in table 5.1. Traffic is first generated from

contending MNs in the cluster to go to the S_AMR. Then after 100 second when the traffic is continuously generated and contention is stabilized, then S_MN starts sending traffic to S_AMR. At this point this new traffic is affected by the contention from other MNs.

Traffic Parameter	Value
Start time	0 (100 sec for contending MNs)
Inter-arrival time	0.01 (0 not acceptable in OPNET)
Packet size	8192 bits
Destination	Destination MN (D_MN)
Stop time	End of simulation

Table 5.1: values for Traffic parameters generated from S_MN to D_MN

The source MN sends MANET traffic at exponential inter-arrival time of 0.01 second, and the constant packet sizes are 8,192 bits for the destination MN and 16,384 bits for the AMR. We set the throughput threshold at a minimum value of 100 bits/sec in order for the second route discovery to be triggered.

In various experiment scenarios, in order to avoid unnecessary traffic which might cause generating confusing results, we are only generating traffic from S_MN to D_MN, and/or from MNs to S_AMR. In most cases, generated traffics are traveling from MNs to D_MN or to S_AMR via the backbone. In cases that traffic is going through the ah_path, this is the only traffic that is generated in the ad-hoc network.

5.2 Simulation objectives

In this section we show how and why we chose specific simulation scenarios. Each scenario then is presented in detail in section 5.3. In section 5.3.1 we show the contention problem. Then in section 5.3.2 we show how the MN looks for a solution by acquiring an alternative path through ad-hoc access network, and as a result it helps in improving performance by showing improvements in end-to-end delay and

throughput. Then in section 5.3.3 we show the result of simulation with the new algorithms implemented in the OPNET model, and show how MN will use these algorithms to search for the alternative path, and how it can use the decision making algorithm to select the better path of the two `bb_path` or `ah_path`. Each sub-section in section 5.2 corresponds to a scenario presented in section 5.3.

In chapter 3 we presented our original solution where MNs will use two interfaces to communicate through `bb_path` and `ah_path`. However due to complexity and cost of multiple interfaces, we proposed an alternative solution where the MN could also use one interface for both types of communications, and we choose this alternative solution for implementation. In our implementation we also use a single channel for both communications due to the fact that changing the channel might cause the traffic to drop while the MN is trying to switch from backbone to ad-hoc traffic.

Throughput calculation for `bb_path` is based on reference [31] throughput calculations and for `ah_path` is based on reference [20] as described in chapters 3 and 4. However for the purpose of implementation we decided to use the throughput measurements provided by OPNET for the `ah_path`. This is due to the fact that considering the time constraints and complexity of the implementation of method in reference [20], we believe that OPNET provided implementation is sufficient for our purposes at this point.

OPNET provides throughput measurements for 802.11 interfaces in WLAN module. We used this throughput calculation in our design, and also implemented the Bianchi saturation throughput calculation in the WLAN module. However our decision making process is implemented in MANET module, since the MANET module includes all the ad-hoc related processes and node models.

Throughout the results, we use throughput as the primary source of performance measurements, and also include end-to-end delay, as a secondary performance measurement to backup the throughput results.

5.2.1 Problem statement: Contention in WMN

In this scenario we will present a simulation to show that increasing number of nodes from 1 to 4 in a cluster will increase the traffic and contention for the channel-use and consequently will drop the performance measures such as throughput and end-to-end delay. In this part we consider cases for which traffic between source and destination uses the backbone route (bb_path). Then we increase the number of MNs in the source cluster, which will result in increased contention for the channel, and consequently the end-to-end delay will increase and throughput will drop.

5.2.2 Solution: MN finds Alternative ah_path, improves Throughput

In this simulation scenario, we allow S_MN to explore other potential alternative paths in order to avoid the contention with other MNs in the cluster, and send traffic via ah_path to the destination. This will allow other nodes to use the channel more freely which consequently drops contention and improves the performance by decreasing delay and increasing available throughput.

In this scenario, we show by simulation that if S_MN finds an alternative path via ad-hoc access network, it will help in reducing contention and consequently improve the end-to-end delay and throughput performances. In such cases where bb_path is experiencing difficulty, the backup ah_path shows improvement in performance measurements.

5.2.3 New algorithms for MN to find the alternative path and use it if necessary

In this scenario we show the modifications that we have done to AODV source code in order to modify Route Table, RREQ and RREP packets to reflect the new algorithms in the source code of MANET and WLAN modules of OPNET. We also show the how algorithms 1-4 are implemented in the source code. This scenario is divided into three sub-scenarios. First scenario is when the source cluster has only 2

MNs, and S_MN takes bb_path to send traffic to D_MN. In the second scenario, source cluster has 4 MNs, and S_MN still prefers the backbone path. However, from the throughput graph, it is clear that the throughput has dropped due to contention. In the third scenario, we increase the number of nodes in the source cluster, consequently the throughput will fall, and S_MN will compare the throughput collected from bb_path and ah_path and after comparison it will decide to take the ah_path as the backup path as a result of drop in the throughput value.

