

1-1-2011

Interference-Aware Multicast in Wireless Mesh Network

Hasan Nicanfar
Ryerson University

Follow this and additional works at: <http://digitalcommons.ryerson.ca/dissertations>



Part of the [Digital Communications and Networking Commons](#)

Recommended Citation

Nicanfar, Hasan, "Interference-Aware Multicast in Wireless Mesh Network" (2011). *Theses and dissertations*. Paper 1453.

This Thesis is brought to you for free and open access by Digital Commons @ Ryerson. It has been accepted for inclusion in Theses and dissertations by an authorized administrator of Digital Commons @ Ryerson. For more information, please contact bcameron@ryerson.ca.

Interference-Aware Multicast in Wireless Mesh Network

By:

Hasen Nicanfar

BSc. In Electrical Engineering

Sharif University of Technology, Iran, 1993

A thesis

presented to Ryerson University to the

in partial fulfillment of the
requirements for the degree of

Master of Applied Science

In the Program of

Computer Network

Toronto, Ontario, Canada, 2011

© Hasen Nicanfar

Author's Declaration:

I hereby declare that I am the sole author of this work. I authorize Ryerson University to lend this thesis project to other individuals for the purpose of scholarly research.

Hasen Nicanfar

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

Hasen Nicanfar

Abstract

**Hasen Nicanfar, Interference-Aware Multicast in Wireless Mesh Network
MAsc., Computer Network, Ryerson University, 2011**

Interference is one of the most challenging issues in wireless network. Communication between a transmitter and receiver pair causes radio wave propagation to interfere with other nearby communication that may result blocking that link. In Multicast, since almost simultaneous communication takes place along several branches of a Multicast distribution tree, these branches tend to mutually block each other due to interference. In this research we introduced an Interference-Aware Multicast routing protocol.

First, it divides the nodes in the Multicast distribution tree in three GORby-Groups: Transmit, REC1 (receiving) and REC2 (standby). Then we calculate the SINR of the node based on interferences caused by all the nodes within that GORby-Group, assuming when the given node is in the transmit mode then all the nodes within its GORby-Group will also be in the transmit mode. Then we use the computed SINRs in our join Multicast group algorithm to achieve the most SINR.

Acknowledgement:

I owe my deepest gratitude to Dr. Muhammad Jaseemuddin, my supervisor who never accepted less than my best efforts, Dr. Bobby Ngok-Wah Ma, Dr. Victor C. M. Leung, Dr. Isaac Woungang, and Dr. Truman Cungang Yang for their time, support, technical guidance and advice.

Also, I would like to express my appreciation to all those who gave me the possibility to complete this thesis:

- My family for their support, and being patient.
- My former Employer, Pita Break and the Ozery family.
- Dr. Amir Esmailpour, Dr. Osama Bazan, Afshin Behzadan, Arseny Taranenko, Christopher Brierley and Kambiz Chorsi for their always help and support during my study at Ryerson.

Table of Contents

Author's Declaration:	II
Abstract	III
Acknowledgement:	IV
Table of Contents	V
Table of Figures	VII
Table of Tables	X
Appendices	XI
Chapter 1: Introduction	1
Chapter 2: Related Work	6
2.1 Basic related works:	6
2.2 Multicast related works:	9
2.3 Interference-Aware related works:	13
Chapter 3: GORby Multicast Routing Protocol	17
3.1 GORBY Definition	18
3.2 MAC Scheduling information	18
3.3 Joining Multicast tree	21
3.4 GORby Interference	23
3.4.1 Interference Calculation	24
3.4.2 Signal to Interference and Noise Ratio	24
3.4.3 Multicast SINR Calculation	29
3.5 Interference-Aware join Algorithm	31
3.5.1 Physical location information	31
3.5.2 Unicast Routing Protocol	31
3.5.3 Node GORby Multicast db tables	31
3.5.4 Join Algorithm definition	35
3.6 Cases of Interference-Aware join	38
3.6.1 Case_1: Joining with one hop away	39

3.6.2 Case_2: Joining with two hops away	40
3.6.3 Case_3: Joining with three hops away (three join options)	42
3.6.4 Case_4 Joining with three hops away (same join option)	44
3.6.5 Case_5 hybrid joining options (same and different join options)	44
Chapter 4: Simulation and Results	46
4.1. Simulation setup	48
4.2. Serial join analysis	51
4.2.1 Simulation result:	60
4.3. Traffic analysis	63
4.4. Parallel join analysis	67
4.5. Summary and overall evaluation	73
Chapter 5: Conclusion and Future Research	75
5.1. Conclusion	75
5.2. Potential future works	76
Appendices	79
References	101

Table of Figures

Figure 1: Hidden node problem	7
Figure 2: A Sample of Multicast Group Developed Tree	18
Figure 3: Joining established Multicast tree (1).....	21
Figure 4: Joining established Multicast tree (2).....	22
Figure 5: Interference Distance.....	25
Figure 6: Interference Factor	26
Figure 7: PIM Hello packet format.....	32
Figure 8: Joining Multicast group with one hop away	39
Figure 9: Joining Multicast group with two hops away.....	40
Figure 10: Joining Multicast group with three hops away (three join options)	42
Figure 11: Joining Multicast group with three hops away (same join option)	44
Figure 12: Sample network topology.....	47
Figure 13: Topology implemented in OPNET.....	48
Figure 14: Rec1 joins the tree	52
Figure 15: Rec2 joins the tree	53
Figure 16: Path options for Rec3	54
Figure 17: Path options for Rec4	57
Figure 18: Serial join analysis, overall traffic dropped by all nodes.....	60
Figure 19: Serial join analysis, average received packet by Rec3	61
Figure 20: Traffic analysis, average packet dropped per traffic	64
Figure 21: Traffic analysis, average (percentage) of traffic sent by RP received by Rec3	65
Figure 22: Traffic analysis, average (percentage) traffic sent by RP received by Rec4.....	66
Figure 23: Joining ten receivers in PIM-SM Multicast protocol	67
Figure 24: Joining ten receivers in GORby Multicast protocol.....	68
Figure 25: Parallel join analysis, average of convergence duration	69
Figure 26: Parallel join analysis, variance of convergence duration	70
Figure 27: Parallel join analysis, average of convergence activity.....	71
Figure 28: Parallel join analysis, variance of convergence activity.....	72

Figure 29: Router "mcast" and "tslot" attributes.....	80
Figure 30: Defining constants in header block	81
Figure 31: Defined SV (Global) variables	81
Figure 32: Initialization section (1).....	82
Figure 33: Initialization section (2).....	83
Figure 34: Variable definition for sending hello packet function.....	83
Figure 35: Sending GMI table record	84
Figure 36: Variable definition for hello message receive function.....	85
Figure 37: Receive hello packet function (1).....	86
Figure 38: receive hello packet function (2)	87
Figure 39: Calculating GIA based on GMI table records	88
Figure 40: Average traffic received by G_R25.....	90
Figure 41: Average traffic dropped the entire nodes	90
Figure 42: Average Multicast traffic received by Rec4.....	91
Figure 43: Average Multicast traffic received by Rec3.....	91
Figure 44: Multicast traffic received by Rec3	92
Figure 45: Multicast traffic received by Rec4	92
Figure 46: Traffic dropped by R_R21	93
Figure 47: Traffic dropped by O_R20	93
Figure 48: Rec4 throughput (bits/sec).....	94
Figure 49: Rec3 throughput (bits/sec).....	94
Figure 50: Rec3 delay (Sec).....	95
Figure 51: Average traffic dropped by R_R24	95
Figure 52: Average traffic dropped by G_R22	96
Figure 53: Average traffic dropped by O_R23	96
Figure 54: Average traffic dropped by O_R20	97
Figure 55: Rec3 traffic received (Packets/sec)	97
Figure 56: Average traffic dropped by entire nodes	98
Figure 57: Average traffic received by Rec3 (Packet/Sec).....	98
Figure 58: Average traffic dropped by entire nodes	99
Figure 59: Average convergence activity by entire nodes.....	99

Figure 60: Average convergence duration (Entire nodes)	100
Figure 61: Average traffic received by Rec4 (Packet/Sec).....	100

Table of Tables

Table 1: Sample of colour assignment to nodes	20
Table 2: State of Subsets in different Time Slot	20
Table 3 : PIM hello packet options	34
Table 4: Traffic characteristics	50
Table 5: Seed Values and Appropriate Symbols	51
Table 6: Rec3 Path1 PGIA calculation	55
Table 7: Rec3 Path2 PGIA calculation	55
Table 8: Rec3 Path3 PGIA calculation	56
Table 9: Rec3 Path4 PGIA calculation	56
Table 10: Rec3 Path5 PGIA calculation	57
Table 11: Rec4 Path1 PGIA calculation	58
Table 12: Rec4 Path2 PGIA calculation	58
Table 13: Rec4 different path PGIA calculation.....	59
Table 14: GIA of nodes for topology of Figure 17	62

Appendices

Appendix A: OPNET development	79
Appendix B: OPNET Reports	89

Chapter 1

Introduction

Researchers have paid more attention to the wireless environment during the last two decades. They have worked with different wireless platforms like Wireless Mesh Network (WMN), Wireless Sensor Network (WSN), Ad-Hoc, Mobile Ad-Hoc Network (MANET), Home Area Network (HAN), Vehicle Area Network (VAN); also, they have investigated and addressed many issues in different sections such as QoS, Routing Protocols by designing different metrics and developing algorithms for these purposes, and so on. However, this research area still requires more attention and work to be done, and there is still room to work in wireless. Especially because of the market demand and attention to the wireless platform and developing new applications based on wireless nodes. In this research, the main target is to develop an algorithm for Multicast over WMN to be Interference-Aware, and this algorithm is capable of being developed for Ad-Hoc, MANET, WSN and other platforms as well.

Interference is one of the most challenging issues to deal with in the wireless environment, because of the nature of wireless communication that requires electromagnetic wave broadcasting. There is an interference in almost all kind of the wireless communications regardless of the source(s) and destination(s) targets (Unicast, Multicast or Broadcast), the packet contents (the messaging data or controlling data), platform (Sensor, Ad-Hoc, Mobile, Mesh and etc.), application (Peer-to-Peer, Email, Phone and etc.) and routing protocol (OLSR, AODV and etc.). In fact, the main root cause of interference is the Physical Layer characteristic in the

wireless communication channel, which affects communication throughput and actual bandwidth.

IEEE 802.11 minimizes the impact of collision and solves the hidden node issue by using Request-to-Send and Clear-To-Send (RTS/CTS) as long as all the nodes are one-hop away from each other. As a matter of fact, the RTS/CTS algorithm constitutes the basis of GORby-Grouping network nodes in this research. In this study, we assumed collisions were being minimized by 802.11 and we pay attention to the interference. To be more precise, if the source of interfered communication is within a maximum distance of a detectable signal (one hop away) or the distance threshold is, for instance, 1000 meters, then its interference will be minimized most likely by RTS/CTS. However, if the distance is more than this limit/threshold (e.g. about 2000 meters), the electromagnetic wave reaches the destination as a noise (or in this case as interference) and will interfere with the actual communication. It actually affects the Signal to (Interference and) Noise Ratio (SINR) and the transmission successful rate.

There is some research [4, 5, 10, 12, 17] that has paid attention to this issue and the side effect of wireless channel communication from different points of view. For example, authors in [10] provided a solution about modeling interference in wireless channels. In [4], I. Tinnirello et al. estimated interference in 802.11 base networks. J. Wang et al. in [5] have developed a Multicast routing protocol in Wireless Mesh Network to be Interference-Aware; and Y. Yu et al. in [12] presented an Interference-Aware Quality of Service (QoS) supported for On-demand Multicast Routing Protocol (ODMRP). Also Y. Yu et al. in [17] studied the impact of the interference on Multicast throughput over an ad hoc network.

The main idea that we are working on in this research is to design an Interference-Aware Multicast routing protocol for WMN in a distribution section. It is based on the idea of Grouping the nodes in the Multicast distribution tree into three GORby-Groups, and then using the node location information to calculate the nodes distance that we use it in our SINR calculation (We will describe it in next chapter in detail). We can calculate SINR of each GORby-Group of nodes, and then join the Multicast group tree based on an SINR based metric. We assume all sources are in the internet and their data reach and enter the WMN via one gateway, which also acts as a Rendezvous Point (RP) in the case of a shared tree based Multicast group.

Our contribution in this research is the idea of GORby-Grouping the nodes into three GORby-Groups based on their interference pattern and the design of an Interference-Aware Multicast routing protocol.

First, it divides the nodes in the Multicast distribution tree in three GORby-Groups: When one GORby-Group is in Transmit (sending) mode/state, then the second GORby-Group is in REC1 (receiving) mode/state and the third GORby-Group is in REC2 (standby). In fact, our investigations show that MAC Layer support required to distribute the nodes in GORby-Groups should be capable of performing perfect Scheduling (e.g. TDMA). To accomplish formation of GORby-Groups, we have used the model of scheduling in automobile production line called GORby (Green-Monday, Orange-Tuesday, Red-Wednesday, blue-Thursday and yellow-Friday). Each colour represents one day of the production plan, and is demonstrated by colour of all bin labels regarding the day of production. For example, all of the products and parts producing for Monday are Green, while Tuesday bin labels are Orange and so on. In our development we have just used first three colours out of five. This model is more useful for streaming flows, like FTP and Peer-to-Peer applications. The GORby model provides a good basis for the implementation of the proposed Multicast routing protocol that prevents (minimizes) collision as well as delivers a good Signal to Interference and Noise Ratio (SINR) in joining the tree process. This makes protocol Interference-Aware.

In the second step, we calculate the SINR of the node based on interferences caused by all the nodes within that GORby-Group, assuming when the given node is in the transmit mode then all the nodes within its GORby-Group will also be in the transmit mode. Then we use the computed SINRs in our join Multicast group algorithm to achieve the most SINR, which we call it GIA (GORby Interference-Aware) metric. Moreover, we define Path GIA (PGIA) for the new branch of the Multicast distribution tree when the first node on the existing tree that joins the new branch is at a distance of more than one hop from the new receiver. The selection of new branch is based on PGIA using the Max-Min approach. To develop this metric, we introduce PGIA to be the minimum of GIA's of all nodes along that path, at which point the receiver (new member) selects the path that can provide the most PGIA.

We need the support of GPS location information system for calculating SINR at nodes in the distribution tree. The GPS location information is accurate by ten to fifteen meters tolerance, which is a good enough estimate for the purpose of the proposed protocol. Based on the assumption of low noise environment, target node SINR can be calculated based on the distance of the node to the sender as well as to all other nodes that their signal transmission would cause interference with the sender signal on a receiver node. This design is based on PIM-SM, and a new Multicast table that records the location and colour of the nodes. Because this model considers the interference of current Multicast group nodes, it needs to know only the location information of these nodes (Multicast group), plus the colour or potential colour of each node in order to calculate the SINR based on each GORby-Group.

This thesis is organized into five chapters and two Appendices. Chapter Two reviews works related to the development and design of the proposed algorithm. This chapter has three sections. The first section surveys various wireless routing protocols for Multicast in the literature. The second section studies Interference-Aware routing developments and research, which are designed in a wireless environment like MANET, Wireless Mesh Network. The third section is about other related mechanisms that are being referred to in our research, like RTS/CTS, 802.11 standard evaluations, OLSR routing protocol, and rules to define a metric.

Chapter Three presents the detail of our algorithm and design. It includes a brief study about the estimation, and defines the GORby-Grouping concept. It also has a detailed explanation of the routing metrics, in node based (GIA) and path based (PGIA). Then this chapter has a detailed explanation of the new proposed Interference-Aware algorithm that is provided when joining to a Multicast group tree. At the end, the proposed protocol demonstrates a mechanism for keeping the tree alive and using the hello packet to send for Keep Alive purposes as well as node information. This information includes location and hop away from RP to be used in calculating nodes colour for GORby-Grouping purposes.

Chapter Four studies the performance of the proposed algorithm through simulation using OPNET. The simulation results were analyzed and the performances of the joining algorithm and Multicast routing protocol in different network topologies were evaluated.

Chapter Five concludes the discussion. Future potential works will be suggested to be done in this area to expand the model. Also, references that we used in our research followed Chapter Five.

Appendix A presents the developments that we have done in OPNET to implement our algorithm for evolution. This appendix also includes some of the actual programming codes. Appendix B presents some of the supporting reports generated by OPNET, which we have use (for instance) average of the source of these reports data in our reports shown in chapter Four.

Chapter 2

Related Work

This chapter describes some of the other research that has been done in relation to the subject of our research. First we describe and recall the basis related work and standards mainly regarding MAC layer. Then we bring Multicast related research that proposed Multicast routing protocol and algorithm/metric. Finally we explain Interference-Aware related works, which proposed interference aware metric and algorithm.

2.1 Basic related works:

In this section, four related topics are being studied: Hidden node problem and RTS/CTS Mechanism, 802.11 Performance analyses, Guidelines on Routing Metrics Design and the success of CSMA/CA evaluation.

- **Hidden node problem and RTS/CTS Mechanism [31, 32]:** To explain the RTS/CTS mechanism that is used in GORby-Grouping in this research, first we need to explain the hidden node issue in the wireless environment.

As it is shown in Figure 1, let us assume node N2 wants to send data to node N3.

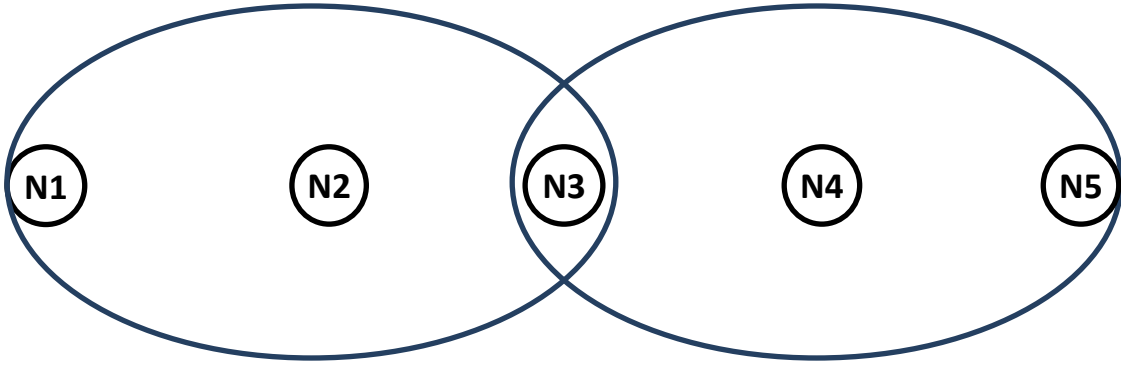


Figure 1: Hidden node problem

In this situation, node N4 does not sense anything in the channel (following the CSMA mechanism in 802.11), and might send data to node N3 as well. Consequently, it causes a collision at node N3, and then node N3 cannot receive data (theoretically) from any of them. It is the same in the case of sending data by nodes N1 and N3 to node N2, which causes a collision at node N2.

To handle this issue, RTS/CTS uses a four-way handshaking, which, for instance, we described avoiding collision at node N3:

- (i) N2 senses the media, if it finds the channel free then sends a Request-to-Send (RTS) packet to node N3.
- (ii) N1 receives RTS (to N3): N1 fires Network Allocation Vector (NAV-RTS), and becomes silent until the end of the communication.
- (iii) N3 receives RTS (to N3): If it is free, it sends a Clear-to-Send packet (CTS) back to node N2.
- (iv) N4 receives CTS (to N2): N4 fires NAV-CTS and becomes silent.
- (v) N2 receives CTS (to N2): Gains access to the media and sends the data/packet.
- (vi) After receiving the packet completely, N3 sends an Acknowledge (ACK) packet to node N2.
- (vii) ACK turns off NAV and lets the next node (any of them) gain access to the media.

We use this mechanism as our basis in chapter three to define our GORby-Grouping model. There are other time frames (and packets) related to this mechanism like SIFS and

DIFS, although we only explained the main steps to show how this mechanism works overall.

- **802.11 Performance Analysis [7]:** This work is actually one of the first and most referred analyses that are being used for other research regarding 802.11 on the MAC layer. It evaluates and analyzes the MAC layer behaviour based on different definitions of the 802.11 standard, especially the Distributed Coordination Function (DCF), which is a CSMA/CA scheme with a binary slotted exponential backoff algorithm. It uses a Markov chain model to show the backoff algorithm and analyzes the throughput of DCF. The analytical model covers basic access and RTS/CTS access which are both covered by DCF, as well as a combination of both of them that RTC/CTS in packets that are more than packet-sized threshold. It defines the probability of going to each stage and also the transmission between the states based on the algorithm definitions as well as the probability of a packet transmission success, which is assumed to be constant (e.g. “p”) for all nodes by this analysis. Then based on the Markov Chain model, this study calculates the probability of each state and moves from one to another based on the packet transmission success probability (p), time slots, and random variable defined by standard. By having this information, this study calculates the probability of transmission for a node/station in a randomly chosen slot time, which would be a function of “p”. In the next phase, the study uses this information to find the throughput of the network.
- **Routing Metric Design [19]:** This research presents a guideline to a design routing protocol metric, and characteristic that this metric should be able to serve the routing protocols, in reactive and proactive mode of a wireless network. Y. Yang and J. Wang in [19] described that each routing protocol consists of two parts including “Path Calculation Algorithm” and “Packet Forwarding Schemes”. In each routing algorithm, the goal is to find the best path between source and destinations. This path should be the lowest cost path. To be precise, it means we need a metric to be used as the cost of the link and then based on this cost, finds the best path. In fact, this will aid with defining a metric to be used as a cost parameter. Moreover, this metric should be used in a routing protocol. It leads to define the routing protocol requirements that are Consistency,

Optimality and Loop free. As well, routing protocol requires its path weights to have isotonicity and monotonicity properties. Isotonicity properties means that if the cost of path A is less than path B, then adding one more link (L_1) to each path (from left or right) does not change the relationship. Based on this example, the cost of path “A + L_1 ” is still less than the cost of path “B + L_1 ”, as well as the cost of path “ L_1 + A” is still less than the cost of “ L_1 + B”. Also monotonicity property means, by refer to the previous example, the cost of path A is always less than cost of path “A + B” and cost of path “B + A”.

Note: The metric presented by this document is based on the SINR, and in the case of path metric, it is based on the minimum of nodes metric value along the path. It is obvious that by adding a new link to a path, the Max-Min function that is a nonincreasing function, always follows the above mentioned two properties. This is described in more detail in chapter three.

2.2 Multicast related works:

To study related work in terms of Multicast protocol, POEM [5] and EGMP [27] are studied in this section, which are most recent and popular ones. From POEM, we adopted the concept of prefix assignment for our colouring model, and EGMP is a location based protocol, which we bring it here because we are using a similar approach of using location information. Also, we describe work of [8], which reviewed and analyzed the different proposed Multicast solutions for a wireless environment.

- **POEM [6]:** Prefix Ordering for Efficient Multicast (POEM) is designed based on Labelled Directed Acyclic Graph (LDAG), which is developed for receiver-initiated Multicast models.
 - First phase (Unicast related steps): It assigns prefix label to each node of the network as per node location in network topology versus distance (layer – hop) to the core of the network. To be more precise, the core prefix has one

digit/character and nodes that are connected to the core, will have two places (two digits/characters) prefix label, and thirds layer, which are the nodes connected to the second layer nodes will have three places (three digits/character) prefix label, and so on. For instance, the core has a prefix label of “1”, the second layer nodes will have prefixes labels like “11”, “12” and “13”, and the third layer nodes prefixes will be “121”, “122”, “134” and “147”. By looking at each label, the number of options (places) is like hop count to the core. Each node keeps and maintains neighbour table (NT) and neighbour of neighbour table (TNT) as well.

- Second phase (Multicast related steps): Multicast group members use hashing to map Multicast group name. Members select the closest prefix label to their group name to act as a Multicast group core. This core will serve the group as long as its prefix is the best match to group prefix. Each node for this purpose maintains another table called group information list (GIL) as well, and required to be updated by using hello messages. This information will get expired by using an expiry time (Short Timeout – ST).
- Then based on this core, they form share tree by sending join requests toward group prefix label in receiver initiated Multicast model. Then if the core fails by moving away from the network (mobility), a neighbour that matches the most after the original core will become the new core for the Multicast group.
- Also, senders send their packets to the core. All they need to do is sending packets toward the group prefix label. If the sender is not part of the group, it uses the prefix routing to reach the core by sending the packet/s to the neighbourhood node that has the most prefix match to the Multicast group ID (core). If sender/source is part of the tree, it uses same model plus if it has some members in his sub tree, it will send the packets to this sub-tree directly.

This algorithm has covered joining and leaving nodes from the group based on above description, which in fact maintains the group in a dynamic manner that is required for mobile ad-hoc networks.

- **EGMP [27]:** Efficient Geographic Multicast Protocol is designed as a Multicast routing protocol for MANET, which is a dynamic environment. It basically uses location

information of the nodes, which they can receive their location information from the available systems such as GPS. Based on the location information, a network is being divided to different virtual zones as per self-organized by nodes. Then in each zone, one node is assigned/selected to be the zone leader to manage the node membership as first duty in the lower layer. In the upper layer, zone leader would act as the zone representative to join a Multicast group. To use this structure in Multicast, zone information would be added to the Multicast packets and it will be sent to the leaders in the upper layer. In the lower layer, after a zone leader receives the packet, will transmit the packet to the Multicast members of this group that are part of such zone.

Up to this step and from an overall view, EGMP has combined the location based modeling and the cluster modeling together to design a new model. In fact, EGMO is a structured location based model that is being used in the Multicast group. By this design approach, EGMP actually distributes handling a dynamic network (the nature of MANET) overhead to the structured base group that is called a zone. This model does not decrease the number of controlling packets and in fact increases them per nodes movement; however, it decreases broadcasting the controlling information. To be more precise, moving management controlling packet is being coordinated inside a zone and does not need to be broadcasted outside of the zone. All a Multicast group needs to know is the leader information and improves the delay and overhead.