5.2.4 Bianchi's saturation throughput

In this scenario we will show the Bianchi Equation implemented in OPNET to calculate the saturation throughput of a cluster in its AMR. In section 5.3.3.1 we will show different scenarios by increasing number of nodes in a cluster, and calculate corresponding throughputs and finding the saturation throughput with this cluster.

5.3 Simulations Results

Figure 5.4 shows the final stage for WMN scenario. Originally we started with only one MN in the source cluster (S_MN) sending traffic to destination MN (D_MN), then we increased the number of MNs in the source cluster from 1 to 4 for the basic case, and from 1 to 6 for the new altered project that includes the new source code. In each case, we investigated the performance of the network.

In each case for each scenario, we measure end-to-end delay and throughput. Then we compare those values for all the scenarios together in order to demonstrate the effect of changes in the traffic on the performance of the network.

Figure 5.4: WMN scenario including the new source code,
Highlighting S_MN, S_AMR, D_AMR, and D_MN

5.3.1 Channel contention in Backbone Path

Figure 5.5 (A) shows the throughput values for the `bb_path` under three different channel contention situations. The throughput is measured at the destination node. Increasing the number of MNs in a cluster shows significant drop in throughput resulted from contention between MNs. We increase channel contention by increasing the number of MNs in the source cluster from 1 MN to 4 MNs. Figure 5.5(A) shows that the throughput at the destination MN decreases while the number of MNs in the source cluster increases. This indicates that as the number of nodes increases, the channel contention increases and throughput falls, and consequently packet drop rate increases. This is also verified by measuring the number of retransmissions in the source clusters, which also increases with the decrease in throughput (Figure 5.5C). Figure 5.5 illustrates the situation when high contention in AMR cluster renders the `an_path` to be the bottleneck of the `bb_path`, and consequently results in various performance degradations.

Figure 5.5: increasing number of MNs from 1 to 4 in a cluster result in increasing contention
(A) Throughput at the destination (B) End-to-end Delay (C) Retransmission attempts (page53)

(A)

(B)

Figure 5.5 (B) shows end-to-end delay for the same scenario. Figure 5.5 shows that when number of nodes is increased from 1 to 4 MNs, the value of throughput is dropped from 600 Kbps to below 200Kbps due to contention. At the same time the

(C)

Figure 5.5: increasing number of MNs from 1 to 4 in a cluster result in increasing contention
(C) Retransmission attempts

end-to-end delay is increased from 0.1 sec to over 0.4 sec. Figure 5.5(C) shows that number packet drops will double from 150 packet to about 300 packets as we increase number of MNs from 2 to 4.

5.3.2 Alternative Ad_Hoc Path as a solution for Contention

Figure 5.6 (A) shows a significant improvement in throughput performance when S_MN uses the ah_path versus bb_path. Throughput for the D_MN is improved for a cluster of 4 MNs, when ah_path is used, to alleviate for the difficulty caused by the contention from MNs in the source cluster. Figure 5.6 (B) show the same results for end-to-end delay in the same scenario.

Figure 5.6 shows that when performance degrades due to contention, S_MN could switch to ah_path. Should S_MN decide to take this path, figure 5.6 shows that throughput performance will improve from 200 Kbps to over 800 Kbps, and the corresponding end-to-end delay drops from 0.4 seconds to below 0.2 seconds.

In sections 5.3.1 and 5.3.2, we assumed that the performance measures such as throughput and delay are readily available to the AMRs and MNs, and we manually switch from `bb_path` to `ah_path`. In the next section we will allow the `S_MN` to investigate the performance measurements and to acquire throughput values automatically based on the implementation of algorithms 1 and 2, and based on these values evaluate the goodness of both paths and decide which path is the best to be used.

(A)

(B)

Figure 5.6: Throughput at the destination (A) and end-to-end delay (B) comparing `bb_path` versus `ah_path` for the case of high contention in the source cluster

5.3.3 Decision Making Process

In this section we will allow the MNs to make their own decision and choose the best path to take based on the performance evaluation of each path. The decision making process is based on the saturation throughput calculated based on Bianchi's Equation. Therefore, in section 5.3.3.1 we present simulation scenario for Bianchi's saturation throughput first, then in the next section (5.3.3.2) we continue with simulation scenarios presenting decision making process based on algorithms 4.3 and 4.4.

5.3.3.1 Bianchi's Saturation Throughput

Saturation throughput of each AMR is calculated based on Bianchi's Saturation Throughput Equation as indicated in algorithm 4.1. This algorithm was implemented in the file `ae_wlan_mac`, compiled and separate scenarios have been built to illustrate this throughput. Then number of MNs around AMR is increased from 1 to 6 gradually, and increased throughput is measured. The final stage of this scenario is presented in figure 5.7 including 6 MNs in the cluster surrounding AMR.

Based on Bianchi's estimated saturation throughput, as we increase number of MNs around an AMR, the throughput shall increase up to a certain point where it stops. The throughput will not increase beyond this point. This is called "Saturation throughput". In our case, the number of MNs was increased incrementally, and in each case the throughput is measured. All the throughput values for all the cases are presented in the same graph in figure 5.8.

Figure 5.8 shows that throughput increases as the number of nodes increases from 1 to 4. Then the throughput reaches saturation at around 5.6 Mbps, and stays there. Beyond this point adding more MNs (5 and 6) will not increase the throughput for overall system. Figure 5.9 shows the same results as 5.8, only in cumulative sum format for clarity.

Figure 5.7: WMN final scenario including 6 MNs surrounding AMR for testing Bianchi's Saturation Throughput

Figure 5.8: Throughput values for saturation throughput of a cluster calculated in the clusterhead AMR based on Bianchi's saturation throughput implemented in algorithm 4.1

Bianchi further explains that throughput eventually increases to an amount higher than the saturation value, and then drops lower to stabilize at saturation. This is clearly observed in case of 6 MNs in figure 5.8 where throughput has increased to around 6 Mbps and then drops to the saturation value which is around 5.6Mbps.

Figure 5.9: Sample Sum or cumulative sum results
for throughput values presented in figure 5.8

5.3.3.2 Simulation scenarios for decision-making and path selection processes

This part of the simulation is based on the changes in the core of AODV source code in OPNET presented in appendix A. New AODV_enabled nodes should be aware of the throughput values for each path. Each AMR calculates its cluster saturation throughput. This throughput is saved as own_throughput. Each AMR also measures its link throughput to the next hop or next AMR.

Based on the current implementation, S_MN will broadcast the RREQ. S_AMR will receive this RREQ and uses regular AODV to forward the RREQ hop-by-hop to the D_MN. D_MN will reply by uni-casting a RREP message back to D_AMR and includes its throughput (this is the down-link throughput of the link from D_AMR to D_MN) in the RREP. This is a one-way throughput of D_AMR to D_MN, and not the throughput for the reverse path. D_MN also sets ROUTE_TYPE to AH. D_AMR will receive this RREP, compares RREP_THROUGHPUT (This is recorded as INTERMEDIATE_THROUGHPUT) to OWN_THROUGHPUT (This is the Bianchi saturation throughput calculated for destination cluster), and updates the RREP_THROUGHPUT with the smaller value. Every WMR along the way takes this throughput and compares it with its own link throughput, and updates the RREP with the smaller value. Since the throughput provided by the backbone links are usually higher than any access network throughput, the original link throughput coming from D_MN which represents the throughput of sub_path3 is likely smaller than any backbone throughput and will be selected as path throughput thus far of mesh_path. Therefore this throughput will have to compete with the throughput of sub_path1 and the smaller value of these two will get elected as the throughput for the route. At the same time D_AMR will also change the ROUTE_TYPE to BB, and after this point it will stay as ROUTE_TYPE_BB and will not change again.

On the other hand, the RREQ will also go through the AH path to the next MN, and uses regular AODV to reach D_MN hop-by-hop. In case of two interface solution, this RREQ would go through the ah_int, however in the current implementation, since we are using the same interface for both ah_path and bb_path, one at a time, we forward the second RREQ on the same interface of the MN. So D_MN will have a second RREQ from ah_path. D_MN will also send the same RREP through ah_path, and a procedure similar to the one in the bb_path will be repeated, except that ROUTE_TYPE will always stay at ROUTE_TYPE_AH for this path. The throughput added to RREP on the ad-hoc path is the link throughput between the D_MN and the next hop (neighboring MN). Each MN along the way will compare this throughput with its own link throughput to the next MN and update RREP accordingly.

At this point S_MN will have two routes BB and AH with each having its own throughput. S_MN will compare these two throughput values and uses the equation in algorithm 4.4 to decide which path to take.

All these processes have been implemented by changing the AODV source codes in files `ae_aodv_rte`, `ae_aodv_route_table`, `ae_aodv_packet_support`, `ae_wlan_mac`, `ae_wlan_dispatcher`, and all of their associated files and header files.

Changes related to throughput of BB paths are reflected in the file `ae_wlan_mac` and for AH paths are in `ae_wlan_dispatch` file. However, due to the fact that these files are located in WLAN module in OPNET which is completely separated from the MANET module, for the purpose of this experiment, values of throughput are measured separately, and estimated values corresponding to those are used for the decision making process.

The AODV routing tables include two new columns for `ROUTE_THROUGHPUT`, and `ROUTE_TYPE`. The value of `ROUTE_THROUGHPUT` comes from `wlan_mac` file where BB throughput calculation is taking place for AMR, or from `wlan_dispatch` file where AH throughput is calculated for MN, depending on whether the node is an AMR or MN respectively. The value of `ROUTE_TYPE` is a Boolean `ROUTE_TYPE_BB` or `ROUTE_TYPE_AH` depending on whether the RREP is coming from an AMR or a MN. This is determined by extracting last digit of the IP address of the source in the RREP. All the AMRs are clusterheads, and their IP addresses are statically set to X.X.X.1, therefore if the last digit of the IP address is 1, then the source is an AMR and the `ROUTE_TYPE` is set to `ROUTE_TYPE_BB`, otherwise it is set to `ROUTE_TYPE_AH`.

Changes related to decision making processes are made in the files `ae_aodv_rte`, `ae_aodv_request_table.ex.c` and `ae_aodv_route_table.ex.c` to include values for throughput and route type.

RREP packet also includes values for throughput and route type. Changes related to RREP are made in the file `ae_aodv_pkt_support.c` to include values for throughput and route type.