The authors studied and showed that how this protocol can address zone management as well as the packet forwarding related issues. From the zone structure point of view, EGMP introduces zone (a square shape sub net of network), zone size (the length size of zone square), zone ID (zone identification), zone center (the physical position of square zone), zone leader (node that acts as leader), tree zone (the zones that are in Multicast group), root zone (zone that includes Multicast root) and zone depth (the distance of the zone to the root zone). In order to select a zone leader, it uses the closer to the zone center and then the highest ID as appropriate parameters to choose the leader, if there is no other leader already selected. This grant will not be withdraw as long as leader stays in the zone, which in fact it is not very good algorithm in dynamic environment. For transmitting information, two models have been introduced, which are inter-zone and intra-zone communications. Nodes inside a zone communicate directly with each other

because they are aware of their physical location. For a multi zone communication, each packet includes the zone ID and will be routed to the zone leader and then the zone leader handles the internal zone communications. In terms of tree structure, it uses location information, which actually reduces tree management overhead and improves the join and leave delay.

- **Ad-Hoc Multicast Routing Protocols Performance Analysis [9]:** M. B. Krishna and M. N. Doja in [9] have presented an analysis from QoS standpoint on different tree based Multicast routing protocols that exists for a wireless ad hoc network. To do this, they have chosen Multicast Ad hoc On-demand Distance Vector (MAODV), Multicast Zone Routing Protocol (MZRP), Multicast Optimized Link State Routing (MOLSR), Associativity-Based Ad hoc Multicast routing (ABAM) and Ad hoc Multicast Routing protocol utilizing Increasing identify Numbers (AMRIS) to be investigated and being analyzed. They analyzed QoS on the above mention Multicast routing protocols based on the Robustness, Efficiency, Combination of unicast protocols, Energy optimization, Network control overhead and Resource management. This analysis showed MAODV, MZRP and MOLSR have more control packet overhead and network neighbourhood node dependence comparing to ABAM and AMRIS. Moreover, if a packet is routed to the outer zone, then MZRP would have the most control packet overhead between all of them. Also the study showed ABAM and AMRIS are based on association of the neighbour nodes and increase in the routing path identity. As a result, MAODV, MZRP and MOLSR are more suited for the Multicast group routing protocol, if the network experiences denser nodes topology. On the other hand and in a less dense nodes topology, ABAM and AMIRS protocols are performing better. Finally, ABAM shows better performance in non-scalable applications and AMRIS in less routing complex applications.

2.3 Interference-Aware related works:

In this section, more interference related works are studied, which is the main area of our research. First one [10] is about modeling interference in a network, and other three (CMIM [4], ASWP [13] and FIRM & FIRM+ [18]) are about Interference-Aware protocols.

- **Interference Modeling [10]:** A. Iyer et al. studied different interference modeling presented by other researchers and provided their own SINR based approach, to model interference in a wireless channel for analyzing purposes [10]. The whole idea and concept here is about looking at interference as a cumulative object. For a target link that is under investigation, some research has brought the interference of other links individually, which authors challenged this idea and modeling. They analyzed and proposed that first of all this interference should be cumulative, and secondly the target link transmission will be successful as long as the SINR is more than a SINR threshold. They also presented two characteristics of the interference called “extent of conflict” and “nature of conflict”. For “extent of the conflict”, they studied interference of one link on target link in an isolated environment. Under this circumstance, they defined a minimum interference range that is the minimum distance between a link and the target link such that transmission of the link causes interference on the target link transmission. Also in “nature of the conflict”, they represented the interference source by all other links totally on target link transmission. Based on this, they defined the additive mode such as the total of interferences by all of the links on a given (target) link receiver.

Note: Our presented model in this document follows above mentioned interference modeling. As well, by GORby-Grouping the network nodes, we calculate the interference caused by each GORby-Group cumulatively and separately. Also, our model follows the rule of having a SINR more than SINR threshold.

- **CMIM [4]:** Centralized Minimum Interference Multicast (CMIM) is developed for a wireless mesh area network topology based on having multi-beam sectorized directional antenna. Two models are being used including Fixed Orientation and Fixed Beamwidth

(FOFB) and Adjustable Orientation and Fixed Beamwidth (AOFB). Firstly, it modeled the problem of interference as an optimization problem by using an integer linear programming model. Then it introduced the idea of using AOFB instead of FOFB, which antennas orientation changes and gets adjusted to reach the minimum interference. To model the interference, they used both beamwidth and antenna orientation and developed a detail optimization problem based on these two. Then to develop CMIM, they introduced three phases. In first phase, they structured a tree by using omnidirectional antenna and added branches/trees one-by-one. Then, they defined the cost of a source and destination pairs to be the minimum of all of the possible paths, which the path cost is equal to the sum of all link cost along the path. Any time a new branches gets added to the tree, the cost of Multicast tree will increase by one; however, the path may get zero cost because they may receive the information as per advantage of Wireless Broadcast Advantages. It means if a node sends information in a wireless environments, others will receive it as long as being in a certain distance around. In second phase, antennas set their beamwidth and ordination based on FOFB model. In third phase, the orientation of antennas is adjusted to reach the minimum interference. In overall, this model used this point that the distance between links (and nodes) is important in causing the interference; however, the beam direction changes this interference and SINR as well, which is the key to design of the model. In other words, the nodes may be close and at the same time, not receiving the wave and not getting affected by a sender broadcasting magnetic waves.

One of the point about this model is actually in phase one of the CMIM. It added the branches to the tree one at the time. The order of this group development and structuring causes different end results. Also as authors mentioned as well, it could be used just in a fixed and static topology. Although it is designed for the WMN, just backbone of a WMN might be static and nodes and receivers may not have a fixed position and they could move around. This movement of nodes will cause having a dynamic environment, which CMIM should execute all three phases again and adjusting the antennas and it will end with a high overhead for maintaining antenna beamwidth and orientation. Moreover, the network load and traffic could be (and it is in fact) dynamic and makes high antenna maintaining overhead again.

- **ASWP [13]:** Ad-Hoc Shortest Widest Path tries to find an Interference-Aware path from a source to a destination in the subject of being widest and then shortest one. It adopts Bellman-Ford architecture and K-Shortest-Path model, and tries to find the optimum path to the destination. It uses conflict graph and clique model to have an Interference-Aware path. Main point about this algorithm is finding the widest path first, and then the shortest one. The result is a better QoS; however, may (will) use more network resources. It starts by the sender, and finds a best path for each clique and step-by-step extends the path to reach the destination.

In this model, assumption is finding the best path between sender and receiver. So, expanding it to a Multicast will cause each source calculates the path between itself and all receivers. If Multicast data is already transferred to some nodes (receiver or relay node), a new join needs to happen between the new member and an already established tree. Even if we want to join the tree, joining point should be fixed, which could not guarantee the best join. This research shows the best path and the best joining node could not be predefined.

Also based on the nature of this model that finds widest first, it results in a Multicast model to have one tree per receiver. Receivers are trying to find widest path and because the tree is already established, this widest path would be a new tree.

Also, it can be used in a source based/initiated Multicast model. Source needs to know all of the receivers and develops and calculates the path to all of them.

- **FIRM and FIRM+ [18]:** Flow-based Interference-Aware Route Management (FIRM) has been developed for a wireless mesh networks to be inter-flow and intra-flow Interference-Aware. It is designed based on providing the maximum bandwidth for transmission. Also it is based on 802.11 for MAC layer, which is more close to our research. The assumption is that a sender and a receiver can choose and have different paths for transferring data, and they choose the best one based on the maximum bandwidth overall for the network. In FIRM, just finding a path with the maximum bandwidth is the target. FIRM+ brings the drawback of selecting a path on current network flow as well, which is an improvement on FIRM. In order to do this, it follows the following steps:

- Available air time at receiver node: Based on current data flow in receiver node that are actually reserved receiver air time, it calculates the available time percentage per time unit. It is one time unit minus assigned time portions for sending and receiving by the receiver node.
- Available air time at sender node: It calculates air time similar to the receiver case, also decreases it by the amount of the time this sender senses the other nodes around data transmission that causes conflict on the sender.
- In the next step, it sets the available air time of the link between a sender and a receiver to be the minimum air time of each one.
- Available link bandwidth: It brings hidden interference affects to the account. It defines a new parameter called “level” on the interference of a link caused by the other nodes transmission. Then multiplies air time by “one minus this parameter” (interference level), and calls the result as “available link bandwidth”.
- FIRM(p): It uses clique modeling to find available bandwidth of each clique, and then sets a path bandwidth to be the minimum of available clique bandwidth of all cliques along the path. FIRM uses FIRM(p) as a metric to choose the best path.
- FIRM+: FIRM does not bring the effect of current flow set up on previous flow in a network, which FIRM+ covers this effect by bringing this into account. It deducts from available bandwidth the side effect of the new traffic setup.

First of all, both FIRM and FIRM+ require full knowledge of the topology including all nodes capability. Then, they need to know the current data flow in all links as well as cliques' situation. And then, both of them require all of this calculation every time a change happens in the network because both are based on a current situation. Also, they are one-time calculation, which cannot be valid in a dynamic environment.

Chapter 3

GORby Multicast Routing Protocol

GORby model developed in this research is an application of a production scheduling technique used in auto-manufacturing. Based on the PIM-SM Multicast routing protocol and IEEE 802.11 standard in the MAC layer, it has been designed to be an Interference-Aware Multicast Routing Protocol on the Wireless Mesh Network (Mainly proposed model would be an algorithm used for joining Multicast group). In our study, we made a number of assumptions and model specifications, which are summarized as follows:

- (i) It is for a receiver initiated join model.
- (ii) It is designed for a group based.
- (iii) This model is based on the PIM-SM protocol.
- (iv) It is designed for a WMN, which is almost a stable network.
- (v) Multicast sources are located outside the wireless network and all of the Multicast traffics from the sources enter the wireless network from the same gateway.
- (vi) Nodes get their location information from a GPS system.
- (vii) All of the nodes are 802.11 (MAC Layer) nodes with the same frequency channel and 360 degree beam (no directional beam and antenna).
- (viii) Transmitted power is the same for all of the nodes in the wireless network.

- (ix) The thermal noise level is assumed to be much lower than the interference level, and thus negligible.
- (x) Environmental condition is the same for all of the nodes, which results in having the same path gain/loss.
- (xi) Data traffic volume is much larger than the control data volume, which is an obvious assumption in this type of networks [18].
- (xii) Just interferences caused by the current Multicast established group traffic are being taken into account.

3.1 GORBY Definition

GORBY is an abbreviation of Green, Orange, Red, Blue and Yellow. In this research just three of original five colours are used, although the full name is kept to show the origin of the model. The colours are used for GORby-Grouping the network/topology based on the traffic and states, including Transmit (Sending) State, REC1 (Receiving) State and REC2 (Standby) State. In the rest of this document, GORBY is changed to GORby (small character for “B” and “Y”) to emphasize that only three colours are used (Green, Orange and Red).

3.2 MAC Scheduling information

Let us assume that source node S sends data to destination node D via nodes (routers) N1 to N10 in the topology shown Figure 2. Also, let us assume the traffic is a stream flow.



Figure 2: A Sample of Multicast Group Developed Tree

In this example, our assumptions are:

- The MAC layer is a scheduled based model and nodes are synchronized.
- The channel is slotted and most of the bandwidth is being used. Also, there is a one way flow traffic for this group (core or source node is S).

In our algorithm, a node can be in one of the following operating states:

- **Transmit** state: A node sends Multicast data to the downstream node/s.
- **REC1** state: A node receives Multicast data from the upstream node.
- **REC2** state: A node is standby in this state, neither sends nor receives Multicast data; however, receives reverse Multicast data controls, if any. The reverse control traffic load is assumed to be low.

Back to Figure 2, let us only pay attention to the first three nodes (S, N1 and N2) and assume S is currently in the transmitting (Transmit) state, and then N1 is in the receiving (REC1) state and should not send any data. In order to avoid a collision at N1, N2 should also refrain from sending data, thus be in the standby (REC2) state. Subsequently, every other three nodes along the path will repeat the same sequence of states. For example, N3 is in the transmitting state, N4 is in the receiving state and N5 is in the standby state and so on. By repeating the same sequence, collision can be avoided.

Table 1 shows the state of each node in Figure 2, with the assumption that S is in the transmitting state. The topology nodes in the transmitting (Transmit), receiving (REC1) and standby (REC2) states are labelled green, orange and red, respectively. In this table, each series means three consecutive nodes, e.g. first series means nodes {S, N1, N2}; second series means {N3, N4, N5} and so on.

As it is depicted in Table 1, when S is in the Transmit State, other nodes in this GORby-Group (Green) would be in the same state. Then, in the next time slot, all of these GORby-Group nodes

will switch to the next state. Table 2 presents more detail of this state changing between the GORby-Groups.

Table 1: Sample of colour assignment to nodes

State	Transmit (GORby-Group1): Green	(REC1) Receive (GORby-Group2):	(REC2) Standby (GORby-Group3):
First Series	S	N1	N2
2 nd Series	N3	N4	N5
3 rd Series	N6	N7	N8
4 th Series	N9	N10	D

Note that based on this model and assumptions, each node uses just about 1/3 of the overall bandwidth at the best. Regardless of following our algorithm or not, and based on the IEEE 802.11 standard, nodes are actually in one of these three states. One of our algorithm outcomes is synchronizing the nodes to prevent collision and reaching this 1/3 of bandwidth utilization, otherwise utilization is less than this value.

As per Table 2, nodes with the same colour (belonging to the same GORby-Group) are in the same scheduling state all the time.

Table 2: State of Subsets in different Time Slot

Time Slot	Green GORby-Group	Orange GORby-Group	Red GORby-Group
1 st Time Slot	Transmit	Receive (REC1)	Standby (REC2)
2 nd time Slot	Standby (REC2)	Transmit	Receive (REC1)
3 rd Time Slot	Receive (REC1)	Standby (REC2)	Transmit

3.3 Joining Multicast tree

Figure 3 shows an example of joining a Multicast group, which is a similar topology presented in Figure 2. Let us assume this topology is a Multicast tree and a new node (X) wants to join, so X broadcasts a Route Request (RREQ). Node R, which is not a group member, receives RREQ and then broadcasts RREQ again to find the path to the group. In this case, RREQ sent by node R is received by node N6 (a group/tree member), and N6 answers the request by sending Route Reply (RREP) back to the R. Following this action, R receives RREP, joins the tree, and then responds to the X by a new RREP. Node X receives RREP and will join the group.

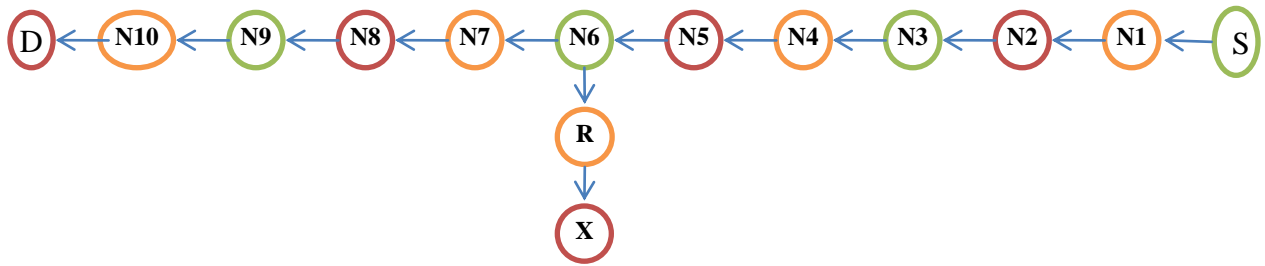


Figure 3: Joining established Multicast tree (1)

So, based on the above mentioned steps, a new branch is established. This sub-trees' nodes (R and X) get appropriate colours based on the N6 colour. For instance, in the case illustrated in Figure 3, the colour of node N6 (Green) is included in the RREP packet responding to the R. When R receives it, it sets itself to be an Orange node (one after Green) and includes its colour in the RREP responding to the X. Similarly, node X joins the group through R and sets itself to the colour one after Orange, which is Red. As a result, R and X become part of the Orange and Red GORby-Groups respectively. After joining the group, for instance in case of node R, as soon as R receives a data packet from N6 for the first time (it is in REC1 state), calculates the time slot of the group and sends the data during the next time slot (goes to Transmit state). It also knows

that based on the algorithm (Table 2), the next state after Transmit is REC2 (Standby). This is how they get synchronized.

We can define a special synchronization packet sent by RP to all of the nodes to synchronize them. We suggest this to be developed more detail as future research in chapter Five.

To develop our presented concept further, let study Figure 4, which is an expansion of Figure 3, by having five nodes that want to join the tree (group). In this example, X1, X2, X3, X4 and X5 are the new receivers that want to join the Multicast group as new members; so, they broadcast RREQ.

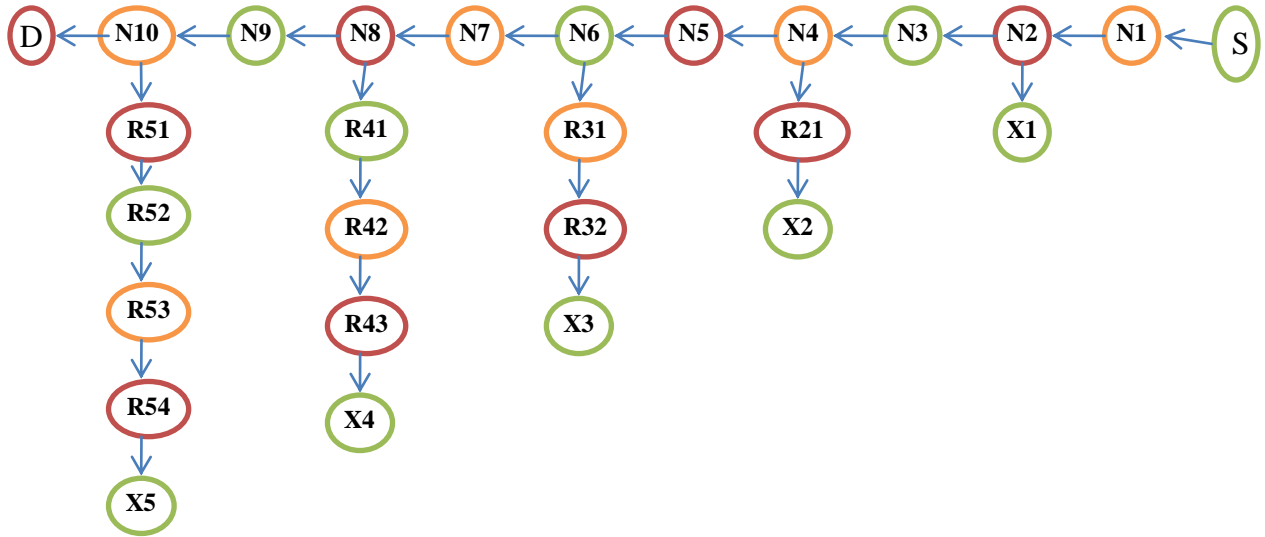


Figure 4: Joining established Multicast tree (2)

In the case of X1, it joins the tree directly (one hop away) by receiving RREP from N2, although in the case of other receiver nodes (as per Figure 3 related discussion), a new sub-tree (branch) should be established (by following previous example steps). This new branch's nodes including the receiver node itself get an appropriate colour based on the join node colour (Following GORby). As we discussed in Figure 2, the colour of the join node is included in RREPs packet responding to RREQs. The middle (relay) nodes receive the RREPs, extract the sender colour

and figure out their own (potential) colour, join the group as relay node and part of the tree (sub tree), and then create RREPs responding to the downstream nodes including own assigned colour. Based on this algorithm, all of the nodes along the path to the receiver will have appropriate colour in the same sequence/format of Multicast group ($G \rightarrow O \rightarrow R$), starting by a main tree join point colour.

3.4 GORby Interference

In this section, we introduce the new concept of GORby interference.

We make the following additional assumptions:

- We have a high Multicast traffic flow, and it is assumed to be the main (or the only) traffic in the network.
- Interference is just caused by the transmission of this traffic stream.

Following section 3.2, at each time slot only one third of the nodes are in Transmit State. So, if interference caused by Multicast traffic on a node be the case, it will have three kinds of interferences. For instance, in a time slot that Orange nodes are in REC1 (Receiving State), they are experienced interference caused only by Green GORby-Group nodes (because only Green GORby-Group nodes are in Transmit State), which is the main key point of our algorithm. In fact, Orange nodes experienced interference caused by other two subsets (Red and Orange) as well; however, because of being in different GORby-Groups and time slots for being in REC1 State, those interferences do not affect Orange GORby-Groups nodes.

The above mentioned description means that there are three kinds of interferences:

- Green Interference: Caused by Green GORby-Group members (being in Transmit State).
- Orange Interference: Caused by Orange GORby-Group members (being in Transmit State).

- Red Interference: Caused by Red GORby-Group members (being in Transmit State).

Also based on GORby Group definition, when “Aj” (current GORby-Group) is in REC1 State, Multicast Group members indexed “i” (previous GORby-Group) in GORby sequence (where “j = i+1 Mode (3)”), are in Transmit State.

- If Aj is in REC1 State → Ai is in Transmit State.
- E.g.: if A0 = G(reen) → A1 = O(range) and A2 = R(ed)

Up to this section, we defined the GORby interference concept.

3.4.1 Interference Calculation

To calculate the received power by Node_b from Node_a Eq.1 is used [10, 31, 32]. This received power could be the main signal or interference:

$$Pr = K \times \frac{Pt}{D^n} \quad Eq. 1$$

Which, “Pr” is Received Power at Node_b, “Pt” is Transmitted Power by Node_a, “K” is a constant value (function of channel, environment, path loss and antenna gain), “D” is the distance between two nodes (Node_a and Node_b), and “n” is a number between two and four. In most research, “n” is assumed to be two and is used in our research as well. Signal at Node_b is detectable if “Pr” is more than Power Threshold [10, 31, 32].

3.4.2 Signal to Interference and Noise Ratio

Let us assume Node_b is in REC1 state. We use Eq.2 to calculate Signal to Noise and Interference Ratio at this node [31, 32]. Node_b will receive data from Node_a successfully if this node SINR is greater than or equal to “β” (threshold) [10].

$$SINR = \frac{Pr}{N + \sum_j I_j} \quad Eq. 2$$

Where:

- SINR is Signal to Interference and Noise Ratio.
- “Pr” is power received (of the signal that carries information, e.g. Node_i).
- “N” is environment noise excluding interference (like thermal noise).
- “ I_j ” is Interference caused by the transmissions of all the nodes, excluding Node_i.

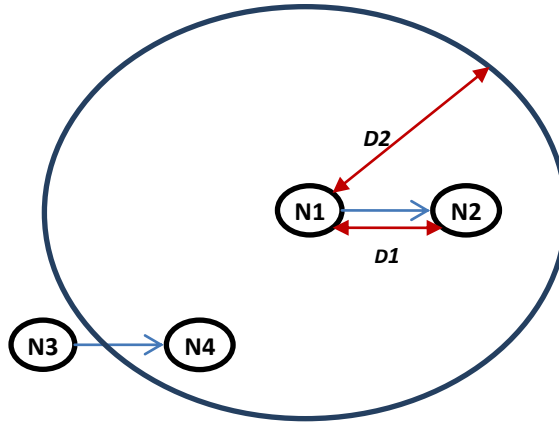


Figure 5: Interference Distance

This equation has been used in many research papers [10]. Bringing scheduling into account we derive Eq. 3 from Eq. 2.

$$SINR = \frac{Pr}{N + \sum_j \alpha_j \times I_j} \quad Eq. 3$$

where α_j is an interference factor that in our case can be zero or one. If the nodes that cause interference are in the same GORby-Group as the main sender, this factor is equal to one; otherwise it is zero. We will use some examples to explain this factor more.

Let us consider topology shown in Figure 5 for this discussion. In this example, Node_1 is sending data to Node_2 (Link is up and running), which Node_2 is one hop away (D1) from Node_1. Then, Node_3 wants to send a data to Node_4. Based on the most of older researches, Node_1 transmission causes only interference on other links transmission up to D2 distance away, which:

$$D_2 = 2.2 \times D_1 \quad Eq.4$$

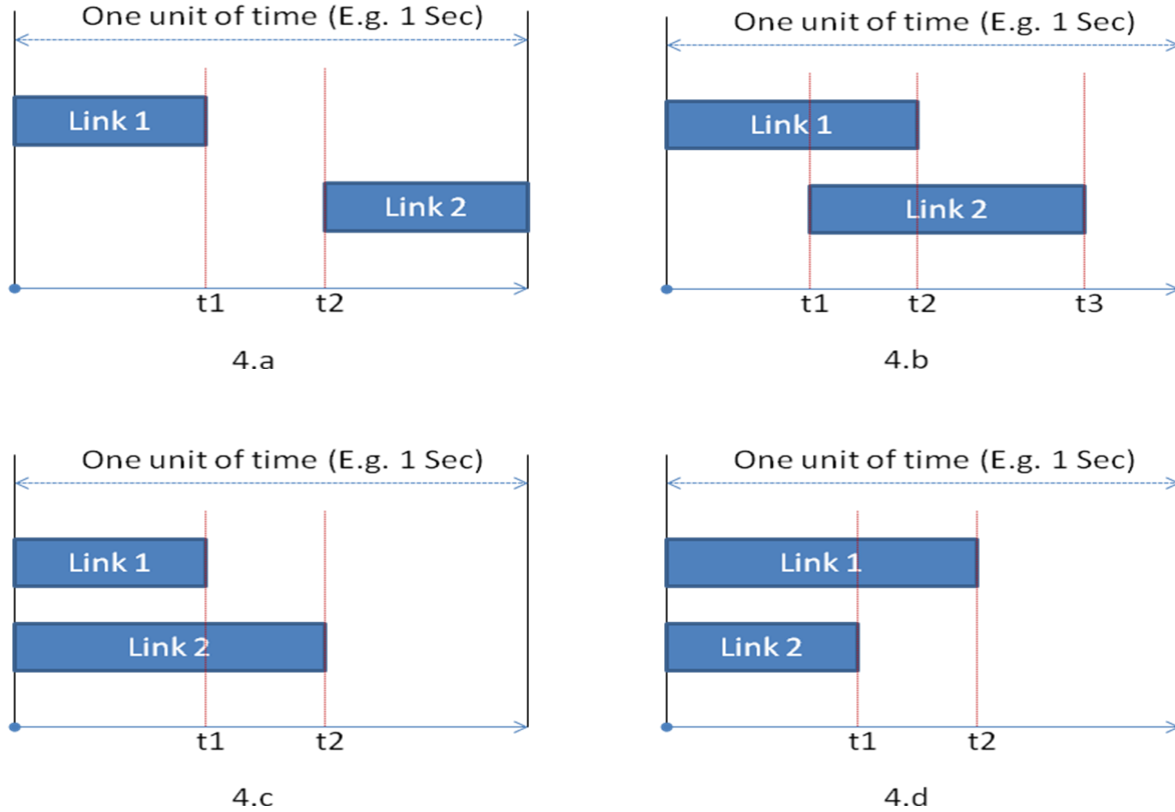


Figure 6: Interference Factor

Recently, other research challenged this assumption value to be more than 2.2 times of D1 [10], in fact if the cumulative interference is considered, the distance can be more than D2. Back to our example, if we assume all of these nodes have 802.11 based MAC layer and because the distance from Node_4 to Node_1 is more than D1, Node_2's CTS packet cannot be detected by Node_4. So, Node_4 answers Node_3 RTS and Node_3 sends the data, which causes having interference of Node_1 transferring to Node_2 over receiving of Node_4 from Node_3.