After making all these changes and implementing the algorithms in the proper files as specified in appendix A, OPNET files were compiled and simulations were run with the new codes. Results are shown in figures 5.10 through 5.16.

Scenario in figure 5.1 is duplicated, the entire source code for the files presented in appendix A are replaced by the new files and compiled. The newly created project is run and results collected. Figure 5.10 shows the new scenario with only 2 MNs in the source cluster.

Figure 5.10: WMN scenario with 2 MNs including the new source code, indicating `bb_path` taken by the traffic sent from `S_MN` to `D_MN`

Figure 5.10 shows that in presence of only 2 MNs in the source cluster, where throughput provided by the `mesh_path` is fairly high, the `S_MN` selects the `mesh_path` and forwards the traffic via backbone.

By increasing number of MNs in the source cluster from 2 to 4, we observe that S_MN still chooses the mesh_path as depicted in figure 5.11. Figure 5.11 shows that in presence of 4 MNs in the source cluster, S_MN still favors mesh_path. At this point we investigate the throughput values for these two scenarios. Figure 5.12 shows the throughput values of scenarios in figure 5.10 and 5.11.

Figure 5.11: WMN scenario with 4 MNs including the new source code, indicating bb_path taken by the traffic sent from S_MN to D_MN

We observe from figure 5.12 that the throughput value at the destination will drop as the number of nodes in the source cluster increases. As we continue adding more MNs at the source cluster, the throughput will keep decreasing until it does not meet the requirement set by algorithm 4.4. At this point it will switch to ah_path and traffic between S_MN and D_MN is taking the ad-hoc path. We have created one scenario with 6 MNs in the source cluster in which the ah_path is selected by the S_MN. Figure 5.13 shows the scenario with 6 MNs in the source cluster.

Figure 5.13 shows that S_MN favors the ah_path due to the fact that throughput performance is decreased below the minimum requirement set by the algorithm 4.4. Figure 5.14 shows the throughput performance for the case where S_MN chooses ah_path over bb_path in presence of 6MNs.

Figure 5.12: Throughput values for 2 scenarios in figures 5.10 and 5.11

Figure 5.13: WMN scenario including the new source code with 6 MNs in the source cluster,
Indicating ah_path taken by the traffic sent from S_MN to D_MN

Figure 5.14 shows that as the number of MNs in the source cluster increases to 6, the throughput decreases initially to a level lower than that of 4 MNs to about 50

Kbps. This indicates that the traffic in presence of 6 MNs still uses the `bb_path` (marked as area 1). Eventually `S_MN` will switch from `bb_path` to `ah_path` and forwards traffic through the ad-hoc network. This is due to throughput performance evaluation of `mesh_path` compared to `ah_path` by `S_MN`. We could see from the figure that the throughput in presence of 6 MNs has increased in comparison to selecting `mesh_path` with 4 MNs. Initially there is drop in throughput until about 130 seconds (marked as Area 1 in the graph). This is due to the fact that initially next hop node for `S_MN` is still `S_AMR`, and `S_MN` still sends traffic via backbone. At this point there are still 6 MNs contending for the channel (contention level is 6).

Figure 5.14: Throughput values for scenario in figure 5.13 with 6 MNs where `S_MN` will switch from `bb_path` to `ah_path`,

After 130 seconds, we observe improvement in throughput. At this point traffic is switched and starts traversing via the `ah_path` and consequently the throughput will take a jump to above that of 2 MNs until around 200 second. This hike is due to the initial surge when switching to the `ah_path` takes place (marked as Area 2 on the graph).

After the initial switch surge, then throughput will drop to a stabilizing point which sits between throughput of 2 MNs and 4 MNs scenarios and continues at a

steady rate beyond this point (marked as Area 3 on the graph). Due to the fact that the number of MNs in the cluster remains the same, it was expected to experience similar contention levels; however the improvement in the throughput indicates that in case of `ah_path` fewer numbers of MNs are contending. In this scenario, it is observed that next hop for `S_MN` is `MN_A2`, and `MN_A4` and `MN_A6` are not visible from point of view of `S_MN`. This could be due to hidden-terminal issue, and indicates that contention level is 3, which means there are only 3 MNs contending for the channel. This is consistent with the throughput level in figure 5.14 which sits between 2MN and 4MN scenarios.

A quick summary of throughput scenarios presented thus far; first, simulation was done with minimal number of MNs in source cluster. Then number of MNs is increased and consequently traffic generated and contention is increased which results in throughput drop. At this point `S_MN` will initiate a second route discovery, finds `ah_path`, and compares the throughput values. In cases where it finds throughput performance better in the `bb_path`, it will select the `bb_path`. Otherwise in case the `ah_path` provides a better throughput, it will switch to `ah_path`.

To further support the results of throughput analysis we look at delay measurements as a secondary source of performance measurement. We use MANET module end-to-end delay back up the results based on WLAN throughput values. The delay measurements for the same scenarios are presented in the figures 5.15 and 5.16.

Figure 5.15 shows that as the number of MNs in the source cluster increases, the end-to-end delay also increases for 4 MNs versus 2 MNs. Then as we increase number of MNs even further to 6 MNs, then the `S_MN` will start the new algorithm, finds the `ah_path` and starts using this new path. Consequently the delay measurements will improve. Figure 5.16 shows that for the scenario with 6 MNs, in which case the `ah_path` is being used, the delay performance measurement has improved. These results are consistent with the results from throughput analysis presented in section 5.3.3.2.

Figure 5.15: MANET delay values for 2 scenarios where number of MNs in source cluster is increased from 2 to 4

Figure 5.16: MANET delay values for scenario with 6 MNs and ah_path is selected

Chapter 6: Conclusions and Future Works

6.1 Conclusions

We proposed an integrated routing system for WMN that exploits both paths through the backbone network and through the ad-hoc access network of mobile nodes. The ad-hoc path is considered as an alternative path and can be used under the following situations: (a) when the primary backbone path is severely constrained due to access links contention, (b) during handover to hide losses due to `bb_path` outage, and (c) for shortest path routing when `bb_path` is longer than `ah_path`. We also proposed a scheme for initiating the route discovery of the ad-hoc path. We have proposed two algorithms by which the mobile nodes could evaluate the conditions of both backbone and ad-hoc paths based on performance measurements of throughput offered by each path, considering such measurements are made available to the MN. We have also analyzed the two algorithms given by references [20] and [31]. Mobile node could use these algorithms to calculate the throughput for backbone and ad-hoc paths.

We have simulated a simple WMN, and investigated the performance of the backbone and ad-hoc paths as the traffic increases in the network. We also investigated the access contention situation and demonstrated the benefit of alternative ad-hoc path, and showed that as the performance degrades, the MN could make a decision to switch between the paths, by which it could improve the performance measurements such as end-to-end delay and throughput. We have also simulated saturation throughput measurement for a cluster of MNs at the clusterhead wireless mesh router.

6.2 Future Works

There are several areas that could be investigated and improved in future works. Some areas involve introducing major new components and/or fundamental changes in the core of our integrated routing systems. Other areas, however look closer and more practical improvements that involve minor changes.

Short-term improvements include expanding this solution to multi-hop ad-hoc access network within clusters. We are also considering using dynamic throughput calculation and reporting between AMRs and MNs. The major obstacle to achieve this goal was the cooperation between Reactive AODV in the ad-hoc network and Proactive OSPF in the backbone networks. Should we be able to overcome this obstacle, we could incorporate performance measurements such as throughput and delay in the routing table of the proactive routing protocol in the backbone, which will be readily available to RREQ coming from MNs. Other short term future work that is highly attractive is to enable MNs to use two separate interfaces for ad-hoc and backbone networks.

Long-term improvements include incorporating link quality metrics (e.g. ETX) in the backbone routing.. We want to incorporate QoS metrics in the decision of using `ah_path`. We are evaluating the use of RTT for determining `ah_path` throughput and ETX for the backbone throughput.

Appendix A

Simulation Environment and OPNET Model files and source codes

OPNET Simulation Environment

In this section we detail the architecture of OPNET simulator and how we used MANET model in OPNET to simulate Alternative Ad-Hoc Path for Wireless Mesh Networks.

OPNET Architecture

OPNET provides a comprehensive simulation environment for modeling of various communication networks and their performance evaluation. OPNET modeler also provides the entire source code for all the tools, which enables users to develop their own accessories and added sections or even an entire model. The package consists of a number of tools, each one focusing on particular aspects of the model. These tools fall into three major categories that correspond to the three phases of modeling and simulation projects. These three phases are Specification, Simulation and Analysis.

As illustrated in figure A.1, these phases work in a circle which allows addition of new specifications as new requirements arise.

Figure A.1: Three phases of OPNET simulation model

MANET Model Architecture in OPNET

For this thesis we used OPNET Modeler 11.5 which includes various MANET models for OLSR, AODV, DSR and TORA. In this section we will explain the model architecture, node models of MANET and all source codes including header and external files that are used by MANET model and AODV processes.

Node Models in MANET

MANET model includes several MANET nodes and configuration tools. All MANET capable nodes are included in the MANET object palette as shown in Figure A.2.

Figure A.2: MANET model Architecture

There are several MANET node objects available in MANET models. Complete list of these objects is presented in figure A.3 and listed below.

Figure A.3: MANET Node Palette

Application Config: Applications can be generated using different definitions. Each application definition can specify usage parameters such as start time, duration and repeatability. Then each application could be used by nodes directly or through pre-made profiles using profile config.

MANET stations: These nodes can be used to generate raw packets on IP over WLAN. They can configure as a traffic source or destination and can be configured to run MANET routing protocols.

Mobility Config: These tool can be used to define mobility profiles that each node references to model mobility. This node controls the movement of nodes based on the configured parameters.

Profile Config: Profiles can be configured to describe activity patterns of a user or group in terms of the applications used over a period of time. We could configure several profiles running on a given network or node. These profiles can represent different groups, such as throughput profile to depict performance evaluation through throughput analysis.

Rxgroup Config: The receiver group configuration node is used to compute the set of possible receivers that a node can communicate with. This utility node can speed up a simulation by eliminating receivers that do not match.

Task Config: Custom applications could be configured using task configuration tool.