Meaning, because they are in the same media and frequency channel for transferring, granting media access by Node_1 does not prevent Node_3 transmission. Indeed, transmitting data on one link causes various degrees of interference in all the links in the network. So, in calculating SINR in one node, the summation these low interferences may be high enough to make SINR less than β threshold [10]. This interference will be directly in Eq.2 calculation; however, it will be in Eq.3 including the factor of α .

Figure 6 illustrates four scenarios of the scheduling result in different interference factors. Let us assume time unit is one second (one slot). These scenarios are based on the topology of Figure 5. To simplify the description, we make the following assumptions:

- Links are in the same media; however, far enough to not be able to detect CTS (received power is less than threshold). So, accessing the media by one link does not prevent accessing media by the other one (cannot receive RTS or CTS).
- Link2: Node_3 is a transmitter and Node_4 is the receiver (main transmission).
- Link1: Node_1 is a transmitter and Node_2 is the receiver (source of interference on Link2).

Four possible scenarios in this example:

- Case 4.a:
 - In this scenario, interference factor is zero; Link1 does not cause any interference on Link2 because they are not active at the same time:

$$t_2 \geq t_1 \rightarrow \alpha = 0$$

- Case 4.b:

- In this scenario, interference factor is not zero and is being calculated as following, because just between t_1 and t_2 Link2 receives signal and interference power at the same time:

$$t_3 \geq t_2 \geq t_1 \rightarrow \alpha = \frac{t_2 - t_1}{t_2} \rightarrow 1 > \alpha > 0$$

- Case 4.c:

- In this scenario, interference factor is one; Link1 activation affects fully the Link2 transmission:

$$t_2 \geq t_1 \rightarrow \alpha = 1$$

- Case 4.d:

- In this situation, interference factor is between zero and one; Link1 activation affects partially the Link2 transmission. It is because between t_1 and t_2 , Link2 does not receive any interference based on the figure:

$$t_2 \geq t_1 \rightarrow \alpha = \frac{t_1}{t_2}$$

Above example behaviours are valid in any wireless network; however, in case of Multicast, calculation of an interference factor is straight forward. As we discussed before and in this research, we assume Multicast network uses the full bandwidth, and main concern is calculation of the Multicast traffic effect (interference) on a new member at the joining time. In this case, because nodes belong to each GORby-Group is at the same scheduling state, interference factor is either one or zero:

- If a node becomes part of the Green GORby-Group: At the time slot that it is in the REC1 State, only Red GORby-Group nodes are in the Transmit State. So, interference factor of the Green and Orange GORby-Groups are zero, and interference factor of the Red GORby-Group's nodes is one.

- If node becomes part of the Orange GORby-Group: At the time slot that it is in the REC1 State, only Green subset nodes are in the Transmit State. So, interference factor of the Orange and Red GORby-Groups are zero, and interference factor of the Green GORby-Group's nodes is one.
- If node becomes part of the Red GORby-Group: At the time slot that it is in the REC1 State, only Orange GORby-Group nodes are in the Transmit State. So, interference factor of the Orange and Red GORby-Groups are zero, and interference factor of the Orange GORby-Group's nodes is one.

Up to this point, we introduced our interference calculation, interference factor and SINR calculation. In the next subsection, we will introduce the concept of Multicast SINR.

3.4.3 Multicast SINR Calculation

Based on the above mentioned definitions, here we introduce the new metric to be the basic metric of our Interference-Aware Multicast algorithm.

Let us assume the previous nodes (Node_a and Node_b) in our network, which Node_b is in the receiving state. Also there are other nodes in the Multicast group that cause interference on the Node_b. Based on the Eq.3 and by replacing “Pr” and “I(j)” from the Eq.1, we can compute the SINR at Node_b:

$$SINR = \frac{K_a \times \frac{Pt_a}{D_a^{n_a}}}{N + \sum_j \alpha_j \times K_j \times \frac{Pt_j}{D_j^{n_j}}} \quad Eq. 5$$

where for any value of “i”:

- K_i is the K factor of Node_b and Node_i link environment.
- Pt_i is the Transmit Power of Node_i.

- D_i is the distance between Node_b and Node_i.
- n_i is the n factor for Node_b and Node_i link.

Let us assume there is no noise in the environment and have the main concern just Interference ($N = 0$). Also all of the nodes use the same power of transmit ($P_{t_i} = P_t$, for all “i”), and environment condition is the same for entire nodes, including antenna gain and path loss ($n_i = n$ & $K_i = K$, for all “i”). To make the calculation simpler, we assumed “n” to be two. This value is arbitrary and does not affect the validity of our algorithm. So, based on this assumption, we have

$$SINR = \frac{K \times \frac{P_t}{D_a^2}}{\sum_j \alpha_j \times K \times \frac{P_t}{D_j^2}} = \frac{K \times P_t \times \frac{1}{D_a^2}}{K \times P_t \times \sum_j \alpha_j \times \frac{1}{D_j^2}} = \frac{\frac{1}{D_a^2}}{\sum_j \alpha_j \times \frac{1}{D_j^2}} \quad Eq.6$$

Moreover, based on the definition of “ α ” in the section “3.4.2” (which is zero or one in Multicast network), we define the new metric called GIA (GORby Interference-Aware join metric) at Node_b as below:

$$GIA = \frac{\frac{1}{D_a^2}}{\sum_j \frac{1}{D_j^2}}, \forall j \in A_g \text{ \& } j \neq a \quad Eq.7$$

In this formula A_g is the total number of nodes of the network belonging to the same GORby-Group (same colour) of Node_a. Node_b can calculate GIA by using the location information of the nodes.

3.5 Interference-Aware join Algorithm

In order to define the algorithm, we need to describe some elements, which following sections are about these required and related elements.

3.5.1 Physical location information

This algorithm uses the physical location information to calculate the distance between nodes. Nowadays, most of the mobile and wireless nodes have GPS to find out their location, and when it comes being inside a building, local servers provide this information to the nodes. Anyhow, having physical location information is given for this algorithm, and providing solution for this information is out of our research scope.

3.5.2 Unicast Routing Protocol

This algorithm and Multicast protocol works with any unicast routing protocols. All we need to have is nodes being able to communicate in unicast mode.

3.5.3 Node GORby Multicast db tables

A) Multicast node keeps a Multicast table per Multicast group if the node is a member of the GORby-Group, called GMI (GORby Multicast Information). The table contains the information that is critical for the correct execution of the algorithm. This information gets distributed to the other Multicast neighbouring nodes as part of the hello packet that is detailed at the end of this section periodically (e.g. every 30 seconds). The content of the table is listed below:

- Status:
 - ORG_0 = zero: Expired information.
 - ORG_1 = one: Set to generator; which means this node has generated this information.

- ORG_2 = two: Upstream node information.
- ORG_3 = three: Other valid nodes information.
- TTL (Time to live): Timestamp of generating this information:
 - If difference between this time and the current time is more than Valid_Time (1200 seconds, which can be set to any number), change status flag to ORG_0 (expired record).
- U_MG_S and D_MG_R flags: We will describe them in the section 3.5.4.
- Location information: Based on GPS (section 3.4.3).

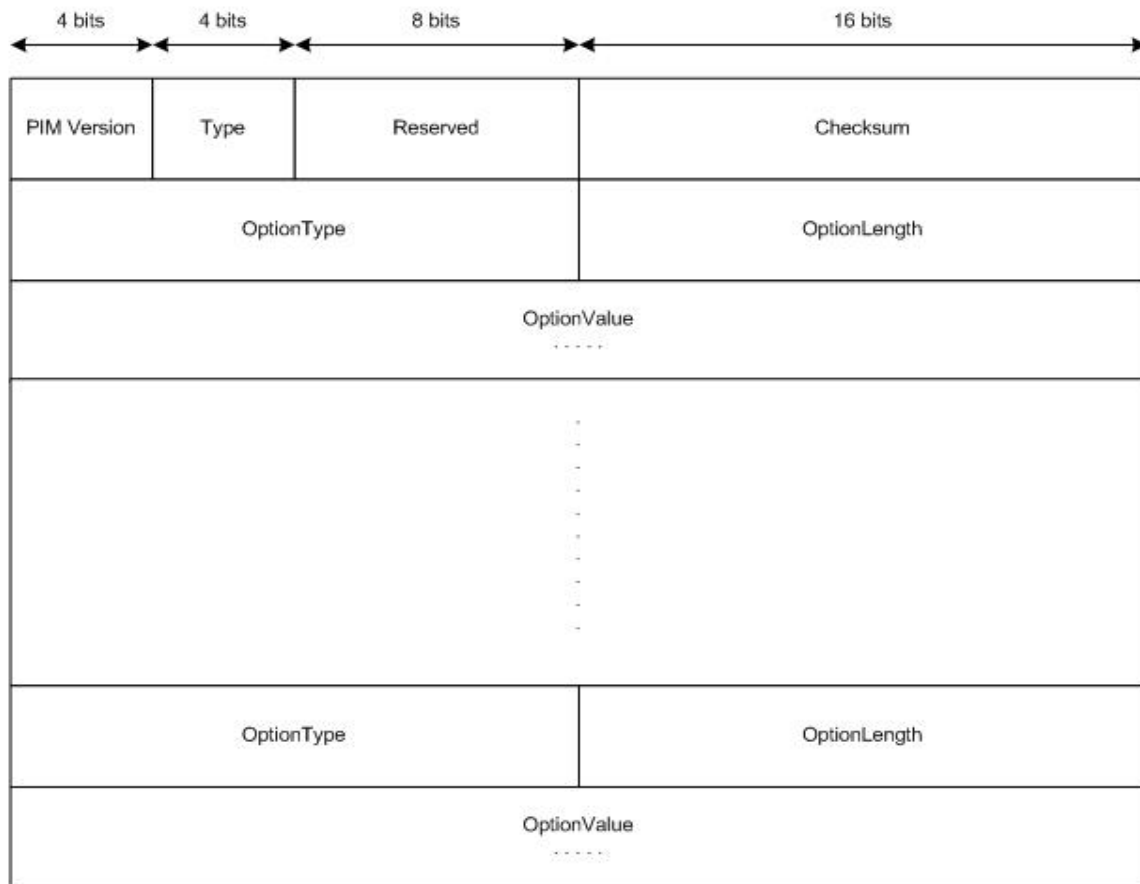


Figure 7: PIM Hello packet format

- Multicast hope count (MHC): Distance to the RP via Multicast tree. MHC is used to set the node colour based on the following logics:
 - RP is the first node of the tree, thus MHC is zero. If RP is always set to Green, then $MHC=0 \rightarrow \text{Green}$;
 - Based on the above argument, $MHC=1 \rightarrow \text{Orange}$ and $MHC=2 \rightarrow \text{Red}$;
 - In general, for a Given MHC, the color can be derived using the following formula:

$$\text{MHC Mod } 3 = \begin{cases} 0 \rightarrow \text{Green} \\ 1 \rightarrow \text{Orange} \\ 2 \rightarrow \text{Red} \end{cases}$$

B) Currently PIM hello packet (Figure 7) has used options presented by Table 3. In our design, we modify hello packet to transfer node GMI table information in order to synchronize them between the nodes. We use option three (one of the reserved options between 3 – 16) as formatted in option value as below:

- a. Option Type: 3.
- b. Option Length: 8.
- c. Option Value:
 - i. Status flag: 2 bits.
 - ii. MHC: 4 bits.
 - iii. Reserved bites: 2 bites for join algorithm procedure flags (D_MG_R and U_MG_S).
 - iv. TTL: One byte.
 - v. Position: Six bytes
 - 1. Latitude (or x position): Two bytes.
 - 2. Longitude (or y position): Two bytes.
 - 3. Altitude (or z position): Two bytes.

Table 3 : PIM hello packet options

Option Type	Option Description
1	Holdtime
2	Lan Prune Delay
3 – 16	Reserved to be defined in future versions of this document
18	Deprecated and should not be used
19	DR Priority
20	Generation ID
24	Address List

C) To prepare the above mentioned option value, which is in fact the GMI table record as well, we follow this procedure:

a. First node that joins the group is RP, so:

- i. Status flag: ORG_1 (one).
- ii. MHC: Green (zero).

It also adds the location information and the current time to prepare the record.

b. Nodes add first record as own information to the GMI table when they join the group:

- i. Status flag: ORG_1 (one).
- ii. MHC: Upstream.MHC + 1.

They also add the location information and the current time to prepare the record.

c. Node that receives this information in hello packet:

- i. Checks if it has already this information in GMI table, and refreshes this record if the TTL of the received packet is more up-to-date than current entry/record TTL.
- ii. Check sender to see if it is upstream node:

1. Yes → Status flag = ORG_2.
2. No → Status flag = ORG_3.

d. Node maintains the record status versus TTL:

- i. Compares value of {Current time – TTL} with Valid_Time (e.g. 1200 seconds); If it is:
 - 1. Less: Valid record, move to the next one.
 - 2. More: Status flag = ORG_0 (expired data).
 - 3. If the upstream node get expired = disconnected from the group and has to initiate a join procedure.
 - e. Node that wants to send hello packet:
 - i. Starts from the first record (own info):
 - 1. Change this record TTL to the current time before sending.
 - ii. Sends the next valid record each time on the sending record.
 - iii. End of GMI: Go to first record.
- D) Following to the previous section, nodes receive the GMI table from any member at the time of joining group. In the first time receiving, node maintains the records by using the above mentioned procedure (C) in hello packet.
- E) Setting Valid_Time depends on how many hop away nodes information are required. Based on our design and for example, if we set hello packets to be sent every 30 seconds, and in average each node keeps the record of up to the ten nodes (to cover up to four hops away), we need to set Valid_Time to be more than: $30 * 10 * 4 = 1200$ seconds. More about this calculation and discussion to develop is coming in the chapter five. Also in an environment with high changes (join and leave, or movement), we need to send hello packet more often.

3.5.4 Join Algorithm definition

This Multicast protocol is a core/tree based and receiver initiated one. Meaning there is at least one core per group. A sender sends information capsulated in a unicast packet to the core, and then core Multicasts it to all of the members. For sending information to the core, it uses OLSR (for instance) as unicast routing protocol to reach the tree core. Assumption is receivers (nodes)

know which group they want to join. Joining Multicast group is based on the following procedure:

- (i) A new member broadcast RREQ including group ID.
- (ii) If the node that receives the RREQ is this specified group Multicast member, goes to step “iv”, otherwise it follows step “iii”.
- (iii) It keeps new member node requester node ID as a potential receiver by setting D_MG_R (Downstream Multicast Group Receiver) flag to “POTENTIAL” (= zero), broadcasts RREQ with own node ID, and goes to step “ii”.
- (iv) Multicast member that receives the RREQ:
 - a. Enter the new receiver (relay or end member) node ID in GMI table of group ID and as a potential receiver by setting D_MG_R flag to “POTENTIAL”.
 - b. Prepare RREP (unicast capsulated) including:
 - i. Its GORby colour reference number (MHC).
 - ii. Its node ID.
 - iii. Set PGIA to infinity.
- (v) Node calculates its distance to the each node information in its GMI table of this group ID that the node wants to join.
- (vi) Then it calculates the GIA based on the information calculated in step “v” per each option (scenario):
 - a. It calculates its GIA parameter for this colour (Eq. 7).
 - b. It compares its GIA calculated in previous step (v) with the GIA received in the step “iv”, and set the PGIA value to the minimum of them:

$$PGIA = \min \{PGIA, GIA\}$$

- c. If PGIA is more than threshold (β), it set U_MG_S (Upstream Multicast Group Sender) flag to “POTENTIAL”.
 - It set its potential colour to be one after the received colour information in the step “iv”.
 - Goes to the step “vii”.
- d. If the PGIA value is less than the threshold (β), it set U_MG_S to “CANCEL”,
 - i. To cancel it, just remove it from GMI table or set the status to ORG_0.
 - ii. Goes to the step “viii”.
- (vii) Then it calculates the maximum PGIA and selects the path as final path:
 - a. If it finds more than one option, chooses the path that provides less hop count to the tree:
 - i. It joins the group and update its Multicast table including its colour.
 - ii. It sets U_MG_S flag to “FIRM” (= one).
 - iii. It informs the upstream node that is the next hop on this path.
 - iv. Then it sends GMI table information via hello packet to the others as we described it in 3.6.3 section.
 - b. The upstream node join the Multicast group:
 - i. Set D_MG_R flag to “FIRM”.
 - ii. If the upstream node is a Multicast member already, it goes to step “viii”.
 - iii. It joins the group and updates its GMI table including its MHC (for colour).
 - iv. It sets U_MG_S flag to “FIRM”.

- v. It informs the upstream node that is the next hop on this path.
- vi. Then it sends the GMI table information via the hello packet described in the section “3.5.3” neighbours.
- c. The upstream nodes along the path follow step “vii-b” until we reach the Multicast member (It already has U_MG_S for a member set to “FIRM”).
- (viii) End of the algorithm.

This information needs to be refreshed periodically (for instance every three seconds), because of nodes mobility and topology changes if any, which our algorithm processing hello packet provided in the section 3.5.3 maintains the GMI table. This maintenance may cause joining group again, which would make sure model is following and is aware of changes in network.

In chapter five and as future potential work to adjust this algorithm for mobile environment with high movement, we suggest to run joining more often instead of sending hello packets, or even a mix of these two models. However, if we implement this model for a stale environment like WMN, hello packet would be enough.

3.6 Cases of Interference-Aware join

In order to illustrate joining algorithm to join a Multicast group based on the GORby definition, we use the following cases as some examples. Our examples cover joining with one, two and three hop/s away distance from Multicast tree. We defined the detail of our algorithm in the previous section (3.5), and in this section we present how algorithm works in these cases.

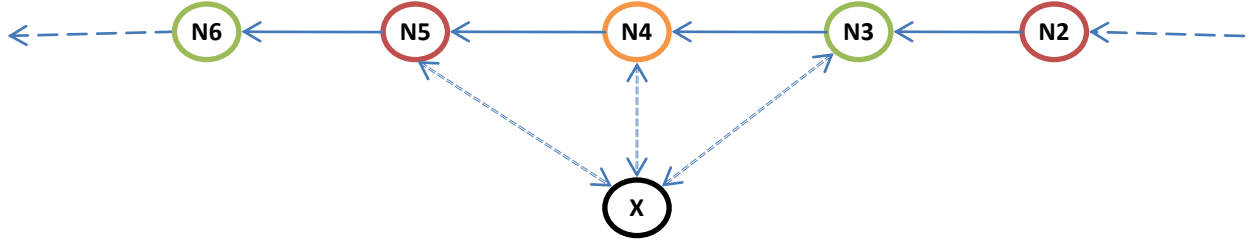


Figure 8: Joining Multicast group with one hop away

3.6.1 Case_1: Joining with one hop away

Figure 8 shows a topology example of Case_1. In this case, node X wants to join the group, broadcasts a join request to other nodes.

- (i) Node X broadcasts RREQ including group ID that wants to join;
 - It also fires a timer to receive response/s, and waits until end of this timer. More detail about timer is in the previous section (3.5).
- (ii) Three nodes N3 (Green), N4 (Orange) and N5 (Red) receive RREQ, replay back by sending RREP and include their own colour in RREP packet.
- (iii) Node X requests GORby Multicast routing table information (that has nodes' location and colour) of the group from the first node that answers the query.
 - Detail information about this table is in the previous section (3.5).
- (iv) Node X calculates GIA for each join (based on above three nodes proposals) options by using Eq.7.
- (v) Then, it selects the node that has the best/most GIA to join Multicast group.
- (vi) In case of having equal GIA, node X follows unicast protocol metric algorithm, which is the minimum cost to source (or core node).

Note: It may face that highest GIA is less than SINR threshold (β), meaning it does not have any option to join the group based on the current network situation.

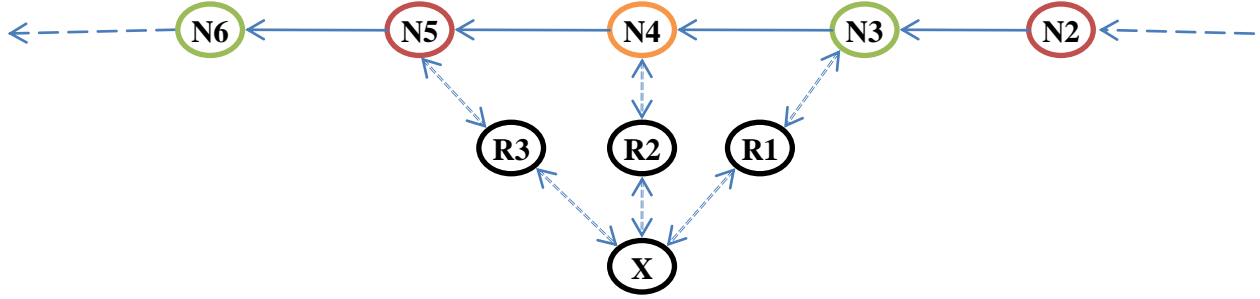


Figure 9: Joining Multicast group with two hops away

3.6.2 Case_2: Joining with two hops away

Figure 9 shows a topology example of Case_2. In this case, node X wants to join the group:

- (i) Node X broadcasts RREQ including group ID that wants to join.
 - It also fires a timer to receive response/s, and waits until end of this timer.
- (ii) RREQ is received by R1, R2 and R3, which are not part of the Multicast group yet. So, these three nodes (R1, R2 & R3) send/broadcast the RREQ to the next hops (in this example, N3, N4 and N5).
- (iii) Multicast members replay back RREP to R1, R2 and R3 (including their own colour).
- (iv) R1, R2 and R3 calculate appropriate GIA after receiving the group nodes' location information, based on join nodes options.
 - They follow the join procedure described in the previous case (Case_1) to join the Multicast group (potentially).

- (v) These three relay nodes, replay back to node X including their own potential colour (if they join the Multicast group) as well as their GIA. For instance:
 - a. In case of R1: It calculates GIA based on receiving information from a Green node (N3), and its colour will be Orange if node X chooses this path to reach the Multicast group.
 - b. In case of R2: It calculates GIA based on receiving information from an Orange node (N4), and its colour will be Red if node X chooses this path to reach the Multicast group.
 - c. In case of R3: It calculates GIA based on receiving information from a Red node (N5), and its colour will be Green if node X chooses this path to reach the Multicast group.
- (vi) Node X has three options (assume receive all RREQ response), each with different colors. So, it calculates its GIAs based on these three colours (as per Case_1).
- (vii) Then it goes to next phase that is calculating path GIA (PGIA) by the following Max-Min algorithm:
 - a. For each path, it calculates the minimum GIA of the nodes based on the received information and local calculation of itself, and sets it as PGIA.
 - b. Then it selects the path with the maximum PGIA value.
- (viii) In case of equal PGIA, it selects the path with the maximum local node X GIA itself (Meaning node X to nodes of R1, R2 and R3).
- (ix) In case of having equal GIA, node X follows unicast protocol metric algorithm, which is the minimum cost to source (or core node).

Note: Just like the previous section, the algorithm may end with no option.

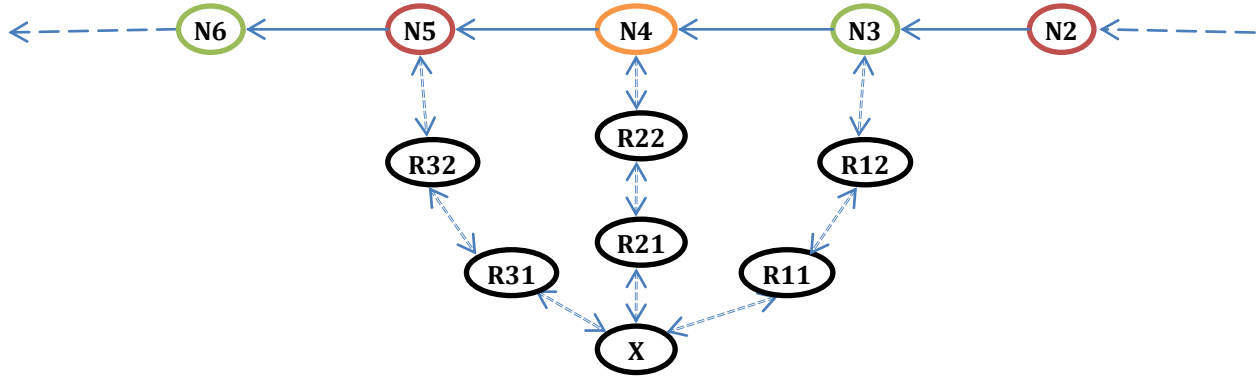


Figure 10: Joining Multicast group with three hops away (three join options)

3.6.3 Case_3: Joining with three hops away (three join options)

Figure 10 shows a topology example of Case_3. In this case, node X wants to join the group:

- (i) Node X broadcasts RREQ including group ID that wants to join.
 - It also fires a timer to receive response/s, and wait until end of this timer.
- (ii) RREQ reaches R11, R21 and R31 that are not Multicast group member neither. So, these three nodes (R11, R21 & R31) send/broadcast an RREQ to the next hops (in this example, R12, R22 and R32).
- (iii) RREQ reaches R12, R22 and R32 that are not Multicast group members neither. So, these three nodes (R12, R22 & R32) send/broadcast a new RREQ to the next hops (in this example, N3, N4 and N5).
 - Following previous case steps (Case_2).
- (iv) Multicast members replay back by sending an RREP to R12, R22 and R32 (including their own colour).
- (v) R12, R22 and R32 calculate appropriate GIA, base on the join nodes option.