WLAN routers: These nodes can be used as wireless routers to route traffic. They are equipped with two interfaces; both wired and wireless interfaces. They are capable of running various traditional and new routing protocols such as OSPF, RIP, AODV, DSR and so on.

Wireless LAN workstations and servers: These nodes can be used to generate application traffic such as FTP, E-mail, HTTP on TCP over IP. These nodes can also be configured to run MANET routing protocols.

MANET node model architecture

MANET node could be a mobile node (MN), a WLAN workstation, or a Wireless Mesh Router (WMR) operating in ad-hoc mode. There are several process models in MANET . some of these models deal with MANET routing protocols such as ip_dispatch, manet_dispatch, manet_rte_mrg, dsr_rte and olsr_rte process models (figure A.4).

Figure A.4: MANET process models in OPNET

The function of ip_dispatch of the ip_encap process is to create a manet_mgr (as a child process as in figure A.2) that manages all ad-hoc routing protocols in OPNET except OLSR which is processed by manet_rte_mrg process model directly. manet_mgr also creates several other processes for the required ad-hoc routing protocol as defined in the configuration parameters. If the selected ad-hoc routing protocol is AODV, then the aadv child process of manet_mgr is called.

AODV Model in OPNET

We used AODV as the main choice for routing in MANET, both for `ah_path` and for the part of `bb_path` between MN and WMRs. In the following 3 sections we detail configuring AODV in OPNET and how to change the source code and external files related to AODV in order to incorporate the new algorithms into the existing AODV models.

AODV is presented in `aodv_rte` process model as a child process of `manet_mgr`. AODV process model is presented in figure A.5.

Figure A.5: AODV process model

Configuring AODV in OPNET

By right clicking on any AODV-enabled node in the project editor, a new window will open to edit attribute values for different parameters. Figure A.6 shows how to configure AODV parameters as defined in RFC3561 in OPNET for a mobile node or any other MANET node.

Figure A.6: Editing attributes and selecting AODV for Mobile Node (MN)

For a node with interfaces that could enable other non-MANET protocols such as OSPF or RIP, MANET protocols such as AODV should also be enabled on the interface, as shown in figure A.7 for a WMR.

Figure A.7: Editing attributes and selecting AODV for Wireless Mesh Router (WMR)

AODV Model Files

There are three types of files that enable MANET and AODV to work in OPNET; **Process model files** are used to generate and process MANET control packets and maintain routing tables and update IP common routing table.

External Source Files: These files are written in the C-code and are extensions to help the process models, available in the <opnet_dir>/std/Manet directory.

Header files are the header files and libraries which define the types and formats and functions used by the C files and process models, and are available in the <opnet_dir>/std/include directory.

Main process models involving MANET and AODV are:

- manet_rte_mgr
- ip_dispatch
- aodv_rte

However, these models refer to many files including header files and external source files that enable MANET protocols to work. All these files are located under the root folder of MANET (in <opnet_dir>/) in the following locations:

- **C:\Program Files\OPNET\11.5.A\models\std\include:**
 - **manet.h:** Data Structures used by all MANET routing protocols
 - **aodv.h:** Data Structures used AODV in MANET
 - **aodv_pkt_support.h:** Data Structures used for the packet fields used in the AODV RREQ, RREP and RRR packet
 - **aodv_ptypes.h :** Prototypes used by the AODV
- **C:\Program Files\OPNET\11.5.A\models\std\manet:**
 - **aodv_packet_queue.ex.c:** C file for AODV Packet Queue APIs
 - **aodv_pkt_support.ex.c:** C file for AODV Packet Support APIs
 - **aodv_request_table.ex.c:** C file for AODV Request Table APIs
 - **aodv_route_table.ex.c:** C file for AODV Route Table APIs
 - **aodv_support.ex.c:** C support file for AODV support functions
 - **manet_support.ex.c:** C file for MANET APIs to interface to IP

Function of the AODV header files:

aodv.h defines constants, data structure for route, request and connectivity tables,

aodv_pkt_support.h file defines types of packets and structure of packet options like rreq, rrep, rerr, and aodv_

aodv_ptypes.h files are function prototypes for external files

Any implementation of new algorithms or changes to the current OPNET files should be reflected in these files. Then the new files should be compiled, and the new simulations should be run using the new files.

All these files are used by the server, so if any user is using MANET model on this server his/her work will be tied up to these files. For this reason, any changes in the

source code of these files should be saved with a new name in a local directory. So the local directory is created in the location:

C:\Documents and Settings\<user>\op_models

And all the necessary files are saved and run locally so that other server users are not affected by the changes. OPNET will first check the local op_models directory to find any files needed by default. Should it not find the files on the local directory, then it will revert back to the root OPNET directory.

In this case we have implemented four algorithms (algorithms 1-4 in chapter 4) in OPNET. Algorithms 1 and 2 are for throughput calculations of backbone and ad-hoc paths respectively. Algorithm 3 is for MN to be able to initiate a second route discovery and find a second alternative ad-hoc path through the access network and algorithm 4 is for the MN to be able to make a decision to which path to use. Algorithm 1 is implemented in the file manet_dispatch. This is because OPNET uses this file to measure performance parameters for MANET nodes. Algorithm 2 is implemented in the file wlan_mac. This is because OPNET uses this file to calculate the throughput value for any wireless node, and any changes to the way throughput calculation is done should be reflected in this file. Finally algorithms 3 and 4 are implemented in the file aodv_rte since this file is where all the routing table, RREQ and RREP files are kept in OPNET.