- Follow Case_1 steps.
- (vi) They replay back to downstream nodes (R11, R21 and R31) their potential colour as well as GIA.
- (vii) These relay nodes (R11, R21 and R31) calculate appropriate GIA, base on the received information from relay nodes R12, R22 and R32 (If any);
- (viii) Then, they follow previous case steps (Case_2) to calculate the appropriate PGIA (Min GIA).
- (ix) These three relay nodes, replay back to node X including their potential colour (if they join the Multicast group) as well as their PGIA path.
- (x) Node X has three options now, which in each option has a different colour too. So, it follows same procedure to calculate its GIA based on these three colours.
- (xi) Then it goes to the next phase that is calculating PGIA;
 - Same as previous case (Case_2).
- (xii) It chooses the path with the highest PGIA value (Max-Min algorithm).
- (xiii) In case of having the equal PGIA, it selects the path with highest GIA for node X itself (Meaning node X to nodes of R1, R2 and R3).
- (xiv) In case of having the equal GIA, node X follows unicast protocol metric algorithm, which is the minimum cost to source (or core node).

Note: Moreover and similar to the previous case option, it may end with no option. Meaning it does not have any option to join the group by this situation.

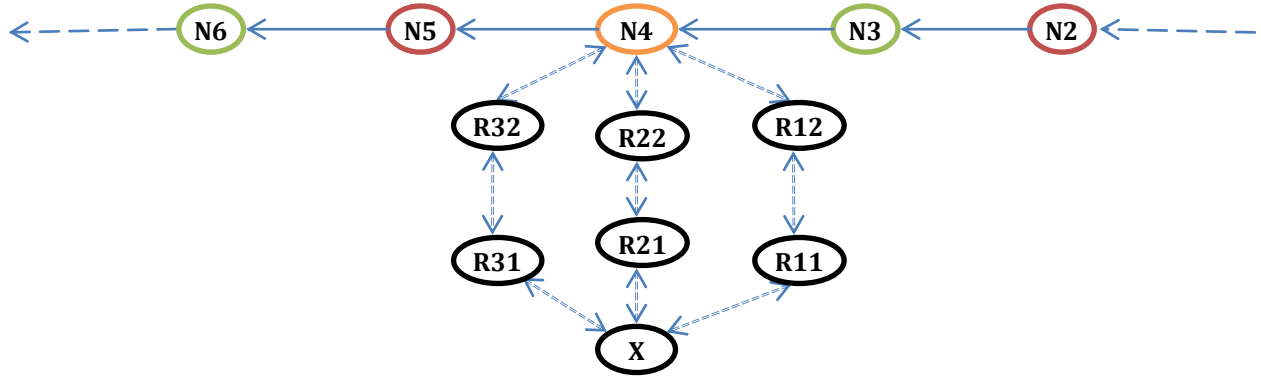


Figure 11: Joining Multicast group with three hops away (same join option)

3.6.4 Case_4 Joining with three hops away (same join option)

Figure 11 shows a topology example of Case_4. In this case, node X wants to join the group and most of steps are the same as Case_3 from algorithm point of view. The only difference comparing to previous case is that node X at the end will have just one colour option. For instance as it shown in Figure 11, it will be just an Orange node, which does not affect algorithm. So, it does not make any difference from algorithm point of view between this case and the previous one (Case_3).

Note: Similar to previous cases options, it may end with no option. Meaning it does not have any option to join the group by this situation.

3.6.5 Case_5 hybrid joining options (same and different join options)

Following to the previous cases and based on described model up to this point, it does not make any difference to be one hop away, two or more hop away. Also, same join point or multiple points does not need special step. A case of having a mix of these models (competing one hop with two and three hops away, and multiple and same join option) follows the same algorithm. As long as each node calculates local GIA parameter, compares it with the receiving one,

calculates PGIA up to this point, and forwards to the downstream nodes, it should be fine. At the end, receiver decides about the path based on the PGIA metrics.

Chapter 4

Simulation and Results

In this chapter we evaluate and analyze our algorithm from different points of view. In order to accomplish this evaluation, we used OPNET v15.0 and studied two Multicast routing protocols performance. The first one is the original PIM-SM [2] including all standard features. The second one is our proposed protocol, which is a modified version of PIM-SM. In the rest of this document, we use the term “GORby” for this modified version of PIM-SM.

We used a basic topology shown in Figure 12 for our analysis. The basic topology and the setup are modified as per study. We compare the performances between GORby and PIM-SM by studying performance measures such as instant network packet dropped throughput (received data by receivers), delay and convergence. We also analyze impact of different traffic loads on this algorithm.

In section 4.1 we present our overall topology and the network configuration in our evaluation and analysis. Section 4.2 describes our algorithm performance evaluation in a serial join such that the receivers join the group in different times. In section 4.3 we study our algorithm performance for different traffic flows and loads (for serial join). Section 4.4 of this chapter deals with parallel join (seven receivers at the same time) situation and we study convergence (activity and delay) of the group. We perform parallel join analysis in different traffic loads as well. At

the end of this chapter and in section 4.5, we bring all of our tests and analysis results into account and present an overall performance evaluation of our proposed algorithm.

In this chapter, we have two series of reports. First series are generated directly by OPNET and we present them in our main part of the document in this chapter (like Figure 18), which are OPNET default and pre-developed reports format. For the second series of report (like Figure 25 and Figure 26), we extracted the detail data of OPNET simulation run results (for instance for different traffic amounts), add related extra step (like taking the average of them) and then generated the new bar chart report. Appendix B presents some of the OPNET default report that we have used their detail data to prepare our second series of the report in this chapter.

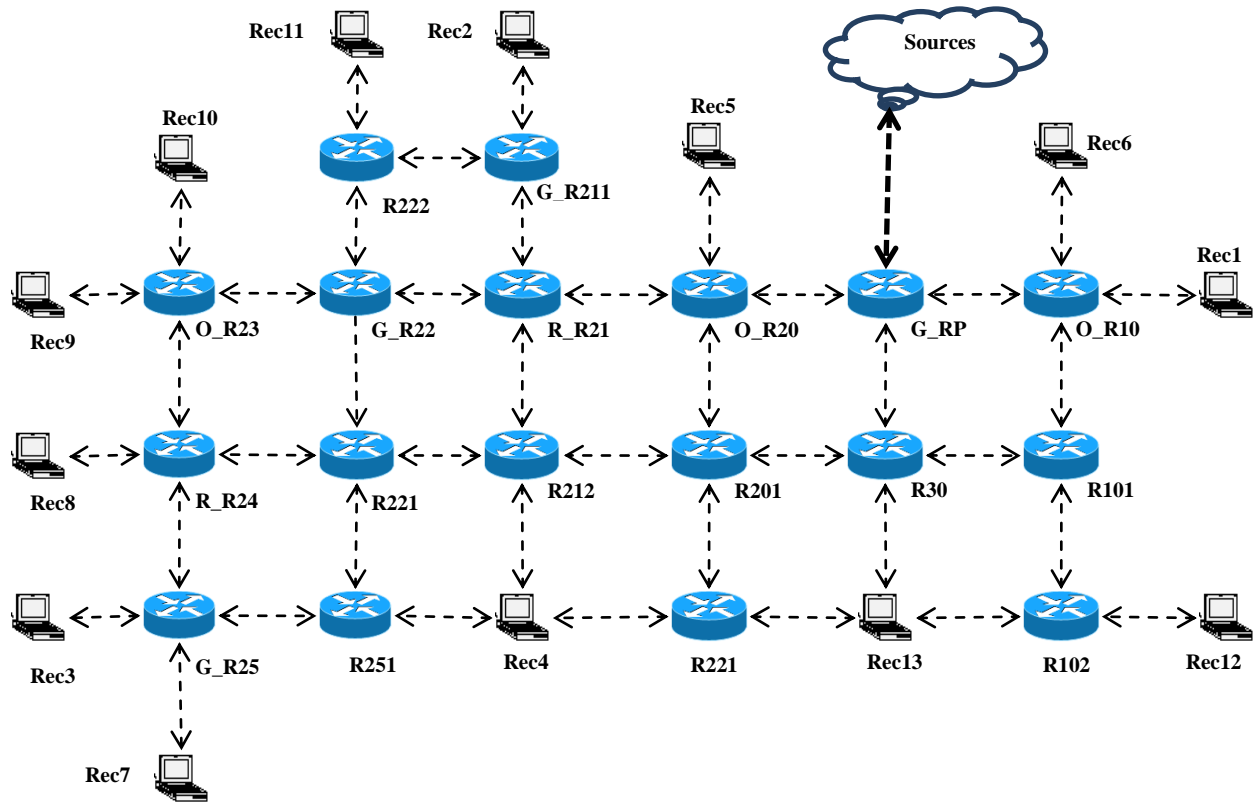


Figure 12: Sample network topology

4.1. Simulation setup

In this evaluation, we use 802.11 nodes, and since we just use single Multicast group with one traffic stream, (which is also one direction,) we assume MAC layer in this situation achieves perfect Schedule. The reason is, each node mainly receives Multicast packets from the upstream node and passes them to the downstream node(s). Also, compared to the controlling data packet, Multicast traffic is high enough to be assumed to be the main traffic [18] and the impact of the controlling packet is negligible.

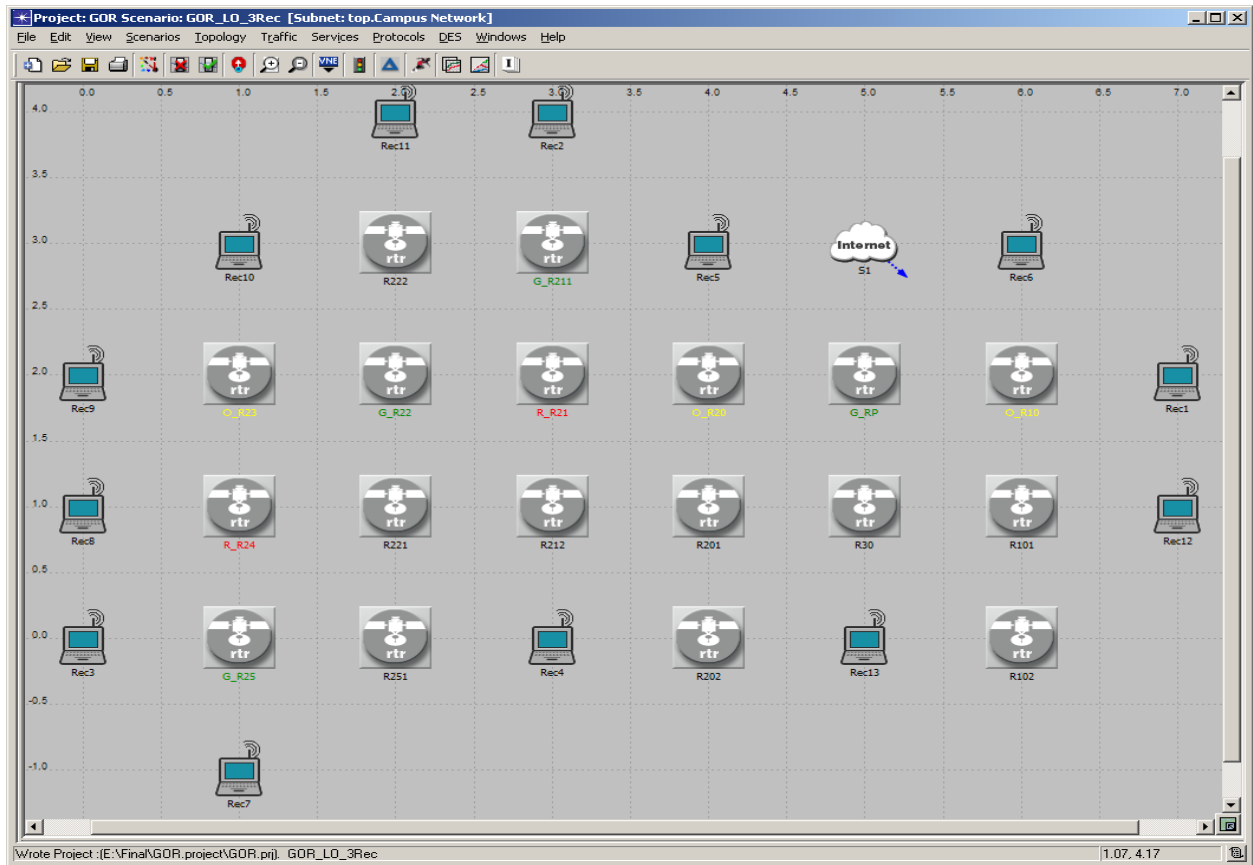


Figure 13: Topology implemented in OPNET

Figure 13 shows the network topology of Figure 12 implemented in OPNET. We mainly used this setup in our evaluation for sections 4.2, 4.3 and 4.4, which is described in detail in each section. Based on our assumptions, Multicast sources are outside of the wireless network and the gateway is RP as the first node that the Multicast traffic enters the wireless network. The following is the common setup and configuration of the nodes (Receiver, Routers).

- **Physical and MAC layer:** It is a “Campus” topology, and distances are in a Km range. Transmit power is fixed for the entire nodes at twenty milliwatts (0.020 watt), and the “Packet Reception-Power Threshold” value for all of them is set to “-95” dbm. For buffer size, we used 1024 Kb that is an OPNET default value. Also, the nodes need to have sub-interfaces to be used per link. The physical layer standard is 802.11b (Direct Sequence) with a data rate of 11 Mbps for the entire nodes, which can be configured overall. All of the nodes’ access point functionality is disabled.
- **Unicast protocol:** We use OSPF in all of the nodes including receivers and routers in this model. To prevent hidden node issue (related to DR selecting process), nodes have to have sub-interfaces to be able to work in different Groups. We have to use this model because of the OSPF as well as PIM-SM (and GORby) protocols. PIM-SM uses DR in the acknowledge step, and just having one interface for nodes would make conflict on DR selection, and simulation would not be able to handle transferring the traffic.
- **Multicast protocol:** PIM-SM and IGMP are set to version two and GORby parameters are the same except the required modified steps. Rendezvous Node (core or RP) is statically selected and configured for all of the nodes as well as Multicast group address that they are going to join. Joining group start times are varied per node and the hello packet interval is 30 seconds with a holding time of 105 seconds. Rec1 joins in 85th Sec while Rec2 and Rec3 join in 105th Sec and 135th Sec respectively. Also, Rec4 to Rec10 join in 210th Sec when we need to use them.
- **Traffic:** Multicast traffics used are listed in Table 4, which includes low, medium and high traffic loads. We change the rate of traffic, and keep the average packet size (1000 Bytes = 250 Words) the same to present different kinds of traffic flow (Voice,

Multimedia, FTP and so on). We also use different seed values that affect packet sizes (as well as 802.11 random back off process), although the average packet size is a constant value. Multicast traffic coming to RP starts at 300th Sec in all of our scenarios.

In each of the following sections (4.2, 4.3 & 4.4) we require per analysis set up changes that are mentioned at the start of each section separately. We run the simulator for 30 minutes in each one of the scenarios. This means we have the traffic flow for at least 25 minutes.

Table 4: Traffic characteristics

Traffic	Symbol	Load (Kbps)	Load(pps)
Low 1	L1	800	100
Low 2	L2	8000	1000
Medium 1	M1	80,000	10,000
Medium 2	M2	320,000	40,000
High 1	H1	1,280,000	160,000
High 2	H2	5,120,000	640,000

We ran our simulator with different seed values (presented in Table 5). By default, we used a seed value of 128, although in some situations we ran the simulator with all values and then make an average. Based on these parameters, we define different scenarios, each one named by its unique set up. General scenarios naming, which are also presented in graphs, includes three sections in a format of X-Y_Z:

- X: Specifies Multicast protocol and seed value, based on Table 5.
- Y: Specifies the traffic, based on Table 4 “Symbol” column.
- Z: Specifies number of active receivers; for example 3R for three receivers, 4R for four receivers and 10R for ten receivers, and the rest of the receivers are off in each scenario.

For instance, G2-L2_4R means, Multicast routing protocol is GORby, there are four receivers active, the traffic load is a low level 2 (1000 Pack/Sec) and seed value is 67.

Table 5: Seed Values and Appropriate Symbols

Seed Value	GORby	PIM-SM
31	G1	P1
67	G2	P2
97	G3	P3
128	G (GOR)	P (PIM)

4.2. Serial join analysis

In this section, we just use Rec1, Rec2 and Rec3. In scenario names ending with 4R, Rec4 is active, while in scenario names ending with 3R, Rec4 is inactive. Also, the rest of the receivers are inactive in this section and analysis. As we described in the previous section, we scheduled members (receivers) to join the Multicast group in a serial order of 85th Sec for Rec1, 105th Sec for Rec2, 135th Sec for Rec3, and if applicable, 210th Sec for Rec4.

- **Rec1:** This is the first member to join and there is no other receiver for the traffic in the network yet. So, it is obvious that Rec1 will connect to the RP (and forms the tree) via one node (O_R10). As RP is Green, the relay node (O_R10) becomes Orange and Rec1 receives the Red colour (Figure 14). PIM-SM and GORby protocol would have the same result in this case. Based on our algorithm, as long as there is no other tree member, nodes would follow the shortest path (similar to standard PIM-SM).
- **Rec2:** This receiver has a similar situation to the Rec1 in our sample topology. In this case (same as Rec1), GORby follows the shortest path, same as PIM-SM. Having the first two members will shape a small tree, to be used in the next members join investigation. Subsequently, we will have four more nodes join with the GORby colours: O_R20 becomes Orange, R_R21 becomes Red, G_R211 becomes Green and Rec2 becomes Orange (Figure 15).

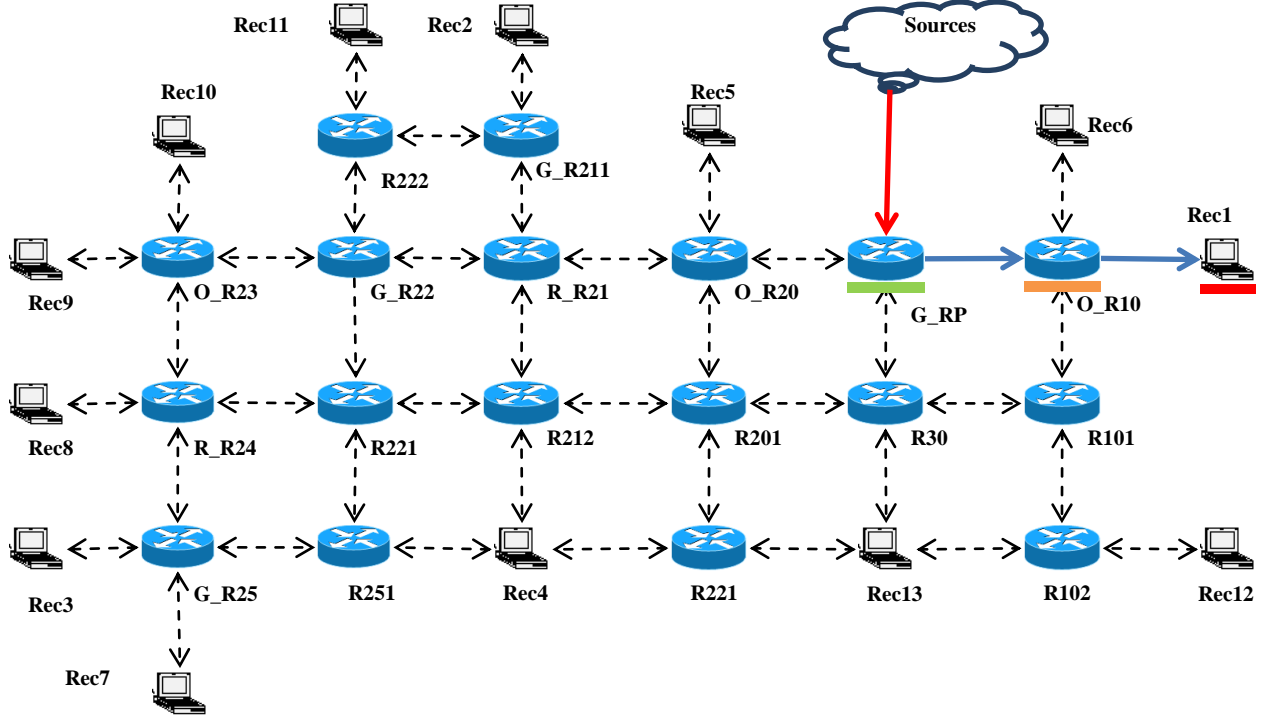


Figure 14: Rec1 joins the tree

- **Rec3:** In the next step, Rec3 joins the tree. Our topology shows this member has different options, some of which are presented in Figure 16 and we study five of them one-by-one:

- Path1: $R_R21 \rightarrow G_R22 \rightarrow O_R23 \rightarrow R_R24 \rightarrow G_R25 \rightarrow \text{Rec3}$
- Path2: $R_R21 \rightarrow G_R22 \rightarrow R221 \rightarrow R_R24 \rightarrow G_R25 \rightarrow \text{Rec3}$
- Path3: $R_R21 \rightarrow G_R22 \rightarrow R221 \rightarrow R251 \rightarrow G_R25 \rightarrow \text{Rec3}$
- Path4: $R_R21 \rightarrow R212 \rightarrow R221 \rightarrow R251 \rightarrow G_R25 \rightarrow \text{Rec3}$
- Path5: $R_R21 \rightarrow R212 \rightarrow R221 \rightarrow R_R24 \rightarrow G_R25 \rightarrow \text{Rec3}$

Table 6 shows Path1 calculation. In this table, in the first step G_R22 should join the tree via R_R21. R_R21 is already part of the tree, so PGIA of this node is infinite based on the GORby algorithm (already part of the tree). When G_R22 joins the tree via R_R21, it receives data from a Red node, but there is not any other Red node in the tree up to this point, so GIA of the node becomes infinite as well. As a result, PGIA of the path up to this point is infinite.