Three files “manet_dispatch”, “wlan_mac”, “aodv_rte” are saved with new names in the local OPNET directory as “ae_manet_dispatch”, “ae_wlan_mac”, “ae_aodv_rte”. Also the node models “wlan_router” and “manet_station” are saved with new names “ae_wlan_router” and “ae_manet_station”.

There are several other C files and header files that are associated with these files that need to be changes in order for these node and process models to function properly. List of all the new files changed and added to OPNET local directory is presented in table 5.1 below.

No	file	locattion
1	ae_aodv.h	C:/<user>/op_models
2	ae_aodv_pkt_support.h	
3	ae_aodv_ptypes.h	C:/<user>/op_models
4	ae_aodv_route_table.c	C:/<user>/op_models
5	ae_aodv_rte.c	C:/<user>/op_models
6	ae_aodv_support.c	C:/<user>/op_models
7	ae_manet_mgr.c	C:/<user>/op_models
8	ae_manet_rte_mgr.c	C:/<user>/op_models
9	ae_wlan_dispatch.c	C:/<user>/op_models
10	ae_wlan_mac.c	C:/<user>/op_models
11	ae_ip_dispatch.c	C:/<user>/op_models
12	ae_aodv_suport.c	C:/<user>/op_models
13	ae_wlan_support.c	C:/<user>/op_models
14	ae_manet_station_adv.nd	C:/<user>/op_models
15	ae_wlan2_router_adv.nd	C:/<user>/op_models

Table A.1: List all OPNET source code files changed for WMN model

Taking results of AODV

By selecting DES from the tool bar on the project editor and selecting the “choose individual DES” (Discrete Event Simulation) option, we could choose different statistics to be simulated. Figure A.8 illustrates how to choose the statistics from OPNET project editor.

Figure A.8: Choose statistics to be included in the simulation from DES menu

References

- [1] Mesh Networking Summit 2004: Making Meshes Real, June 23-24, 2004 Snoqualmie, Washington
<http://research.microsoft.com/meshsummit/>
- [2] Mesh Networking Academic Resource Toolkit 2005, <http://research.microsoft.com/netres/kit/>
- [3] D. Aguayo, J. Bicket, S. Biswas, G. Judd, R. Morris; "Link-level Measurements from an 802.11b Mesh Network", M.I.T. Computer Science and Artificial Intelligence Laboratory, 2004
- [4] W. Liao, Y. Tseng, J. Sheu; "GRID: A Fully Location-Aware Routing Protocol for Mobile Ad Hoc Networks, Telecommunication Systems, 2001
- [5] Tzi-cker Chiueh, Ashish Raniwala, Rupa Krishnan, Kartik Gopalan
"Hyacinth: An IEEE 802.11-based Multi-channel Wireless Mesh Network":
<http://www.ecsl.cs.sunysb.edu/multichannel/>
- [6] F. Akyildiz, I. Wang, and X. Wang, Wireless Mesh Networks: A Survey, Computer Networks Journal (Elsevier), March 2005.
- [7] R. Bruno, M. Conti, E. Gregori; "Mesh networks: commodity multihop ad hoc networks".
Communications Magazine, IEEE, March 2005, Volume: 43, Issue: 3
On page(s): 123- 131
- [8] E.M. Royer and C-K Toh, "A review of current routing protocols for MANET", IEEE Personal Communications, vol. 6 No 2, pp. 46-55. April 1999.
- [9] S. Murthy and J.J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks", ACM Mobile Networks and App. J., Special Issue on Routing in Mobile Communication Networks, Oct. 1996, pp. 183-97. <http://www.ics.uci.edu/~atm/adhoc/paper-collection/aceves-routing-winet.pdf>
- [10] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In Proc. ACM/IEEE MobiCom, pages 85–97, October 1998.
- [11] DSDV (Highly Dynamic Destination-Sequenced Distance Vector routing protocol) - C. E. PERKINS, P. BHAGWAT Highly Dynamic Destination-Sequenced Distance Vector (DTDV) for Mobile Computers Proc. of the SIGCOMM 1994 Conference on Communications Architectures, Protocols and Applications, Aug 1994, pp 234-244.
- [12] OLSR (Optimized Link State Routing Protocol) - PHILIPPE JACQUET, PAUL MUHLETHALER, AMIR QAYYUM, ANIS LAOUITI, LAURENT VIENNOT, THOMAS CLAUSEN Optimized Link State Routing Protocol (OLSR), RFC 3626. <http://www.olsr.net/>, <http://www.olsr.org/>
- [13] Ad-hoc On-demand Distance Vector - C. PERKINS, E. ROYER AND S. DAS Ad hoc On-demand Distance Vector (AODV) Routing, RFC 3561
- [14] Dynamic Source Routing - DAVID JOHNSON, DAVID MALTZ, YIH-CHUN HU: The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks, Internet Draft, draft-ietf-manet-dsr-10.txt, work in progress, July 2004 / DAVID B. JOHNSON, DAVID A. MALTZ: Dynamic Source Routing in Ad Hoc Wireless Networks, Mobile Computing, Thomasz Imielinski and Hank Korth (Editors), Vol. 353, Chapter 5, pp. 153-181, Kluwer Academic Publishers, 1996
- [15] S. Shakkottai, T. S. Rappaport, and P. C. Karlsson, "Cross-layer design for wireless networks." WNCG Technical, Report TR-2003-04-00001, Wireless Networking and Communications Group,

Department of Electrical and Computer Engineering, The University of Texas at Austin. Available for download at <http://www.wncg.org>.