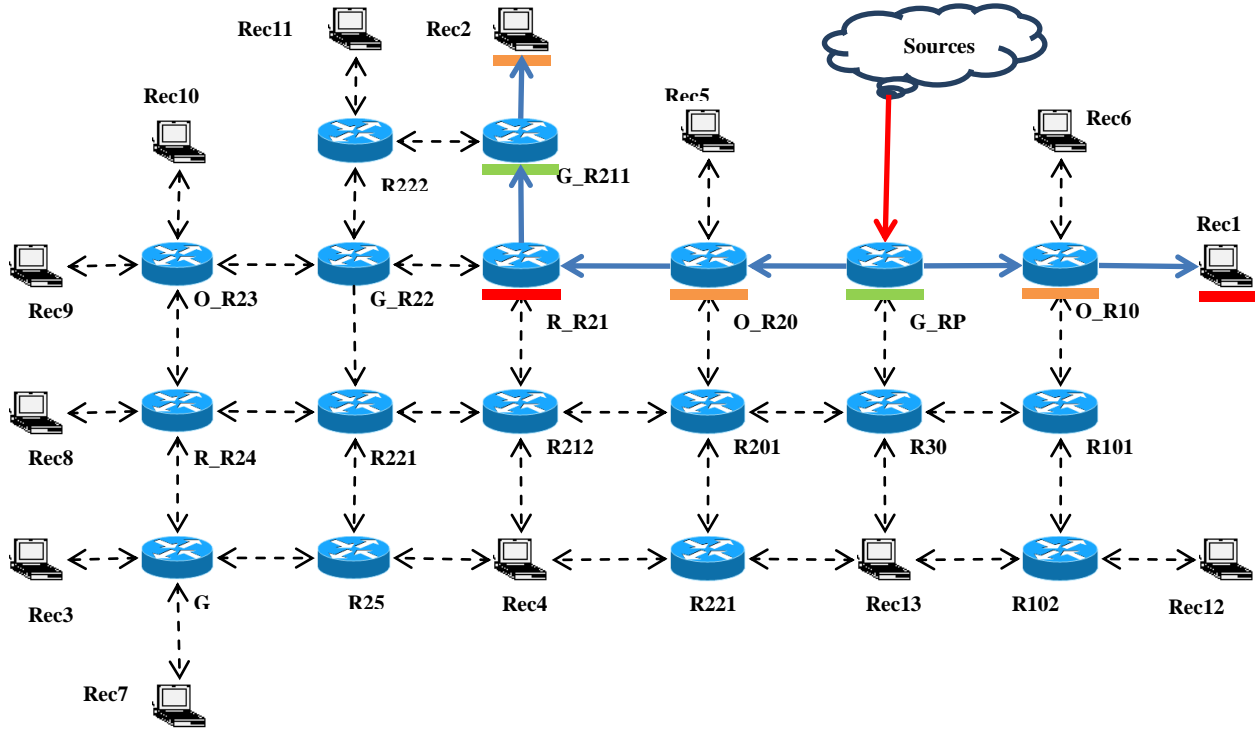


Figure 15: Rec2 joins the tree

In step two, O_R23 should join, which as it is shown in table, GIA of this node becomes 3.81 based on Eq.7 of section 3.4.3. O_R23 in position $(\{x, y\} = \{1, 2\})$ has an upstream node G_R22 in position $\{2, 2\}$, and is a green node. Other green GORby-Group members so far are G_R211 in $\{3, 3\}$ and G_RP in $\{5, 2\}$. The following steps show how GIA of this node can be calculated based on Eq.7 by having the nodes' known position information. The rest of the paths' nodes GIA calculation would be similar to this node calculation, which can be seen in the following example:

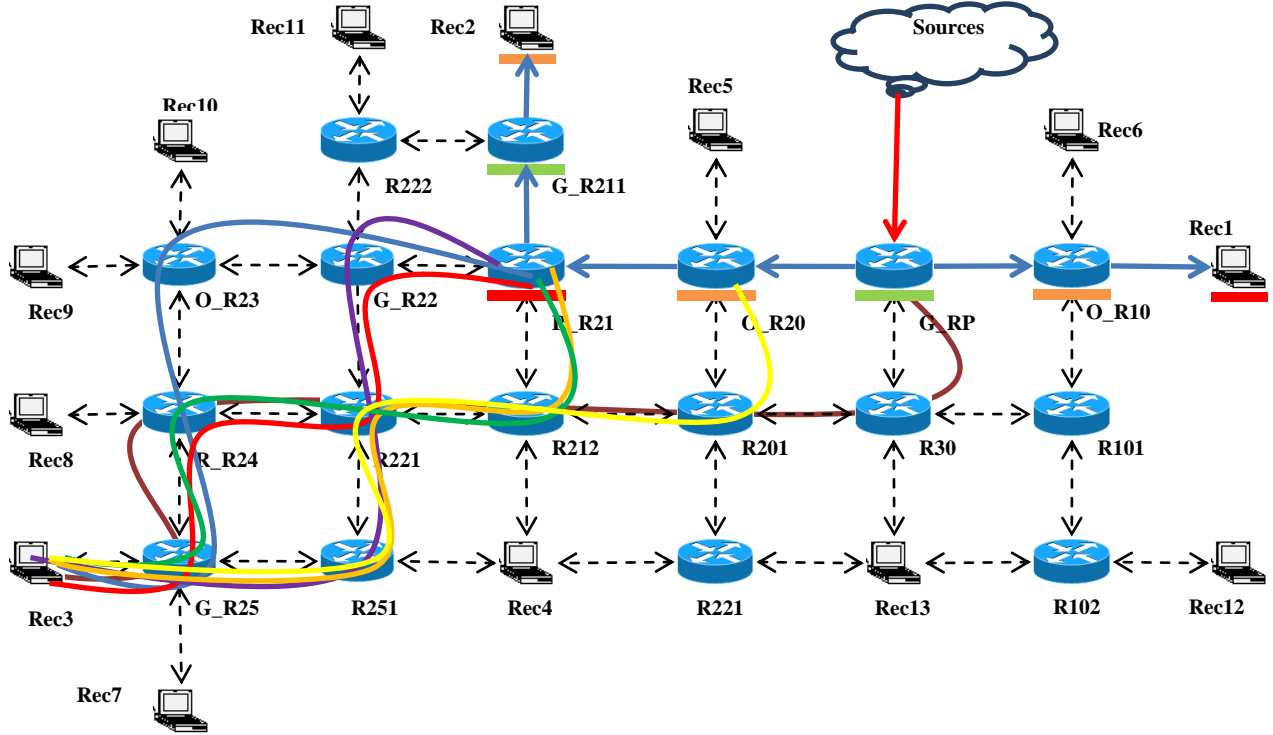


Figure 16: Path options for Rec3

$$\begin{aligned}
 GIA &= \frac{\frac{1}{D_a^2}}{\sum_j \frac{1}{D_j^2}}, \forall j \in A_g \text{ \& } j \neq a \\
 &= \frac{1}{\frac{(x_{O_R23} - x_{G_R22})^2 + (y_{O_R23} - y_{G_R22})^2}{1} + \frac{(x_{O_R23} - x_{G_R211})^2 + (y_{O_R23} - y_{G_R211})^2}{1} + \frac{(x_{O_R23} - x_{G_RP})^2 + (y_{O_R23} - y_{G_RP})^2}{1}} \\
 &= \frac{1}{\frac{(1-2)^2 + (2-2)^2}{1} + \frac{(1-3)^2 + (2-3)^2}{1} + \frac{(1-5)^2 + (2-2)^2}{1}} = \frac{1}{\frac{1}{4+1} + \frac{1}{16+0}} = \frac{1}{\frac{1}{5} + \frac{1}{16}} = \frac{1}{\frac{21}{80}} = \frac{80}{21} \\
 &\cong 3.81
 \end{aligned}$$

Interference-Aware Multicast in Wireless Mesh Network

Up to this point we have a PGIA of 3.81 (Minimum of infinite and 3.81). Other steps are the same as step two, until we reach Rec3 and this path ends with a PGIA of 3.81.

Table 6: Rec3 Path1 PGIA calculation

Step	Node	Upstream Node			Current Node Result		
		Name	Colour	PGIA	Colour	GIA	PGIA
One	G_R22	R_R21	Red	Infinite	Green	Infinite	Infinite
Two	O_R23	G_R22	Green	Infinite	Orange	3.81	3.81
Three	R_R24	O_R23	Orange	3.81	Red	7.22	3.81
Four	G_R25	R_R24	Red	3.81	Green	8	3.81
Five	Rec3	G_R25	Green	3.81	Orange	4.65	3.81

Table 7 includes a summary of Path2 calculation by GORby algorithm. The first step of this path is similar to Path1, and the second step calculates R221. At the end, Path2 will have PGIA of 3.33.

Table 7: Rec3 Path2 PGIA calculation

Step	Node	Upstream Node			Current Node Result		
		Name	Colour	PGIA	Colour	GIA	PGIA
One	G_R22	R_R21	Red	Infinite	Green	Infinite	Infinite
Two	R221	G_R22	Green	Infinite	Orange	3.33	3.33
Three	R_R24	R221	Orange	3.33	Red	7.22	3.33
Four	G_R25	R_R24	Red	3.33	Green	8	3.33
Five	Rec3	G_R25	Green	3.33	Orange	4.65	3.33

Table 8 shows Path3 calculation, in which Rec3 will have PGIA of 3.33 finally. Table 9 also shows Path4 calculation and how PGIA is 3.33.

Table 8: Rec3 Path3 PGIA calculation

Step	Node	Upstream Node			Current Node Result		
		Name	Colour	PGIA	Colour	GIA	PGIA
One	G_R22	R_R21	Red	Infinite	Green	Infinite	Infinite
Two	R221	G_R22	Green	Infinite	Orange	3.33	3.33
Three	R251	R221	Orange	3.33	Red	5.71	3.33
Four	G_R25	R251	Red	3.33	Green	8	3.33
Five	Rec3	G_R25	Green	3.33	Orange	4.65	3.33

Final path to study for Rec3 to join the tree is Path5, which Table 10 presents the related PGIA calculations. Based on these tables and the above four tables, just Path1 provides the maximum PGIA out of these five paths, which is 3.81. So, Rec3 chooses Path1 to join the tree.

Up to this point, we used algorithm steps and showed how Rec3 based on GORby calculates and chooses the best path to join the tree. In the next step, we will analyze the subsequent joining of Rec4.

Table 9: Rec3 Path4 PGIA calculation

Step	Node	Upstream Node			Current Node Result		
		Name	Colour	PGIA	Colour	GIA	PGIA
One	R212	R_R21	Red	Infinite	Green	Infinite	Infinite
Two	R221	R212	Green	Infinite	Orange	3.33	3.33
Three	R251	R221	Orange	3.33	Red	5.71	3.33
Four	G_R25	R251	Red	3.33	Green	8	3.33
Five	Rec3	G_R25	Green	3.33	Orange	5.26	3.33

- **Rec4:** After Rec3, if Rec4 joins the tree, our topology shows this member has different options as well, some of which are presented in Figure 17 and we will analyze them:

- Path1: R_R21 → R212 → Rec4
- Path2: G_R25 → R251 → Rec4

- Path3: G_R22 → R221 → R251 → Rec4
- Path4: R_R24 → R221 → R251 → Rec4
- Path5: R_R24 → R221 → R212 → Rec4
- Path6: O_R20 → R201 → R202 → Rec4
- Path7: G_RP → R30 → R201 → R202 → Rec4
- Path8: O_R10 → R101 → R30 → R201 → R202 → Rec4

Table 10: Rec3 Path5 PGIA calculation

Step	Node	Upstream Node			Current Node Result		
		Name	Colour	PGIA	Colour	GIA	PGIA
One	R212	R_R21	Red	Infinite	Green	Infinite	Infinite
Two	R221	R212	Green	Infinite	Orange	3.33	3.33
Three	R_R24	R221	Orange	3.33	Red	7.2	3.33
Four	G_R25	R251	Red	3.33	Green	8	3.33
Five	Rec3	G_R25	Green	3.33	Orange	5.26	3.33

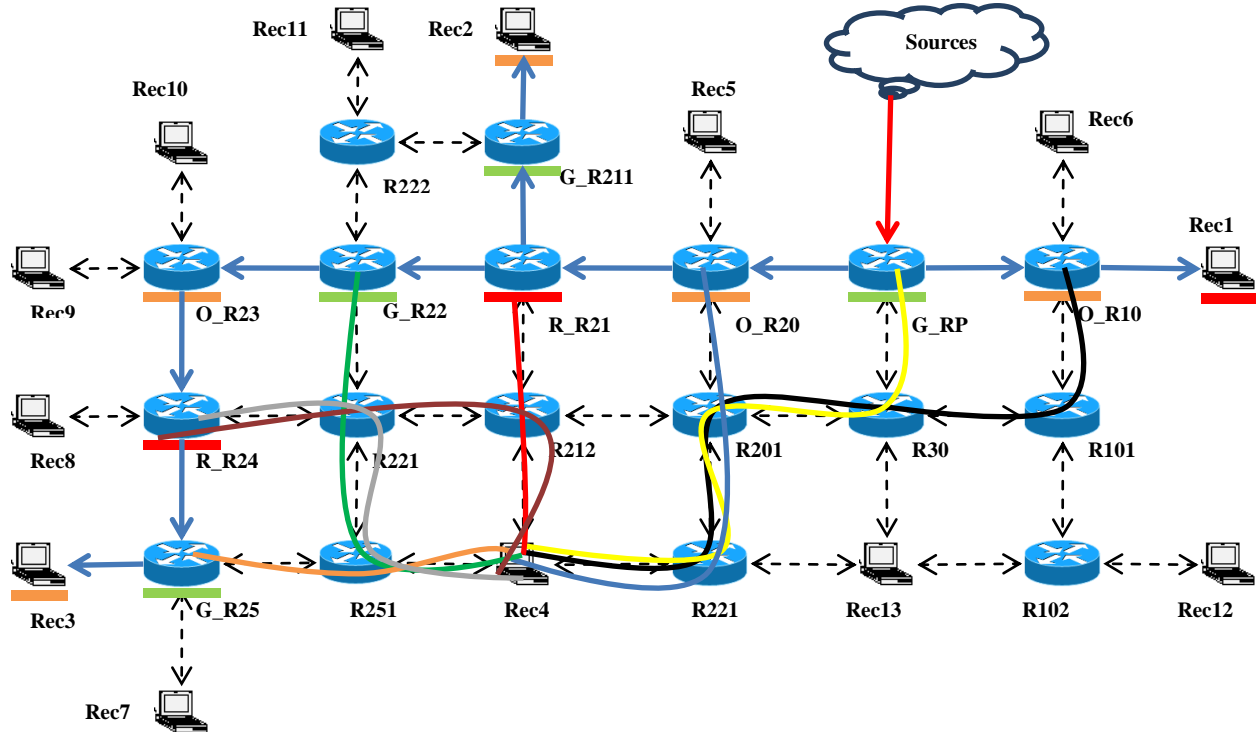


Figure 17: Path options for Rec4

Table 11 shows the first path option (Path1) related calculation by GORby algorithm, which is also PIM-SM option. GORby calculation of Path2 is presented in the Table 12. Path2 is the final option of GORby.

Table 11: Rec4 Path1 PGIA calculation

Step	Node	Upstream Node			Current Node Result		
		Name	Colour	PGIA	Colour	GIA	PGIA
One	R212	R_R21	Red	Infinite	Green	4	4
Two	Rec4	R212	Green	4	Orange	1.46	1.46

Table 12: Rec4 Path2 PGIA calculation

Step	Node	Upstream Node			Current Node Result		
		Name	Colour	PGIA	Colour	GIA	PGIA
One	R251	G_R25	Green	Infinite	Orange	2.34	2.34
Two	Rec4	R251	Orange	4	Red	2.48	2.34

To analyze Path3 for Rec4, just one step is enough. The first row of Table 13 shows this path requires R221 to be connected to the G_R22, which would have a PGIA of 1.25. We do not need to calculate the rest, because PGIA is not going to be more than 1.25 anyhow while Path2 PGIA is 2.34. So, Path3 loses to Path2 based on this calculation.

There is a similar situation on Path4 and Path5 about R221 to join via R_R24, which would have a PGIA of 2 that is less than 2.34 (Path2 PGIA), and would lose to Path2. Because PGIA is always the minimum value of the nodes GIA along the path, none of these two paths would have a PGIA more than 2, and consequently they both would lose to the Path2 in competition.

Table 13: Rec4 different path PGIA calculation

Path	Node	Upstream Node			Current Node Result		
		Name	Colour	PGIA	Colour	GIA	PGIA
Path3	R221	G_R22	Green	Infinite	Orange	1.25	1.25
Path4 & 5	R221	R_R24	Red	Infinite	Green	2	2
Path6	R201	O_R20	Orange	Infinite	Red	3.3	3.3
	R202	R201	Red	3.3	Green	3.3	3.3
	Rec4	R202	Green	3.3	Orange	1.46	1.46
Path7	R30	G_RP	Green	Infinite	Orange	3.5	3.5
	R201	R30	Orange	3.5	Red	1	1
Path8	R101	O_R10	Orange	Infinite	Red	4.2	4.2
	R30	R101	Red	4.2	Green	3.8	3.8
	R201	R30	Green	3.8	Orange	1	1

In the first step, Path6 needs R201 to be connected via O_R20, and has a GIA of 3.3. So, the PGIA of this path up to this point becomes 3.3. In the next step it needs R202 to join via R201 and has the same GIA (3.3), as a result PGIA is still 3.3. However, in the final step, Rec4 would have a GIA of 1.46 and the whole path PGIA becomes 1.46, so it loses to Path2.

For Path7, we calculate two steps and reach a PGIA of 1 that is less than 2.34 of Path2 PGIA. The final path that we study here is Path8. For this path we brought three steps calculations for nodes R101, R30 and R201 and we reach to the PGIA of 1 again (similar to Path7). So, this path would have a PGIA maximum of 1 and is less than 2.34 of Path2 PGIA, and loses to the Path2. There are other options as well, and all of them have the PGIA less than 2.34, so Path2 became the final option based on the GORby algorithm for the Rec4 to join the tree.

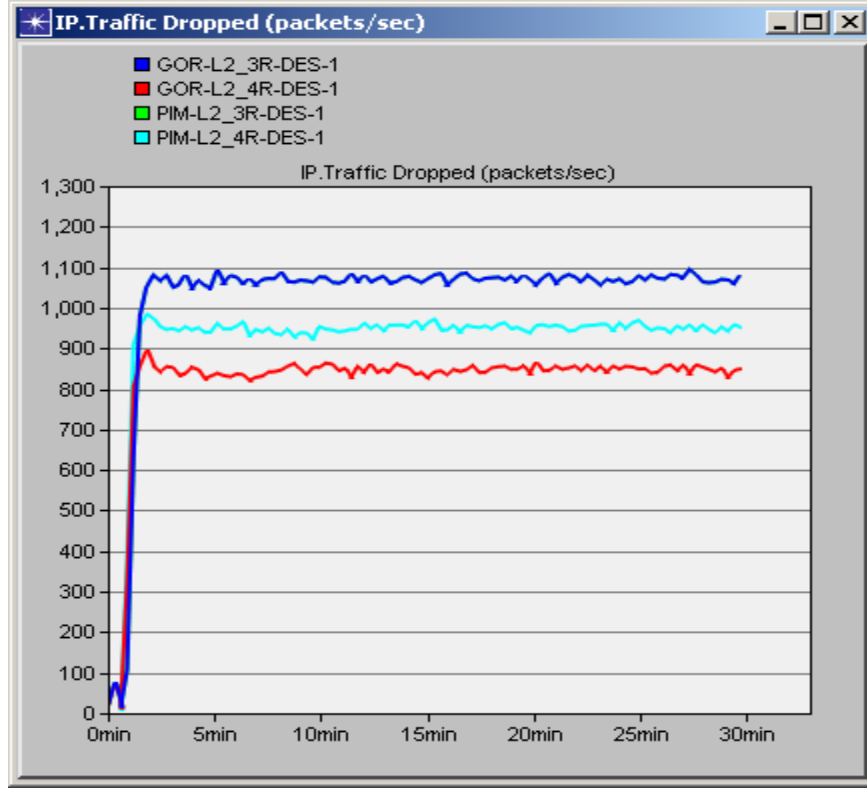


Figure 18: Serial join analysis, overall traffic dropped by all nodes

4.2.1 Simulation result:

We use the following four scenarios in a constant average traffic of 1000 Pack/Sec:

- P-L2_3R: Multicast routing protocol is PIM-SM; there are three receivers.
- G-L2_3R: Multicast routing protocol is GORby; there are three receivers.
- P-L2_4R: Multicast routing protocol is PIM-SM; there are four receivers, and Rec4 joins the group at the 210th Sec. This join is via R212 to reach R_R21.
- G-L2_4R Multicast routing protocol is PIM-SM; there are four receivers, and Rec4 joins the group at the 210th Sec. This join is via R251 to reach G_R25.

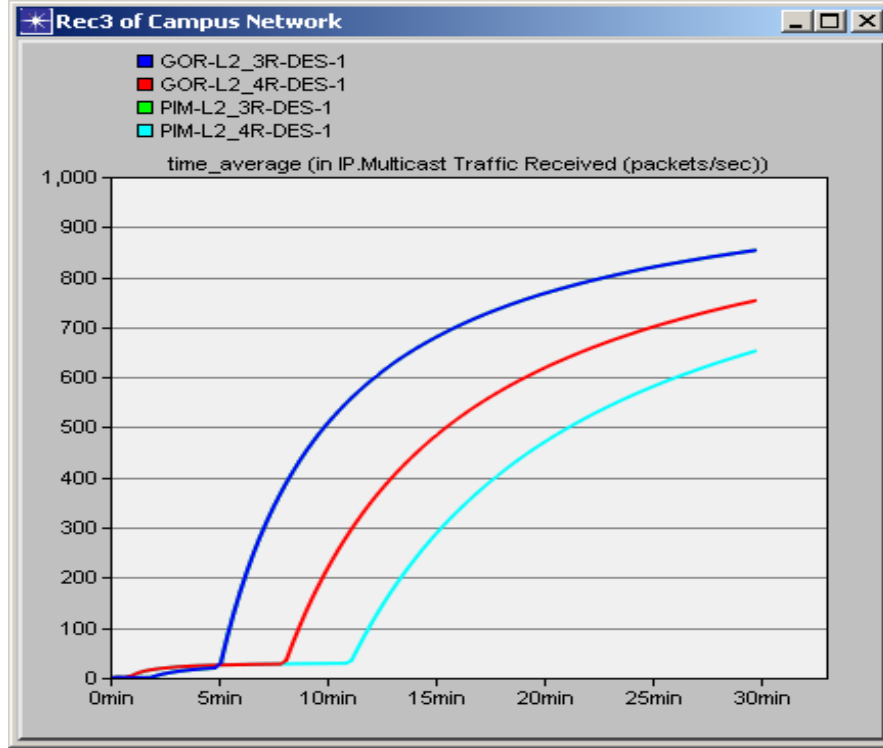


Figure 19: Serial join analysis, average received packet by Rec3

Figure 18 shows the total packets dropped by the entire nodes in the network. In this figure, GORby and PIM-SM graphs for three receivers' scenarios are the same (top graph), because we have the same tree in these two scenarios. It shows that in a comparison between GORby (middle graph) and PIM-SM (bottom graph) in four receivers scenarios, packets dropped by using PIM-SM protocol is between 5% and 10% more than by using GORby protocol. Having Rec4 joining network following PIM-SM suggestion, caused interference from node G_R212, which is in GORby-Group Green, over nodes O_R20 and O_R23 in receiving state, and this increased interference is one of the reasons for packet dropping.

Figure 19 shows average traffic received by Rec3. Again this figure supports GORby (top graph) and show that we are receiving more data in this receiver (same in case Rec4) in the case of using our protocol. Because nodes (routers and receivers) have less interference (and better SINR) in GORby, they would have fewer packets dropped.

Table 14: GIA of nodes for topology of Figure 17

Scenario	Node	Colour/Index	X	Y	GIA
PIM/GOR-L2_3R	O_R23	Orange / 5	1	2	1.95
PIM/GOR-L2_3R	R_R24	Red / 6	1	1	7.22
PIM-L2_4R	O_R23	Orange / 5	1	2	1.40
PIM-L2_4R	R_R24	Red / 6	1	1	7.22
PIM-L2_4R	G_R212	Green / 4	3	1	2.86
PIM-L2_4R	Rec4	Orange / 5	3	0	1.46
GOR-L2_4R	O_R23	Orange / 5	1	2	1.95
GOR-L2_4R	R_R24	Red / 6	1	1	1.57
GOR-L2_4R	O_R251	Orange / 8	2	0	2.34
GOR-L2_4R	Rec4	Red / 9	3	0	2.48

Table 14 presents a summary of some key nodes GIA to compare them, and shows that by using GORby protocol, nodes GIA (SINR) are improved.

- R23: This node GIA is 1.9512 in scenario PIM/GOR-L2_3R. Then in PIM-L2_4R becomes 1.4035 and in GOR-L2_3R is 1.9512. This means that adding a new path would cause more interference on this node.
- R24: GIA in case of PIM/GOR-L2_3R and PIM-L2_4R is 7.22; however, it is 1.5663 for GOR-L2_3R.

Note: It is still more than 1.4035 of original scenario R23.

- R212 – Rec4 (PIM-L2_4R): GIA is 2.8571 for R212; and 1.7391 for Rec4, so PGIA is 1.7391.
- R251 – Rec4 (GOR-L2_3R): GIA is 2.3423 for R251; and 2.4880 for Rec4, so PGIA is 2.3423.

As the above table and description shows, we have a better situation in scenario GOR-L2_4R. It even shows that changes on R23 and R24 would impact other path PGIA, which in case of PIM-L2_4R is worse.

For this section and analysis, there are more graphs in Appendix A that support the design and show better performance in GORby protocol. They are same receivers and some intermediate routers, and because they are similar to Figure 18 and Figure 19, we did not include them here in the main discussion to make it shorter.

We studied serial join with a fixed traffic in this section. In the next section, we analyze the impact of different traffic flow in our protocol.

4.3. Traffic analysis

In this section, we use different traffic types presented in Table 4, to evaluate the performance of GORby versus that of PIM-SM under different traffic loading conditions. More specifically, in this section, we add five more traffic flows to have in total six scenarios for “three receivers” models and six scenarios for “four receivers” models. Each scenario’s name presents the scenario protocol, number of receivers and traffic, based on the description in section 4.1. We also use multiple seed sizes based on Table 5, and then calculate the average of the results.

We study the average (time) packet dropped by the entire network nodes that shows the difference between GORby and PIM-SM, and improvement of GORby over PIM-SM that is almost the same (approximately 8%) for different traffic loads. Except in traffic load of 640 KPack/Sec that we have fewer packets dropped in PIM-SM, which was an unexpected result by OPNET. This figure is for a seed value of 128, although by changing the seed value, the graphs may change a little bit. So, we run the simulator for four seed values of 31, 67, 97 and 128, which the average value of these runs per traffic is shown in Figure 20.

Figure 20 presents the average packet dropped per four scenarios (PIM-SM and GORby, 3 and 4 receivers) for each traffic flow, which is compatible with Figure 18. In this figure:

- G3R: represents average for scenario of GORby Multicast protocol with three receivers.
- G4R: represents average for scenario of GORby Multicast protocol with four receivers.
- P3R: represents average for scenario of PIM-SM Multicast protocol with three receivers.

- P4R: represents average for scenario of PIM-SM Multicast protocol with four receivers.

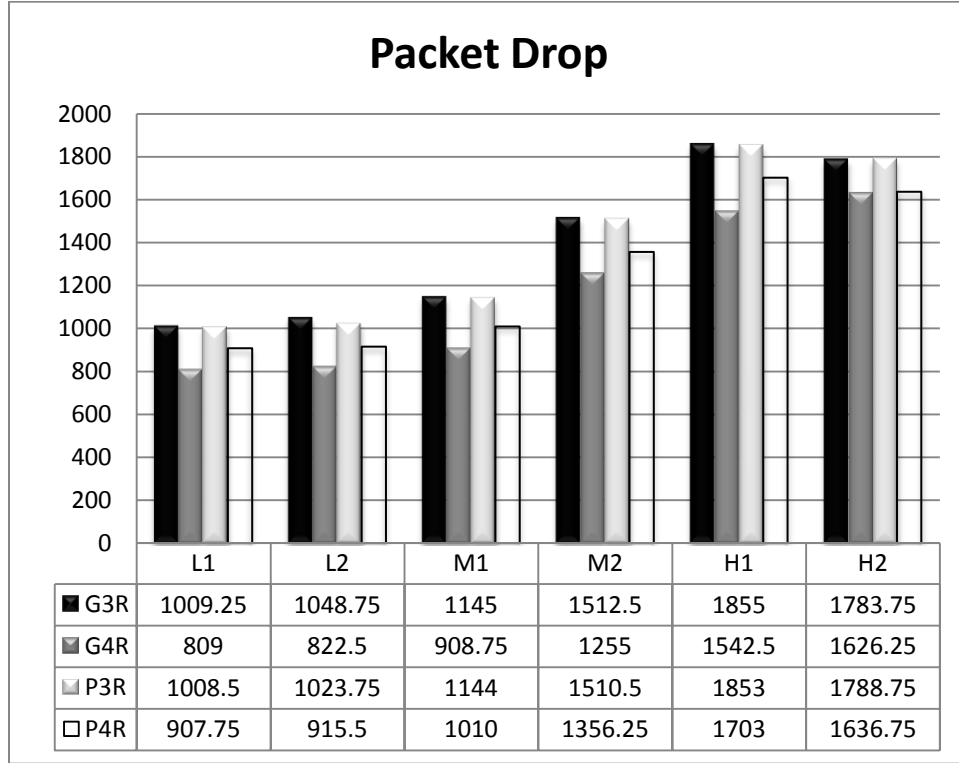


Figure 20: Traffic analysis, average packet dropped per traffic

Figure 20 presents the average packet dropped per four scenarios (PIM-SM and GORby, 3 and 4 receivers) for each traffic flow, which is compatible with Figure 18. In this figure:

- G3R: represents average for scenario of GORby Multicast protocol with three receivers.
- G4R: represents average for scenario of GORby Multicast protocol with four receivers.
- P3R: represents average for scenario of PIM-SM Multicast protocol with three receivers.
- P4R: represents average for scenario of PIM-SM Multicast protocol with four receivers.

We also calculate the average for all of the traffic models. The average traffic dropped of PIM-SM in three receivers is 0.99692 times of GORby, which is almost 1. Although for four

receivers, this value for the PIM-SM is 1.08117 times of the GORby value. Therefore we have 8.117% improvement in the packet dropped.

In any type of traffic load, GORby performs better in terms of packet dropped. Increasing the traffic causes more packets dropped while the difference between PIM-SM and GORby in four receivers stays the same.

We also analyze traffic received in Rec3 by having a load of 160,000 Pack/Sec. By increasing the traffic load, GORby performs better. When nodes are not able to transfer the packet successfully, they have to buffer them for retransmission while still receives more packets from upstream nodes. Increasing traffic flow causes more transmissions by all nodes and more interference as well. Consequently, nodes drop more packets and less traffic is received by receivers.

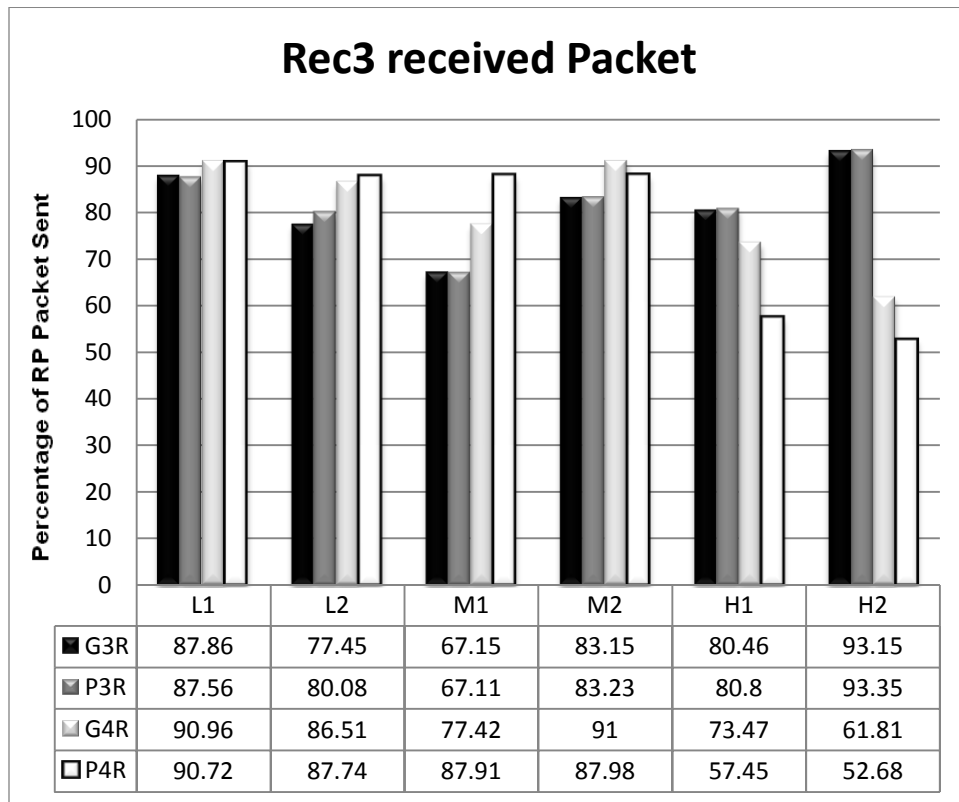


Figure 21: Traffic analysis, average (percentage) of traffic sent by RP received by Rec3

Figure 21 and Figure 22 present the percentage of traffic sent by RP that Rec3 and Rec4 received on average for different seed size simulations.

Figure 21 shows P3R and G3R are almost the same because we have the same tree in these two scenarios, although the packet received percentages are different in the four receiver scenarios. On average between all kinds of traffic shown in Figure 21, Rec3 receives 0.9904 times traffic in G3R (GORby) versus P3R (PIM-SM), which is almost 1. However, the average packet received by Rec3 in G4R is about 1.0359 times of P4R, which shows that Rec3 receives about 3.6% more traffic in GORby.

Figure 21 also shows, when traffic is low, PIM-SM performs better, although in high traffic flows, GORby performance is significantly better than PIM-SM scenarios. When the traffic is low, nodes have more chance of gaining access to the media in the case of packet transmission failure. Although when we have a high traffic flow that increased interference, packets get dropped more often because of high traffic throughputs and are affected by interference.

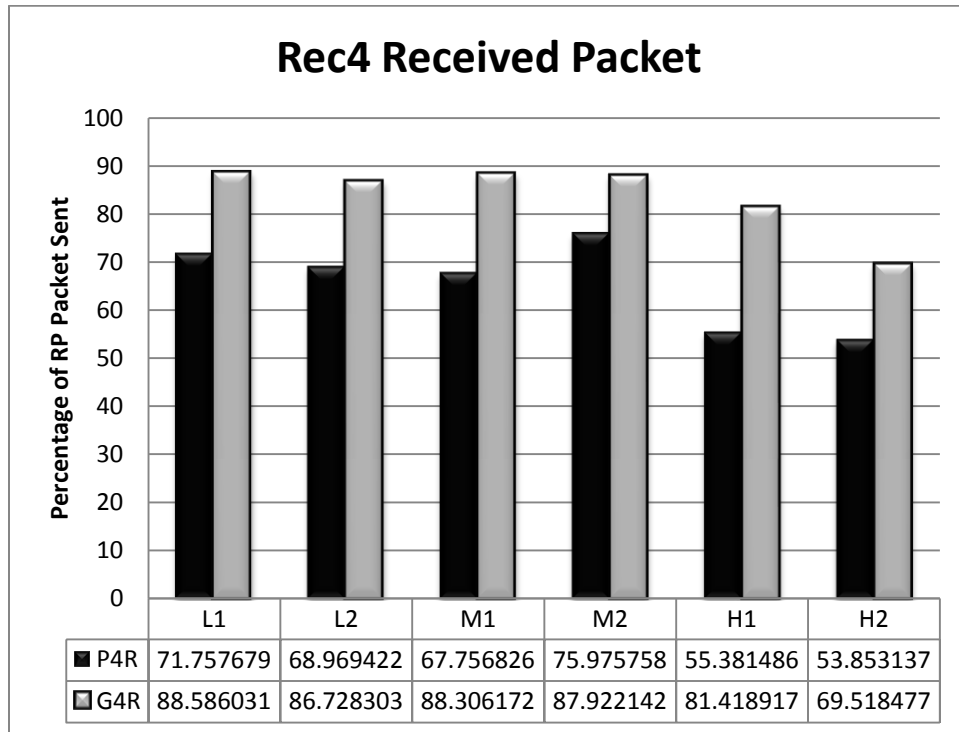


Figure 22: Traffic analysis, average (percentage) traffic sent by RP received by Rec4

Figure 22 shows that regardless of the traffic flow, GORby performs significantly better in terms of packet received by Rec4. Even though it is a longer path, we have less interference in GORby protocol for this receiver path. Also on average, Rec4 in G4R receives 1.2763 times of P4R, which resulting in about 27.63% improvement.

We evaluated GORby versus PIM-SM for serial join and different kinds of traffic (and seed values) in this section. In the next section, we will evaluate our protocol from convergence point of view and in multiple join.

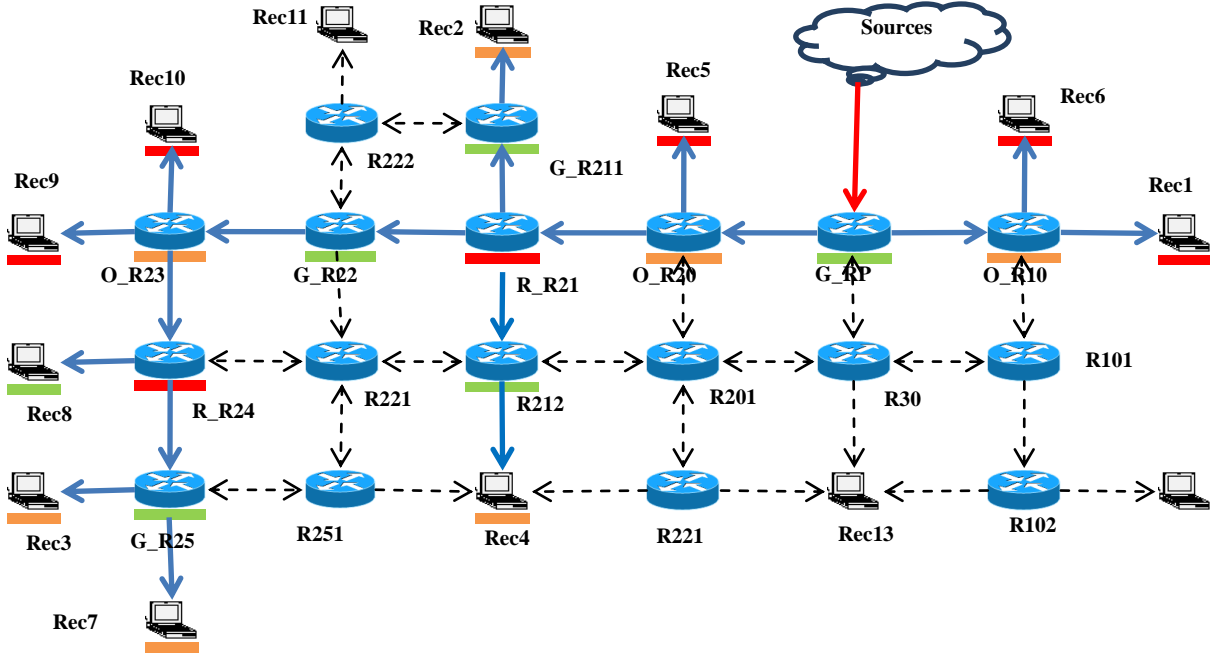


Figure 23: Joining ten receivers in PIM-SM Multicast protocol

4.4. Parallel join analysis

In this section, we use the original topology presented in Figure 12 and the OPNET setup is presented in Figure 13, and increase the number of receivers joining the tree to ten. Three receivers Rec1, Rec2 and Rec3 join in the 85th, 105th and 135th Sec. Then, seven more receivers

(Rec4 to Rec10) join the group at the same time, which is at the 210th Sec. Rec4 follows the same steps as section 4.2, and Rec5 to Rec10 have one option and will join the tree directly with one hop away.

The set up was configured in order to study and evaluate GORby by comparing it to PIM-SM following the previous sections. Figure 23 presents PIM-SM results and Figure 24 presents the results for GORby.

For this section, we created twelve scenarios, and their names follow the general naming convention presented in section 4.1. For instance, PIM-M1_10R means PIM-SM protocol, 10 receivers and M1 traffic model (Table 4), and GOR-M1_10R represents GORby protocol, 10 receivers and M1 traffic model (Table 4). We also (similar to the previous section), used four seed values of Table 5, and for some of our evaluation, we calculated the average value of the parameters to present the performance indicators.

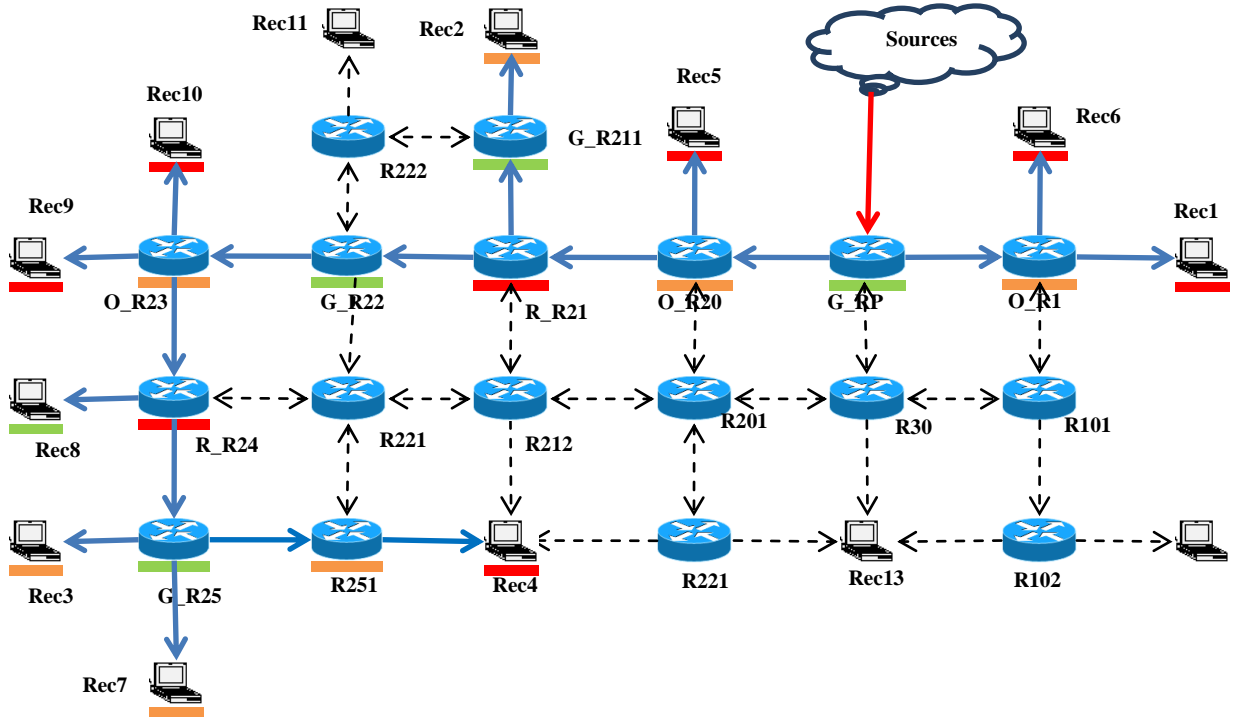


Figure 24: Joining ten receivers in GORby Multicast protocol

Firstly we study about dropped traffic, which follows the studies found in previous sections. Increasing the number of receivers does not cause any change on relative GORby performance in terms of packet delivery over PIM-SM. Although in overall (based on previous case) the rate of packet drop is increased, GORby performs better compared to PIM-SM in ten receivers as well. Increasing the number of receivers does not have a negative impact on GORby performance compared to PIM-SM.

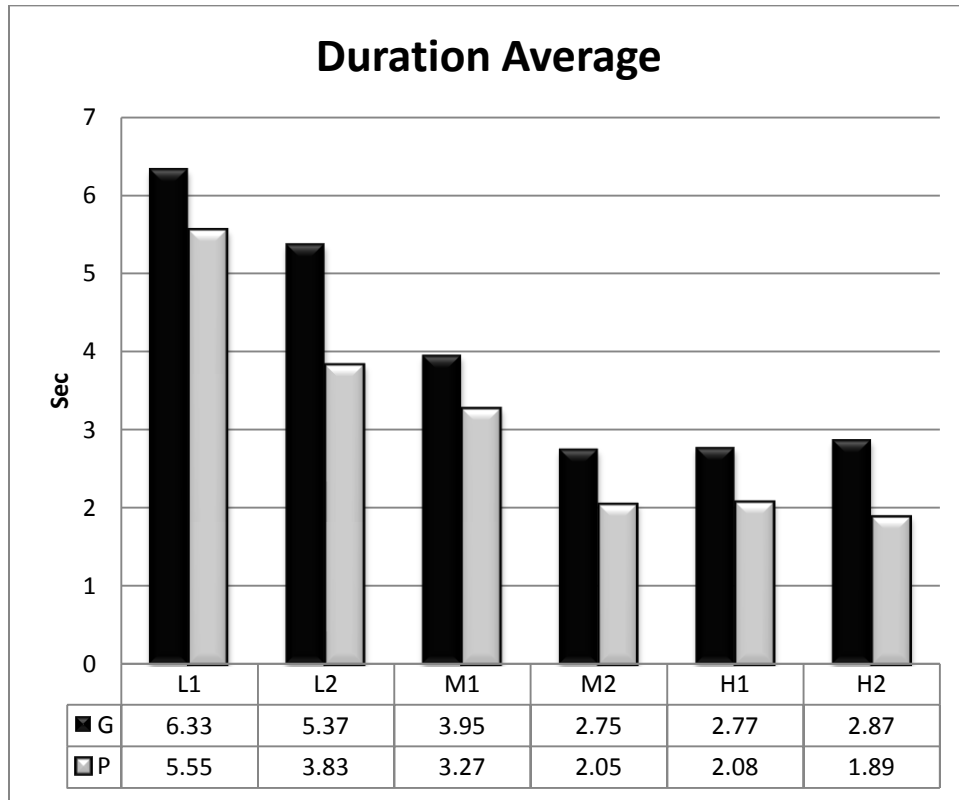


Figure 25: Parallel join analysis, average of convergence duration

Then we study average (time) convergence activity and duration for this topology. As our simulation results show, GORby needs more time for convergence, which is a major drawback of this protocol. It is because new members need to download Multicast tree nodes location and colour information in order to be able to calculate GIA, and then choose the best joining options. This requirement causes a larger convergence time; however, because GORby is designed for

Mesh network with fairly stable topology (compared to mobile wireless networks), the joining process is most likely only performed once for each receiver. Thus the large convergence time is not a significant issue. Also, our simulation results show that having more new members joining request would increase the convergence time further. In PIM-SM protocol we have a two way handshaking to join. When a node receives a request from a new receiver, it responds by RREP if it is part of the tree and the new member will join; and if it is not, it joins the tree and then will respond to the request, just two steps in each join.

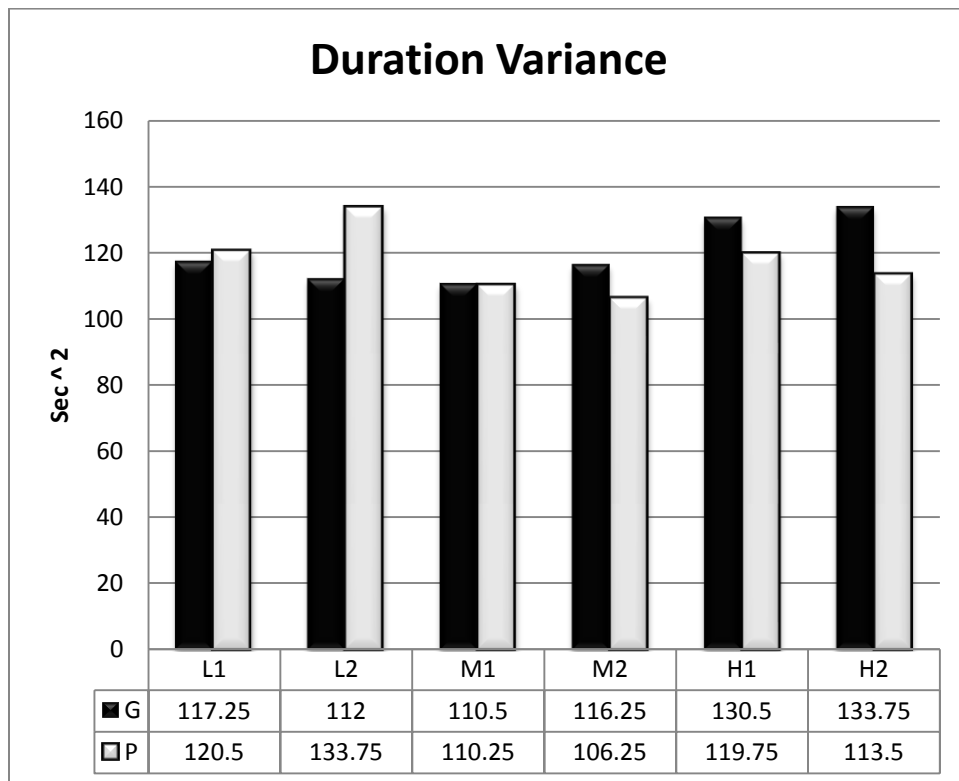


Figure 26: Parallel join analysis, variance of convergence duration

However, in GORby protocol we have three way handshaking to join. New receiver broadcasts the RREQ (First step), nodes that receive this query, will respond if the node is a part of the tree. Nodes that are not part of the tree will find the path, set up the path as temporary (Potential) and respond back to the new member (Second step). Based on the received information, the new

receiver chooses one of the received options and responds to the node that provided the chosen path to become a part of the tree (Third step). These activities require more time and more control data transferring between nodes for convergence. GORby has more overhead compared to PIM-SM in convergence time, although we designed this protocol for WMN which is a stable topology. Because nodes join once and keep the connection by the hello packet (that we have in PIM-SM as well), and this overhead can be ignored compared to other benefits that GORby provides. Hello packet of GORby has just four bytes data more than PIM-SM, and for example, nodes send the hello packet every 30 Sec, so still we have more overhead to maintain the protocol; however, it is a very low controlling traffic comparing to the main network data traffic.

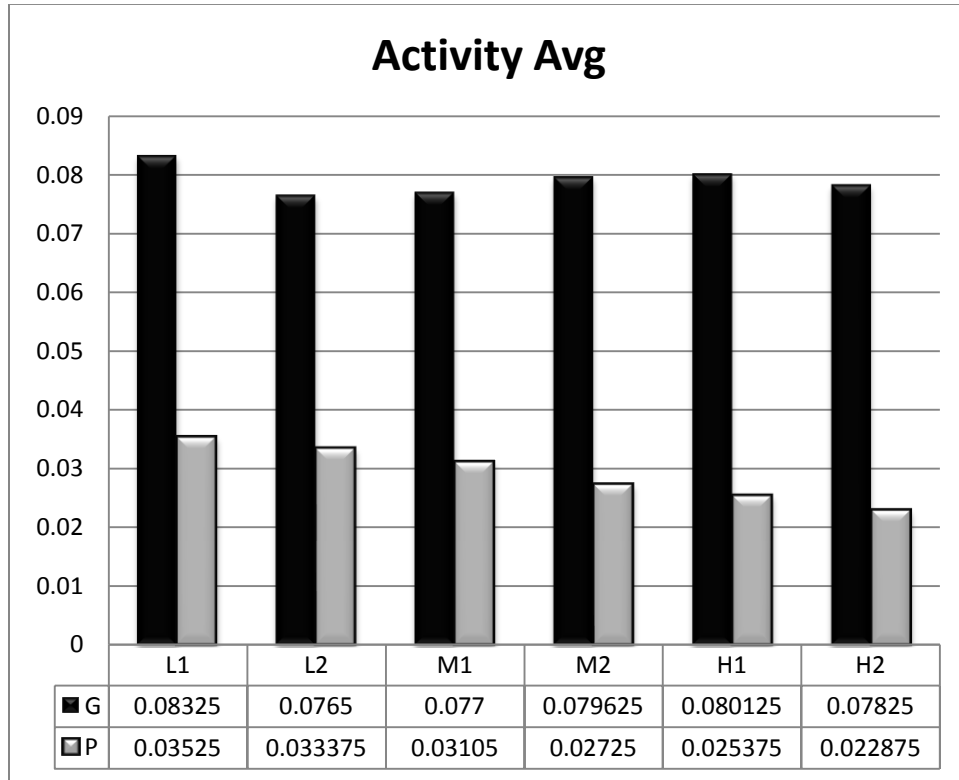


Figure 27: Parallel join analysis, average of convergence activity

For this analysis, we run different seed values and calculated the average of the convergence duration for the entire network nodes (Figure 25). We also calculated the variance for this

parameter that is shown by Figure 26. These two figures show convergence activity fluctuation is almost the same for GORby compared to PIM-SM, although there is an increase in the average time value in all kinds of traffic. We also calculated the average for these six traffic flows that shows our network needs about 1.28763 times of PIM-SM for GORby to converge. The average convergence time in GORby is about 28.76% more, which was expected. Following our previous discussion in this section, three way handshaking requires more time. The next point that convergence indicator shows, is although we need theoretically 50% more transaction (in terms of quantity) for convergence in GORby, the required time is not 50% more. In fact, we just need less than 30% more time because of better packet transferring by GORby protocol compared to PIM-SM.