- [16] M. Chiang, To layer or not to layer: balancing transport and physical layers in wireless multihop networks, in: IEEE Annual Conference on Computer Communications (INFOCOM), 2004, pp. 2525–2536.
- [17] L. Qin and T. Kunz, survey on MANET routing protocols and cross layer design, Technical Report SCE-04-14, Carleton University.
- [18] D.S.J. De Couto, D. Aguayo, B.A. Chambers, and R. Morris, “Performance of multi-hop wireless networks: Shortest path is not enough,” Proceedings of First Workshop on Hot Topics in Networks (HotNets-I), Oct. 2002.
- [19] P. Gupta, P.R. Kumar, “The capacity of wireless networks”, IEEE Transactions on Information Theory 46 (2) (2000) 388–404.
- [20] Manthos Kazantzidis, “Wireless Adaptive Multimedia using Network Measurements”. UCLA CS Technical Report 200102, Feb 14, 2001
- [21] D.S.J. De Couto, D. Aguayo, J. Bicket, R. Morris, A high-throughput path metric for multi-hop wireless routing, in: ACM Annual International Conference on Mobile Computing and Networking (MOBICOM), 2003, pp. 134–146.
- [23] R. Draves, J. Padhye, and B. Zill, Routing in Multi-radio, Multi-hop Wireless Mesh Networks, ACM MobiCom, Philadelphia, PA, September 2004.
- [24] Y. Yang, J. Wang, R. Kravets; Designing Routing Metrics for Mesh Networks. WiMesh-2005, Santa Clara, CA 26th September 2005.
- [25] R. Draves, J. Padhye, and B. Zill, Comparison of Routing Metrics for Static Multi-Hop Wireless Networks, SIGCOMM’04, Aug 2004, Portland, USA.
- [26] Srinivasan Keshav, “A Control-Theoretic Approach to Flow Control”, ACM Annual International Conference on Mobile Computing and Networking (MOBICOM), 1991
- [27] L. Iannone, et al, Cross-layer Routing in Wireless Mesh Networks, 1st Int. Symp. in Wireless Communication Systems, Mauritius, September 2004.
- [28] U.C. Kozat, I. Koutsopoulos, L. Tassiulas, A framework for cross-layer design of energy-efficient communication with QoS provisioning in multi-hop wireless networks, in: IEEE Annual Conference on Computer Communications (INFOCOM), 2004, pp. 1446–1456.
- [29] R. d. Renecsse, et al, QoS enabled Routing in Mobile Ad-hoc Networks. Electrical Engineering Department, Centre for Telecommunications Research, King’s College London, UK.
- [30] L. Iannone and S.Fdida, MeshDV: A Distance Vector mobility-tolerant routing protocol for Wireless Mesh Networks, IEEE REALMAN, 2005
- [31] Giuseppe Bianchi, Performance Analysis of the IEEE 802.11 Distributed Coordination Function”. IEEE journal on selected areas in communications Vol. 18, No. 3, March 2000
- [32] M. Jaseemuddin, A. Esmailpour, O. Bazan, and A. Alwan, “Integrated Routing protocol for Wireless Mesh Networks”. IEEE CCECE, 2006, Ottawa, Canada
- [33] J. Wu, M. Jaseemuddin, and A. Esmailpour, “Integrating UMTS and Mobile Ad-hoc Networks”. IEEE WiMob, 2006, Montreal, Canada

[34] www.nortelnetworks.com.

[35] Stephen Mueller, Rose P. Tsang, and Dipak Ghosal, "Multipath Routing in Mobile Ad Hoc Networks: Issues and Challenges". Department of Computer Science, University of California, Davis, CA 95616. Proceedings of IEEE WCNC 2000

[36] Marc Mosko, J.J. Garcia-Luna-Aceves, "Multipath Routing in Wireless Mesh Networks"

[37] A. Raniwala, K. Gopalan, T. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks". ACM SIGMOBILE Mobile Computing and Communications Review archive Vol. 8 , No 2, April 2004

[38] P. Kyasanur and N. Vaidya, "Routing and Interface Assignment in Multi-Channel Multi-Interface Wireless Networks", IEEE WCNC, 2005.

[39] Mingyu Wu, Andras Farago, TR.: "A Survey of MAC Protocols in Ad Hoc Networks". Computer Science, The University of Texas at Dallas

[40] Ossama Younis and Sonia Fahmy, "HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad-hoc Sensor Networks". Proceedings of IEEE INFOCOM, March 2004