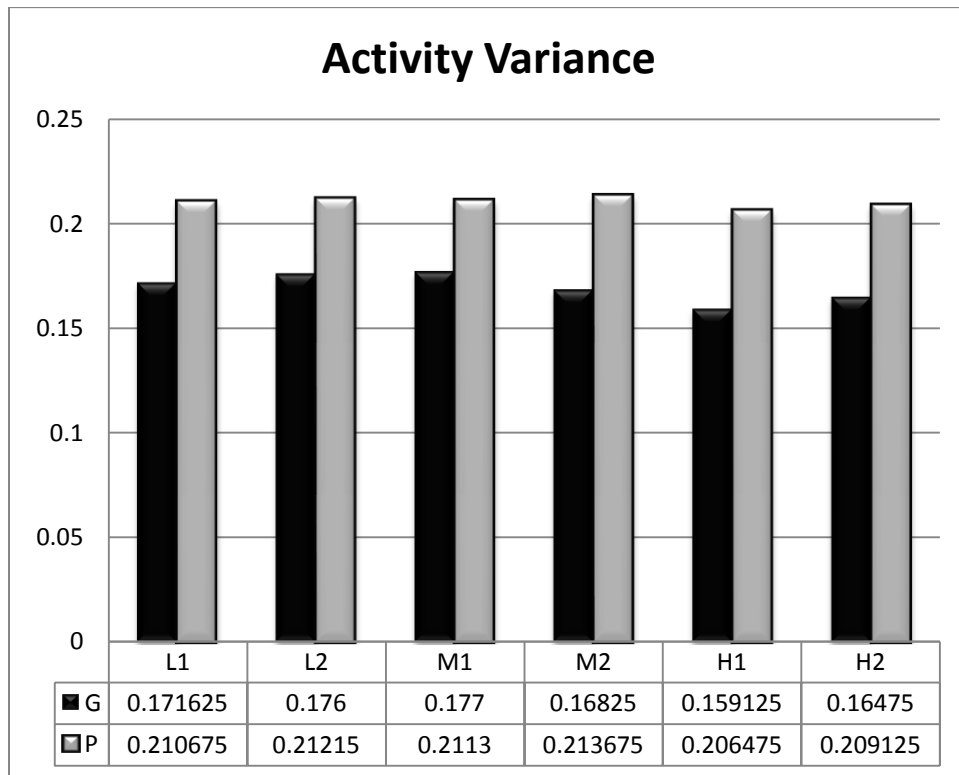


Figure 28: Parallel join analysis, variance of convergence activity

To complete our analysis in convergence, we prepared two graphs shown in Figure 27 and Figure 28 for convergence activity parameter. First of all, same as convergence duration variance, convergence activity variance presented in Figure 28 shows the stability of this parameter is almost the same for both protocols. Although in terms of the number of activities, we have a significant increase in GORby compared to PIM-SM, which was expected. Network nodes in PIM-SM just need to know the shortest path to RP in order to join the tree. In GORby, these nodes request information from different paths, and then choose one of them. In the first view as previously explained, it may be seen to have one extra action, but in these three way of handshaking, there are more than two nodes involved. So, it increased the number of activities per node, and subsequently the overall average of the entire network increased. Our calculation shows, the average of activities in GORby is 2.71015 times of PIM-SM, which is about a 171.02% increase in this indicator.

4.5. Summary and overall evaluation

In this chapter, we started by having a Multicast tree with one RP. Receivers joined the tree one by one (Rec1, Rec2, Rec3 & Rec4). We studied nodes options and showed how they choose the best path based on our algorithm. We ran the simulation to compare GORby and PIM-SM protocols performance (section 4.2) for the first four receivers (Rec1 till Rec4). To do this, we studied different network parameters like Packet Drop, Convergence and Packet percentage Received, which is similar to the throughput. To summarize these evaluations, and by using the OPNET simulator, in most of the case (except convergence), we found between a 4% and 9% improvement on traffic handling, for example in traffic received by receivers or overall dropped packet by all of the nodes. In the convergence study, however, PIM-SM provides better performance compared to GORby, although having better performance in other parameters, makes GORby an improvement on PIM-SM for Multicast protocol. Especially in a more stable network with fewer changes like Mesh, GORby performs better. Our analysis showed that the performance of GORby is not closely related to the traffic flow. Having said this, because GORby takes interference into account, it performs better in WMN in most of our evaluation analysis scenarios. In convergence activity, PIM-SM performs better compared to GORby.

GORby requires almost three times more activities and on average needs about 30% more time for convergence, which is because of the GORby algorithm in joining Multicast tree. As we discussed, we designed GORby for WMN, and this negative impact on convergence compared to other benefits that we received from GORby is acceptable. Therefore compared to PIM-SM, GORby has the best overall performance.

Chapter 5

Conclusion and Future Research

This chapter has two sections and objectives. First section is conclusion of the research, and in the second section, potential future works are presented.

5.1. Conclusion

In this thesis we presented an Interference-Aware Multicast join protocol. The main idea of this protocol is to estimate the interference at a node, and based on that estimate evaluate the potential of that node for joining the Multicast distribution tree. First of all, we distribute the nodes in the Multicast distribution tree into three GORby-Groups, and scheduled them based on time slots to be in Transmit, REC1 and REC2 states, which decreases nodes collisions. In the next step and by using GORby-Grouping characteristic, we presented concept of GORby interference. GORby interference means potential interference on a target node, at the time that this node wants to join Multicast group tree, is caused by all nodes belonging to the potential sender GORby-Group. Then, based on this potential cumulative interference, we calculated potential SINR, which we called it GIA (a new metric developed by this research). GIA metric

plays the main role on joining a Multicast tree decision. Calculating SINR (GIA) is a local function of each node, based on location information of other Multicast tree nodes (per GORby-Group).

To expand GIA, we presented a new metric called PGIA, which is a path GIA, and calculated based on the minimum (worth case) of all nodes' GIA along the path to the Multicast group tree. To calculate PGIA, each node receives PGIA from upstream node, and then defines new PGIA to be the minimum of local GIA and receiving PGIA (from upstream node). PGIA calculation is done locally by each node of a potential path. New member that wants to join the Multicast group tree chooses the path that provides the maximum PGIA, and joins the tree.

We used OPNET to simulate GORby, and evaluated our proposed solution and concepts by some examples and scenarios. We analyzed GORby performance versus PIM-SM, in different traffic load, and in low and high join request situations. We found GORby provides between a 4% and 9% improvement on traffic handling, for example in traffic received by receivers or overall dropped packet by all of the nodes. In convergence study, PIM-SM provides better performance compared to GORby, although having better performance in other parameters, makes GORby an improvement on PIM-SM for Multicast protocol. Especially in a network with fewer changes like Mesh, GORby performs better. Performance of GORby is not closely related to traffic flow, although in traffic analysis showed it performs for some receivers better in higher traffic. GORby requires almost three times more activity, and on average needs about 30% more time for convergence, which is because of the GORby algorithm in joining Multicast tree. As we discussed, we designed GORby for WMN, and this negative impact on convergence compared to the other benefits that we received from GORby is acceptable. Therefore compared to PIM-SM, GORby has the best overall performance.

5.2. Potential future works

This research can be expanded and developed from different stand points. In this section, some of these potential works are presented:

- GORby can be expanded to divide nodes from a different point of views. Based on the application and environment/platform, the idea of GORby-Grouping the nodes for interference estimation can be expanded to unicast protocol. For sure unicast has its own characteristics and constraints, and requires its own analysis.
- In this design we assumed only calculating interference caused by the current tree on a new join (through one hop or multi hops). The interference is bidirectional affect. A potential interference of the new path over the current tree can be added to the algorithm.
- The next assumption of this design was new members join the tree one by one. Therefore, because we only brought the current tree interference into account, this assumption is fine; however, if a potential join interference being considered, nodes cannot be added one at the time.
- This model is developed mainly for core based and shared tree structures. Current design with a minimum changes can be adapted to be a source based model. In fact in this case, sender would acts like a core for Multicast group.
- GORby can be developed to cover multi sources as well. Current model covers multi sources in a core based structure from core down, although for the communication between sources to core need some attention.
- Also, GORby can be developed for multi source and source based structure Multicast group(s). We will have a multi group situation, and interference caused by each group on other groups should be considered. This is almost most complete model, which will cover all of the previous mentioned potential works. To reach to this extend, GORby needs to use more than three colours. Convex optimization could be one of the approaches of handling this case or partially helps on modeling and analyzing the problem, although it is a complicated model.

- Finding optimum size of time slot is a potential future work in this area. Having smaller size causes more dropping Multicast data, and having bigger size causes fewer throughputs. Finding the optimum size is more related to the type of traffic and application that uses the Multicast group to send the data.
- We set the time slot manually, meaning like selecting type of the Multicast protocol (PIM-SM or GORby). We assumed the network administrator set this value in all of the routers and nodes in the network. However, this action can be a dynamic one, and calculated and set by for example RP (or another node that can be the network server). Following to the previous suggestion about finding optimum slot size, RP can calculate the value because it is the first one that receive the traffic, and then send it to the group members. Also, as soon as a new member joins the group, it receives this value from upstream with other controlling values like colour.
- In our join algorithm, a member chooses the path mainly based on GIA/PGIA, and the only time that the member brings hop count into account is when it finds more than a path as the best path. Having both metric in member join algorithm is the next area to develop GORby algorithm.
- Because we keep the network topology the same in both PIM-SM and GORby algorithms evaluations, we blamed most of the packet dropped to SINR (and GIA); however, nodes' buffer could cause some of this packet dropping as well. Finding the best queue size and bringing it into account with the previous research suggestion (GIA/PGIA and hop-count) to choose the best path to join is the next research area to study and improve GORby.
- Modifying GORby for Dense mode is our next potential future work.
- Our last suggestion to improve this algorithm is working on synchronizing the nodes. We may need to update this synchronization periodically.

Appendix A

OPNET development

To implement GORby in OPNET, one router is modified to have two more attributes, which are:

- “mcast”: It has just two options including “PIM-SM” and “GORby, not editable.
By choosing this, we can set the Multicast protocol to be original “PIM-SM” or “GORby” version of it. This attribute just have these two options, and does not accept any other value.
- “tslot”: It has six options including {30, 50, 90, 150, 240, 300} milliseconds, editable.
It is used to set the time slot for GORby Multicast protocol. This attribute accepts other values to be entered as well.

It is obvious that if we set the router to use original “PIM-SM”, setting the second attribute does not make any difference. Also, if we set the time slot too small to not be able to even send even one packet during this slot, all traffic will be dropped.

Figure 29 shows these two attributes, which can be selected/entered for a router. In terms of receivers, we use same promoted nodes and convert it to be a receiver as well.

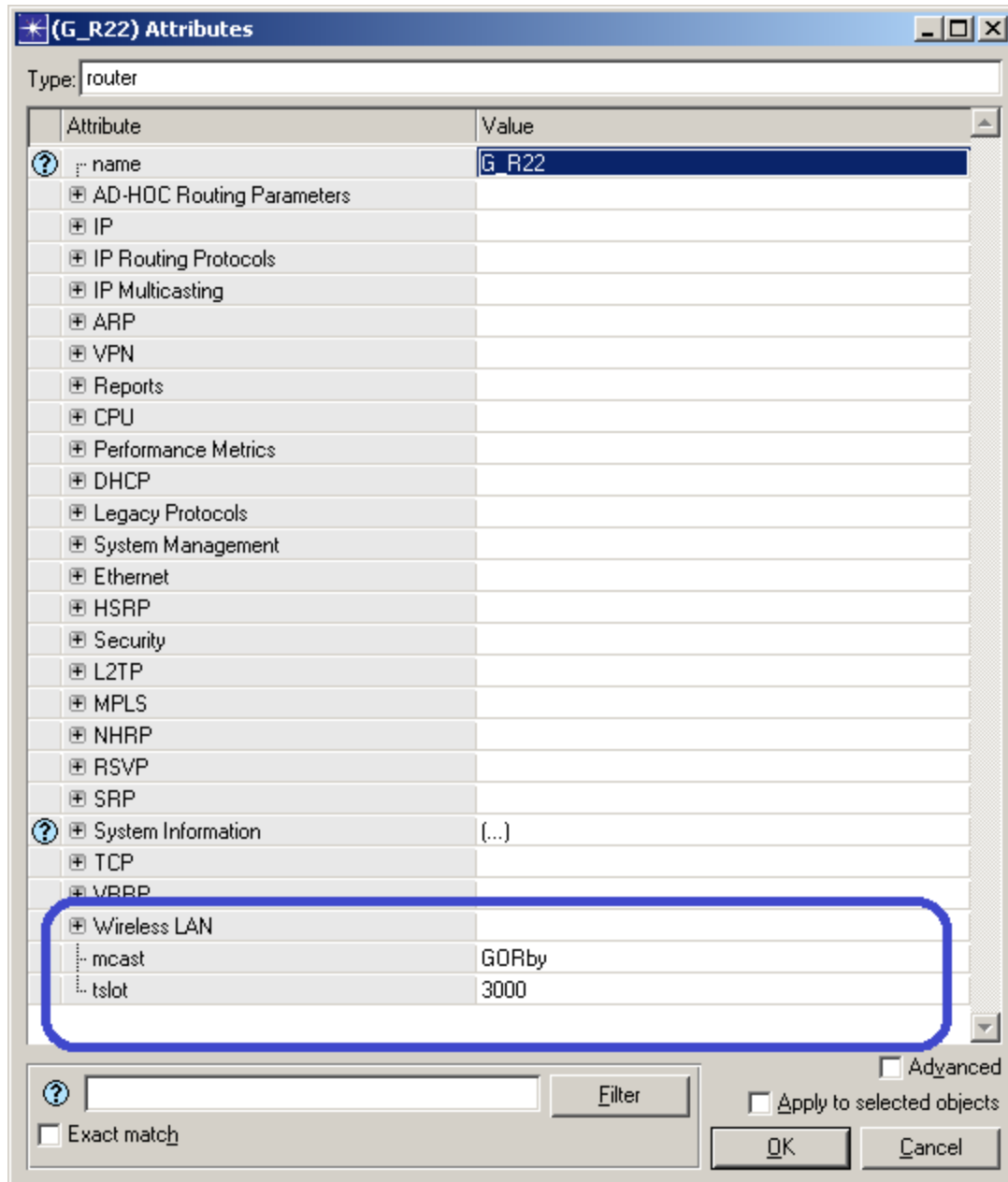


Figure 29: Router "mcast" and "tslot" attributes

In order to program the nodes, we changed mostly one of child process of “IP Process”, named “ip_pim_sm”. In order to do this, first we defined some constant in “Block Header”, which is presented by Figure 30, and some global variables in “State Variable”, which is presented by Figure 31.

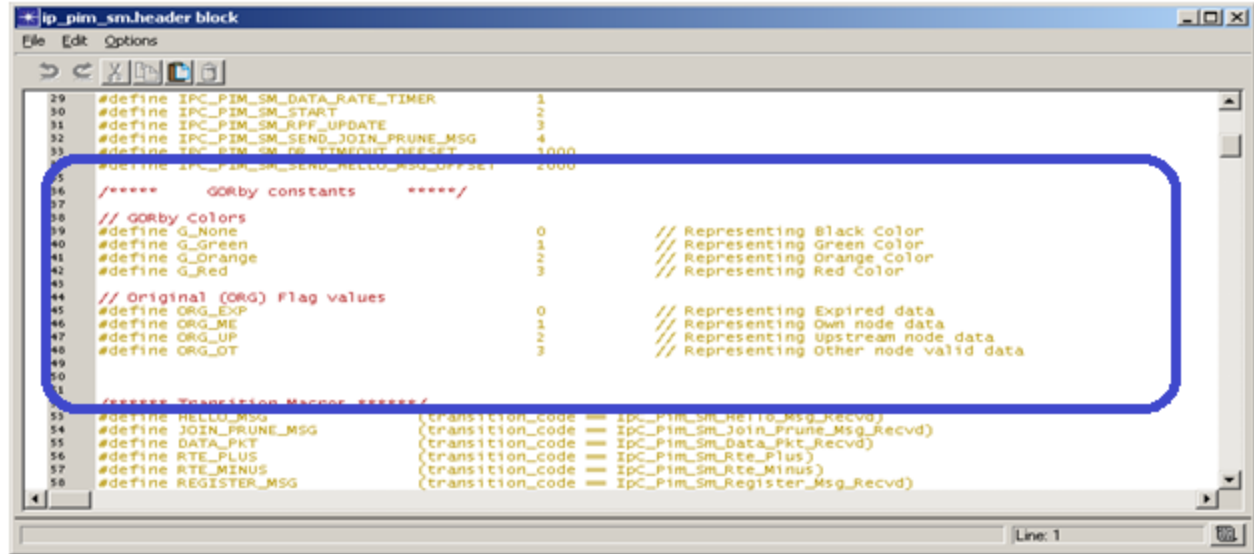


Figure 30: Defining constants in header block

Figure 32 shows required development in initialization section of this child process. First block are required variables, and second section shows initialization part of our development. We have more development in initialization section that is presented by Figure 33.

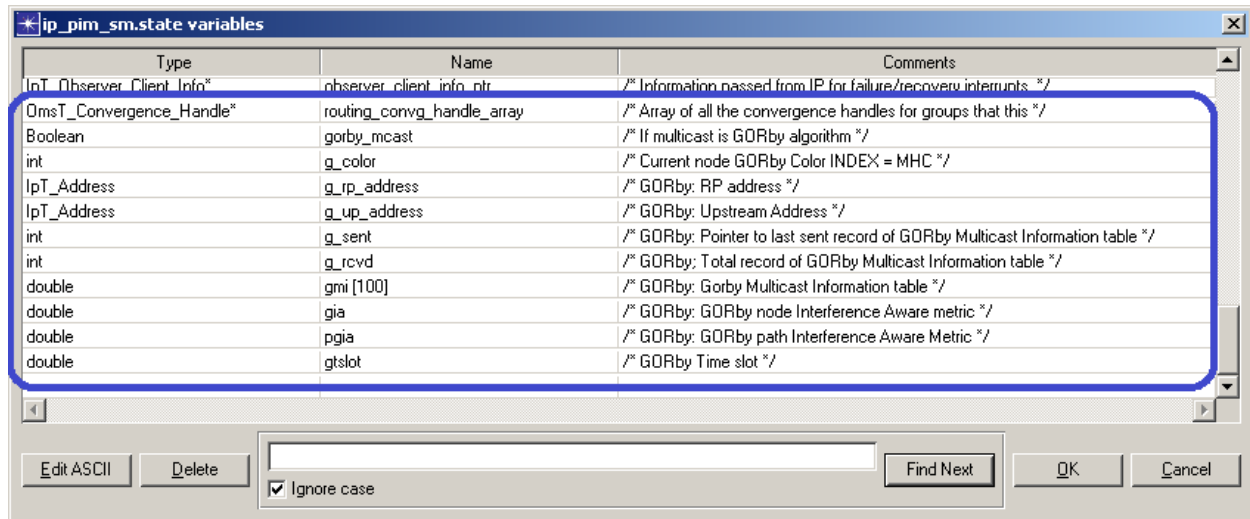


Figure 31: Defined SV (Global) variables

```

1  /***** Transition Executives *****/
2  static void
3  ip_pim_sm_do_init (void)
4  {
5      int          higher_layer_proto_id;
6      boolean      selected = ORG_FALSE;
7
8      // GORby Variables definitions start
9
10     Objid         own_objid_g1;
11     Objid         node_objid_g1;
12     char          g_mcast[6];
13     double        x_pos;
14     double        y_pos;
15
16     // GORby Variables definitions end
17
18     FIN (ip_pim_sm_do_init ());
19
20     /** Initialize all state variables and send Hello      **/
21     /** message on each IP interface, which is enabled for **/
22     /** multicast. Register this process model in the      **/
23     /** process registry. And also register this protocol **/
24
25     /* Initialize the state variables                      */
26     ip_pim_sm_sv_init ();
27
28     // GORby Start
29
30     /** Initializing GORby related parameter to start . It Reads **/
31     /** node attribute of "mcast" that shows this node is GORby **/
32     /** enabled (or it is PIM-SM), and leaves the result in VS **/
33     /** (global) variable "gorby_mcast" to be used in process. **/
34     /** It reads "tslot" attribute as well, and save it in VS. **/
35     /** Also set node color index to black as well as read node **/
36     /** positions (x & y). Load first record of gmi table with **/
37     /** info also set number of gmi record to one and number of **/
38     /** sent records to zero, to start with.                **/
39     own_objid_g1 = op_id_self ();
40     node_objid_g1 = op_topo_parent (own_objid_g1);
41     op_ima_obj_attr_get (node_objid_g1, "mcast", g_mcast);
42     op_ima_obj_attr_get (node_objid_g1, "tslot", &gtslot);
43     gorby_mcast = !strcmp(g_mcast, "GORby");
44
45     g_color = G_None;
46     op_ima_obj_attr_get_db1 (node_objid_g1, "x position", &x_pos);
47     op_ima_obj_attr_get_db1 (node_objid_g1, "y position", &y_pos);
48     gmi [0] = floor (1000 * y_pos) * 0x100000000 +
49         floor (1000 * x_pos) * 0x10000 + g_color * 0x100 + ORG_ME;
50
51     g_sent = 0;
52     g_rcvd = 1;
53
54     // GORby End
55
56     /* Determine whether PIM-SM Sim Efficiency mode has been*/
57     /* turned on. This function may update some attribute */
58     /* configurations as well.                             */
59
60

```

Figure 32: Initialization section (1)

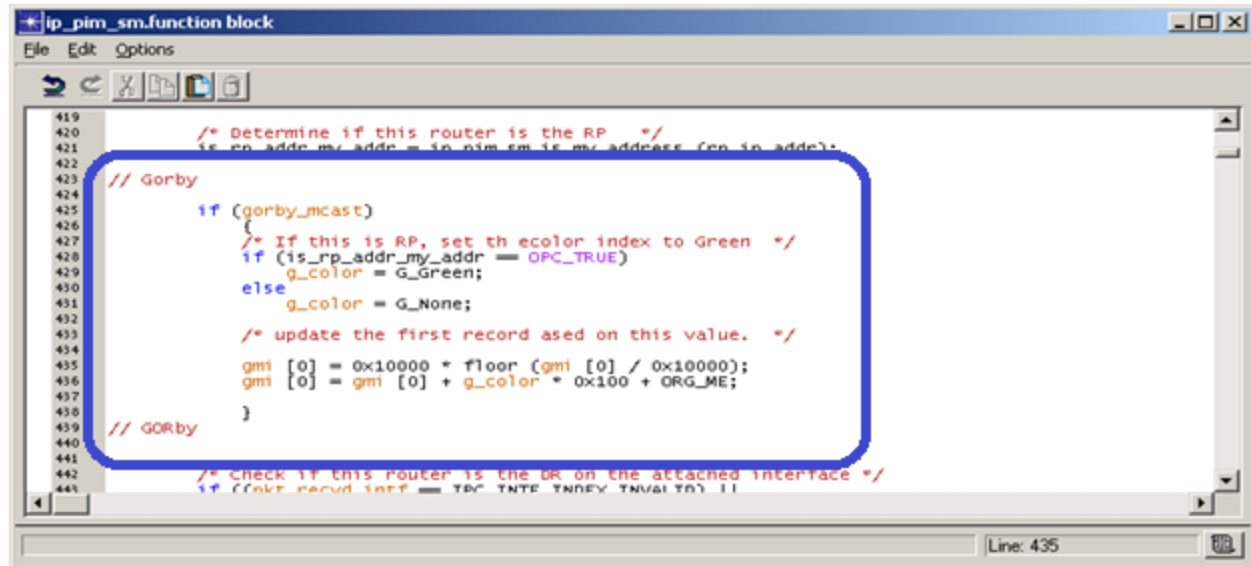


Figure 33: Initialization section (2)

Figure 34 and Figure 35 show required development in sending hello packet section of this child process. Figure 34 presents required variables for this function, and Figure 35 shows additional steps of creating hello packets.

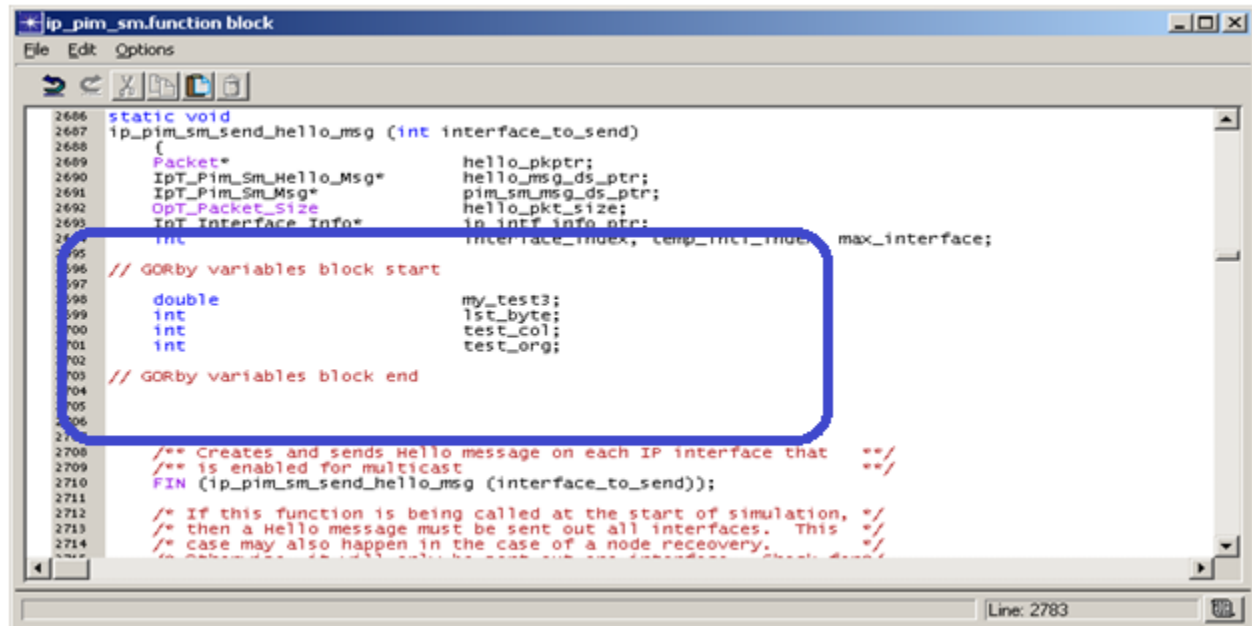


Figure 34: Variable definition for sending hello packet function

```

2752      /* Set the fields of the Hello message data structure */
2753      /* First set the holdtime. */
2754      hello_msg_ds_ptr->options_tlvptr [holdtime].option_type      = 1;
2755      hello_msg_ds_ptr->options_tlvptr [holdtime].option_len      = 2;
2756      hello_msg_ds_ptr->options_tlvptr [holdtime].option_value    = intf_array [interface_
2757
2758      /* Then set the priority. */
2759      hello_msg_ds_ptr->options_tlvptr [priority].option_type      = 19;
2760      hello_msg_ds_ptr->options_tlvptr [priority].option_len      = 4;
2761      hello_msg_ds_ptr->options_tlvptr [priority].option_value    = intf_array [interface_
2762
2763      // GORby Start
2764      /* run this section if GORby multicast is selected. */
2765      if (gorby_mcast)
2766      {
2767          /* Run this loop if it is not the first time. */
2768          /* First record is current node info, and will be sent anyhow. */
2769          if (q_rcvd > 1)
2770          {
2771              /* Search to find the first valid record to send. */
2772              /* Color should not be black, and record should not be expired one */
2773              do
2774              {
2775                  /* increase the sent record counter by one, and if it reaches */
2776                  /* to the end of GMI table, resets it to zero */
2777                  q_sent++;
2778                  if (q_sent == q_rcvd)
2779                      q_sent = 0;
2780
2781                  /* Extract color index and ORG from the record */
2782                  1st_byte = gmi [q_sent] - 0x10000 * floor (gmi [q_sent] / 0x10000);
2783                  test_col = floor (1st_byte / 0x100);
2784                  test_org = 1st_byte - 0x100 * floor (1st_byte / 0x100);
2785              } while ( (test_col == G_None) || (test_org == ORG_EXP) );
2786
2787              /* q_sent points to the record that supposed to be sent */
2788              /* It loads hello packet with this record. */
2789              my_test3 = gmi [q_sent];
2790
2791              hello_msg_ds_ptr->options_tlvptr->option_type      = 3;
2792              hello_msg_ds_ptr->options_tlvptr->option_len      = 8;
2793              hello_msg_ds_ptr->options_tlvptr->option_value    = my_test3;
2794          }
2795      }
2796      //GORby End
2797
2798      /* Create a PIM-SM message data structure */
  
```

Figure 35: Sending GMI table record

Figure 36 presents required variable for receiving hello packet function, and Figure 37 and Figure 38 show additional development of this function. Calculating GIA parameter is presented in Figure 39.


```

1997     FOUT;
1998 }
1999
2000 static void
2001 ip_pim_sm_hello_msg (void)
2002 {
2003     /* Received a Hello message from a PIM-SM neighbor router. Process */
2004     /* this message and use it to determine who the DR for the incoming */
2005     /* interface will be. */
2006
2007     {
2008         IpT_Address          nbr_src_address = IPC_ADDR_INVALID;
2009         IpT_Pim_Sm_Nbr_Entry* nbr_entry_ptr = OPC_NIL;
2010         IpT_Pim_Sm_Dr_Entry*  dr_entry_ptr = OPC_NIL;
2011         IpT_Pim_Sm_Hello_Msg* hello_msg_ds_ptr = OPC_NIL;
2012         IpT_Pim_Sm_Msg*       pim_sm_msg_ds_ptr = OPC_NIL;
2013         int                   nbr_holdtime, nbr_priority;
2014         char                  err_str [256];
2015
2016         // GORby variables block start
2017
2018         int j;
2019         double my_test;
2020         double test_org;
2021         double test_col;
2022         double test_x;
2023         double test_y;
2024         double msg_org;
2025         double msg_col;
2026         double msg_x;
2027         double msg_y;
2028
2029         // GORby variables block end
2030
2031
2032     }
2033     FIN (ip_pim_sm_hello_msg ());
  
```

Line: 2028

Figure 36: Variable definition for hello message receive function

```

2090 /* packet that is being received. Get this information from the */
2091 /* packet now. */
2092 op_pk_nfd_access (pimsm_pkptr, "message", &pim_sm_msg_ds_ptr);
2093 hello_msg_ds_ptr = (IpT_Pim_Sm_Hello_Msg *) pim_sm_msg_ds_ptr->msg_ds_ptr;
2094 nbr_holdtime = hello_msg_ds_ptr->options_tlvptr [holdtime].option_value;
2095
2096 /* Also store the neighbor's priority information. */
2097 nbr_priority = hello_msg_ds_ptr->options_tlvptr [priority].option_value;
2098
2099 // GORby
2100
2101 if (gorby_mcast)
2102 {
2103     /* Extract GORby information from hello packet, if this is */
2104     /* a GORby node. Update the GMI table based on received info */
2105     /* - First check if received info color is black. If yes, no */
2106     /* update is required. */
2107     /* - Check to see it has already the record for this node. */
2108     /* - If has it, check to see it came from upstream, and if */
2109     /* yes and also sending own info, extract the color index, */
2110     /* and increment it set node color index, and update first */
2111     /* record. Also update found record color as well. */
2112     /* - If record does exist, add it to GMI table, and increase */
2113     /* end of table pointer. If it has came from upstream node */
2114     /* about itself, extract the color and update own color. */
2115     my_test = hello_msg_ds_ptr->options_tlvptr->option_value;
2116     msg_y = floor (my_test / 0x100000000);
2117     msg_x = floor ((my_test - (msg_y * 0x100000000)) / 0x10000);
2118     msg_col = floor ((my_test - (msg_y * 0x100000000) - (msg_x * 0x10000)) / 0x100);
2119     msg_org = my_test - 0x100 * floor (my_test / 0x100);
2120
2121     /* Check if record already exist */
2122     if (!(msg_col == G_None))
2123     {
2124         for (j=0; j < g_rcvd; j++)
2125         {
2126             test_y = floor (gmi [j] / 0x100000000);
2127             test_x = floor ((gmi [j] - (test_y * 0x100000000)) / 0x10000);
2128             test_col = floor ((gmi [j] - (test_y * 0x100000000) - (test_x * 0x10000)) / 0x100);
2129             test_org = gmi [j] - 0x100 * floor (gmi [j] / 0x100);
2130
2131             if ((msg_x == test_x) && (msg_y == test_y))
2132             {
2133                 gmi [j] = 0x100 * floor (my_test / 0x100);
2134
2135                 if ((nbr_src_address == g_up_address) && (msg_org == ORG_ME))
2136                 {
2137                     /* It is from upstream node about itself, update my color */
2138                     g_color = msg_col + 1;
2139                     gmi [0] = 0x10000 * floor (gmi [0] / 0x10000);
2140                     gmi [0] = gmi [0] + (g_color * 0x100) + ORG_ME;
2141                     gmi [j] = gmi [j] + ORG_UP;
2142                 }
2143             }
2144             else
2145             {
2146                 /* It is not about upstream node */
2147                 if (j == 0)
2148                 {
2149                     /* This is the first record, add the first record of

```

Figure 37: Receive hello packet function (1)

```

ip_pim_sm.function block
File Edit Options

145         else
146             /* It is not about upstream node */
147             {
148                 if (j == 0)
149                     /* It is about the first node, set to OWN info */
150                     gmi[j] = gmi[j] + ORG_ME;
151                 else
152                     /* It is about other nodes, set to Others */
153                     gmi[j] = gmi[j] + ORG_OT;
154             }
155         break;
156     }
157 }
158
159 /* Record does not exist (it is new info). Adds it to GMI table */
160 /* and increases last record pointer. */
161 if (j == g_rcvd)
162 {
163     gmi[j] = 0x100 * floor(my_test / 0x100);
164     if ((nbr_src_address == g_up_address) && (msg_org == ORG_ME))
165     {
166         /* It is from upstream node about itself, update my color */
167         g_color = msg_col + 1;
168         gmi[0] = 0x10000 * floor(gmi[0] / 0x10000);
169         gmi[0] = gmi[0] + (g_color * 0x100) + ORG_ME;
170         gmi[j] = gmi[j] + ORG_UP;
171     }
172     else
173     {
174         /* It is not about upstream node, set to Others */
175         gmi[j] = gmi[j] + ORG_OT;
176     }
177     g_rcvd++;
178 }
179 }
180
181 // GORby
182
183
184
185
186
187
188
189 /* See if there is already an entry in the neighbor list for */
190 /* the node that sent this Hello message. */
191 nbr_entry_ptr = ip_pim_sm_nbr_entry_get(nbr_lptr, nbr_src_address);
192
193 /* If this node has already received a Hello message from this */
194 /* neighbor, then the information specific to this neighbor */
195 /* should be updated in the neighbor table. */
196 if (nbr_entry_ptr != OPC_NIL)
197 {
198

```

Figure 38: receive hello packet function (2)

```

ip_pim_sm.function block
File Edit Options

508 x_pos = floor ((gmi [0] - (y_pos * 0x100000000)) / 0x10000);
509 sprintf (c1, "[%d]%,%,%,x:%4.2f, y:%4.2f", g_color, G_color_str, G_up_addr_str, x_pos, y_pos);
510 Oma_Ot_Table_Cell_Value_Set (pim_rp_rte_table, ip_pim_sm_rp_table_columns_name_array [IpC_Pim
511
512 /* Calculating gia metric based on GMI table information. */
513 /* Start from second record (first is own node information. */
514 /* - If it is upstream node, calculates nominator factor. */
515 /* - If it has the same color as upstream node, uses it to */
516 /* calculate denominator part of gia. */
517 /* - If at the end denominator is still zero, meaning there */
518 /* is no subnet members around, so set gia to infinity. */
519 gia_d = (double) 0;
520 for (j=1; j<g_rcvd; j++)
521 {
522 test_y = floor (gmi [j] / 0x100000000);
523 test_x = floor ((gmi [j] - (test_y * 0x100000000)) / 0x10000);
524 test_col = floor ((gmi [j] - (test_y * 0x100000000) - (test_x * 0x10000)) / 0x100);
525 test_org = gmi [j] - 0x100 * floor (gmi [j] / 0x100);
526
527 if (test_org == ORG_UP)
528 {
529 up_xpos = test_x;
530 up_ypos = test_y;
531 gia_n = (double) 1 / ((pow((x_pos/1000 - up_xpos/1000), 2) +
532 pow((y_pos/1000 - up_ypos/1000), 2))); // 1 / 0^2
533 }
534 else
535 {
536 n_color = test_col;
537 dis_color = abs (n_color - (g_color - 1)); // (g_color - 1) = upstream color
538 sn_color = dis_color - 3 = floor(dis_color/3);
539
540 if (sn_color == 0)
541 {
542 n_xpos = test_x;
543 n_ypos = test_y;
544 gia_d = (double) gia_d + (double) (1 / ((pow((x_pos/1000 - n_xpos/1000), 2)
545 + pow((y_pos/1000 - n_ypos/1000), 2)))); // 1 / 0^2
546 }
547 }
548 }
549
550 if (gia_d == 0)
551 gia = (double) pow (10,10);
552 else
553 gia = (double) (gia_n / gia_d);
554
555 // GORby End
556
557 }
558 Oma_Ot_Table_Cell_Value_Set (pim_rp_rte_table, ip_pim_sm_rp_table_columns_name_array [IpC_Pim
559 }
560 else
561

```

Figure 39: Calculating GIA based on GMI table records

Appendix B

OPNET Reports

In this Appendix, some of the OPNET reports that we generated and have not used them directly in our evaluation chapter are presented.

We follow our same general scenarios naming, which are also presented in graphs, includes three sections in a format of X-Y_Z:

- X: Specifies Multicast protocol and seed value, based on Table 5.
- Y: Specifies the traffic, based on Table 4 “Symbol” column.
- Z: Specifies number of active receivers; for example 3R for three receivers, 4R for four receivers and 10R for ten receivers, and the rest of the receivers are off in each scenario.

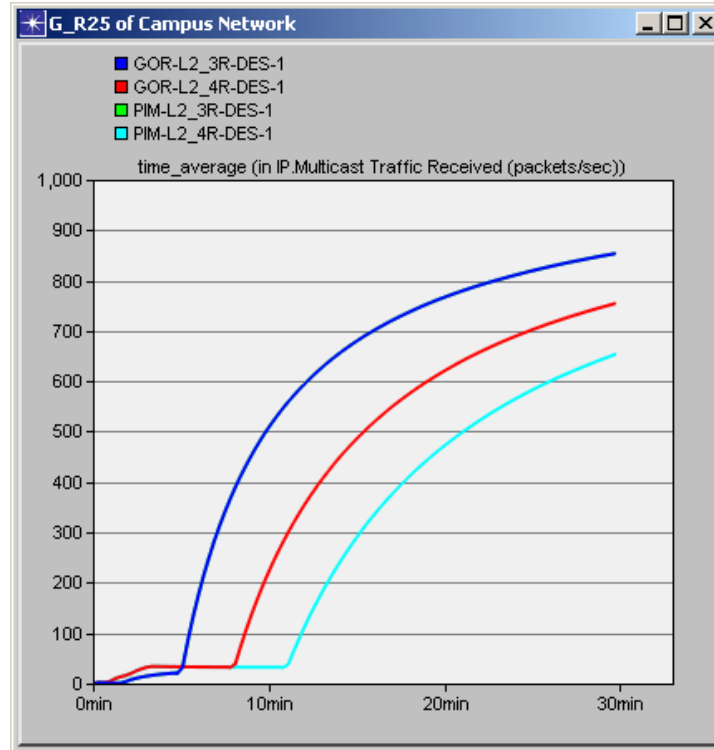


Figure 40: Average traffic received by G_R25

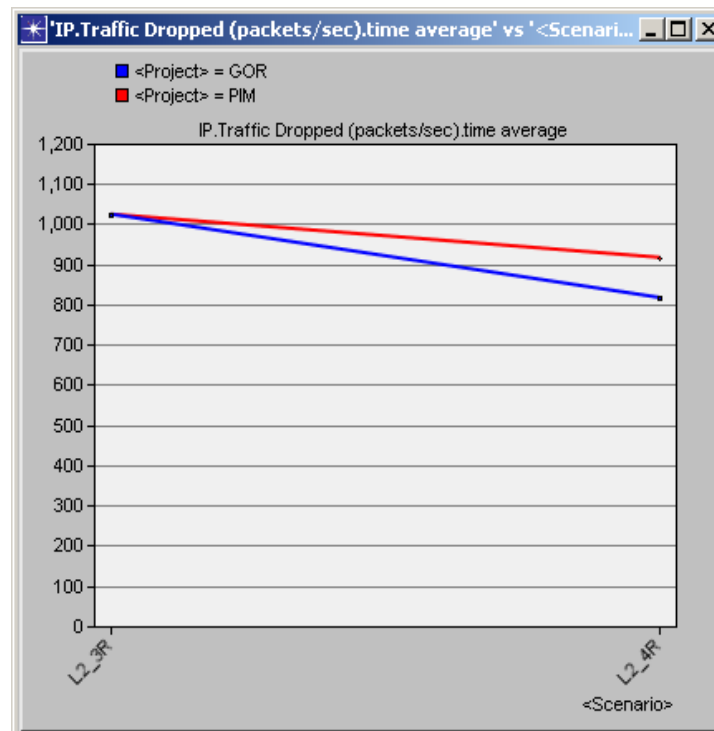


Figure 41: Average traffic dropped the entire nodes

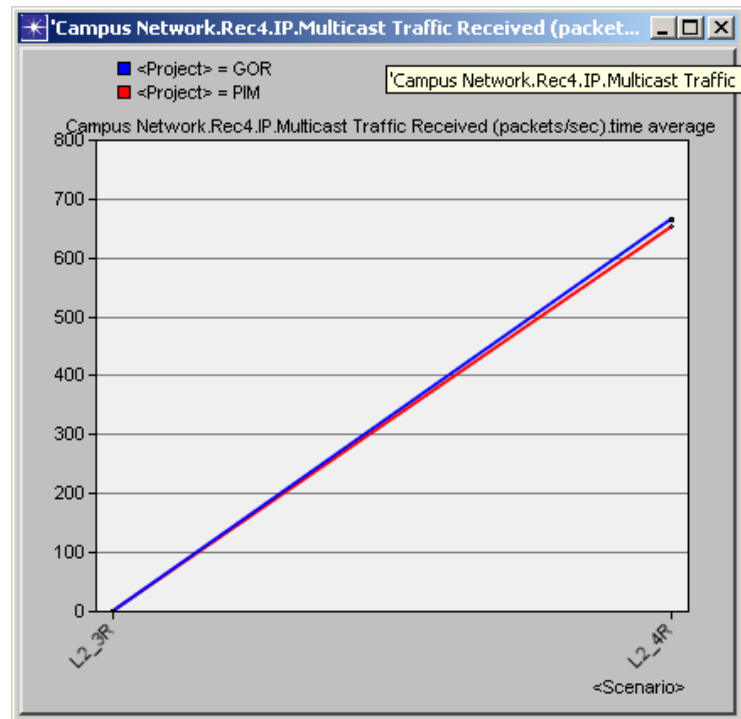


Figure 42: Average Multicast traffic received by Rec4

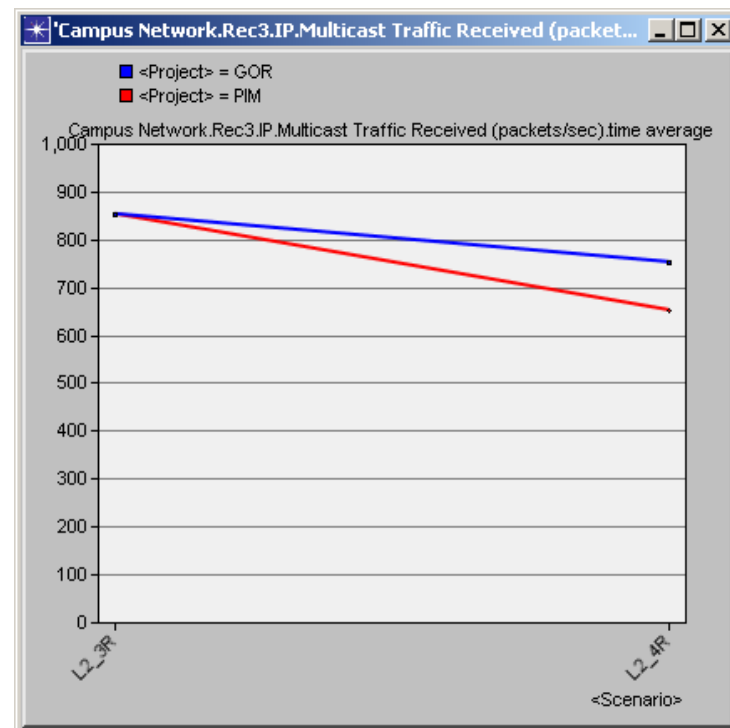


Figure 43: Average Multicast traffic received by Rec3

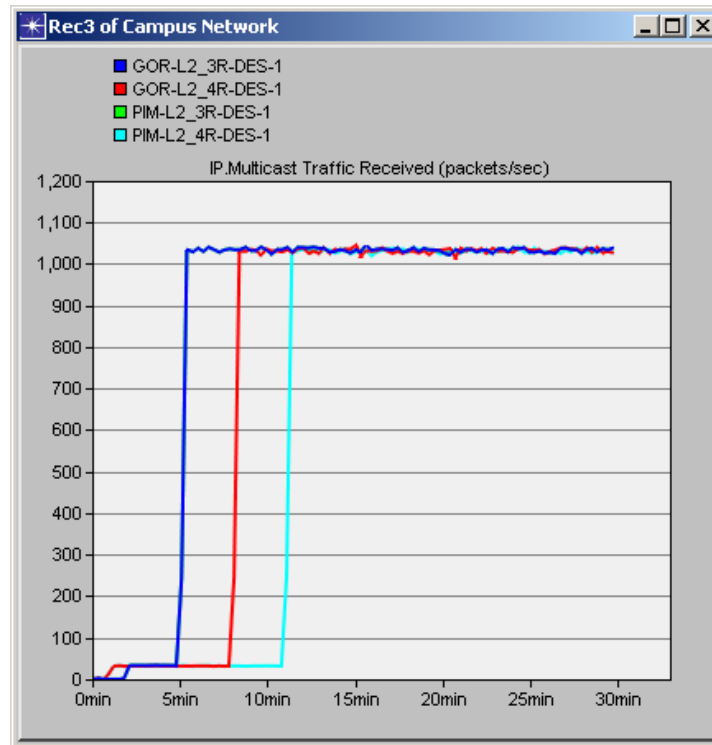


Figure 44: Multicast traffic received by Rec3

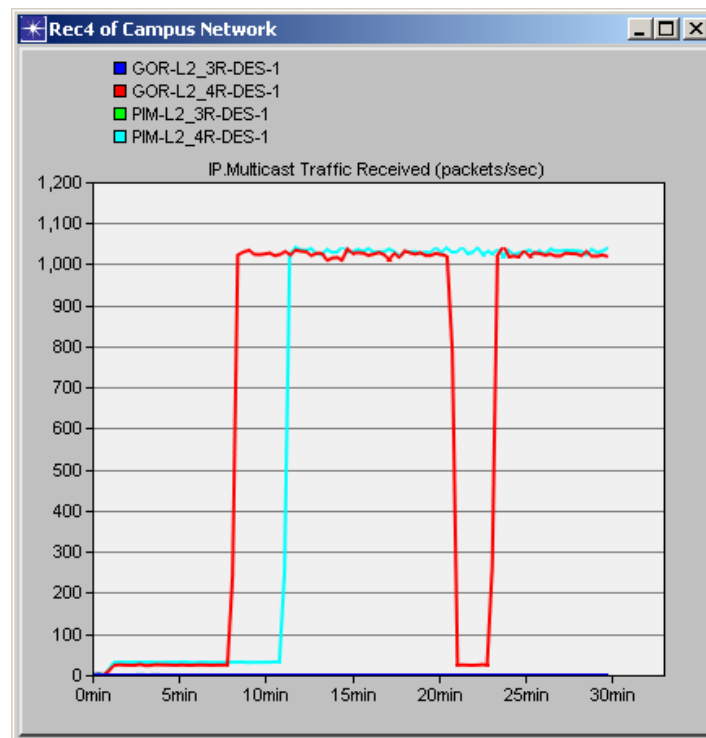


Figure 45: Multicast traffic received by Rec4

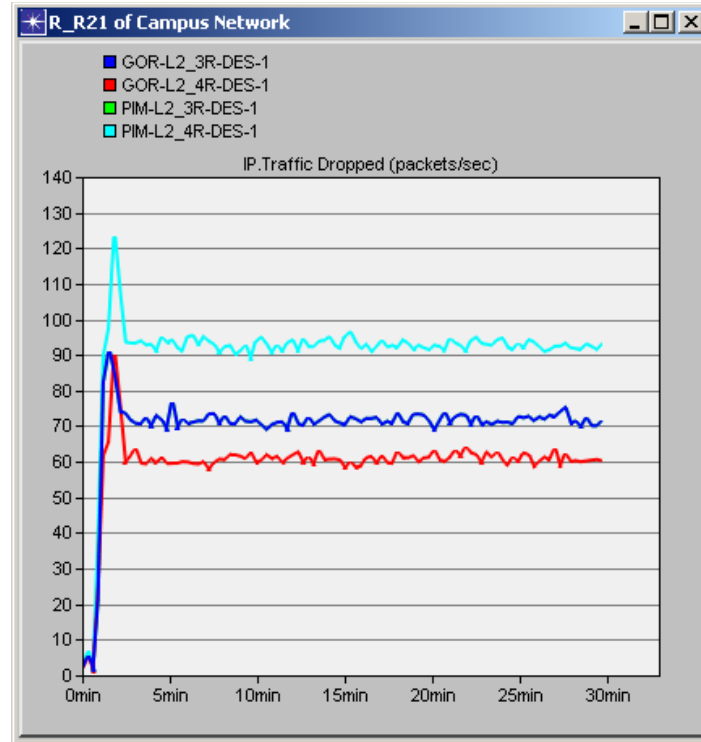


Figure 46: Traffic dropped by R_R21

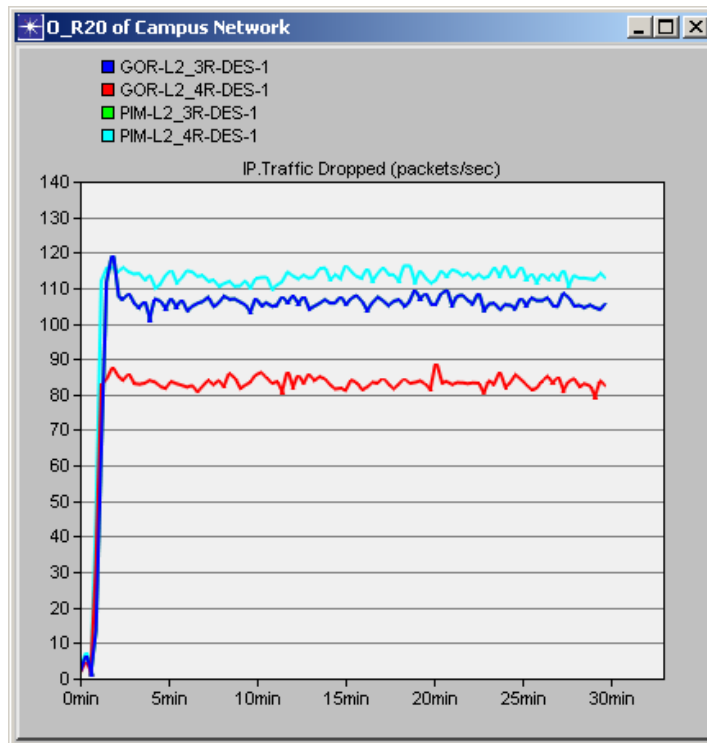


Figure 47: Traffic dropped by O_R20

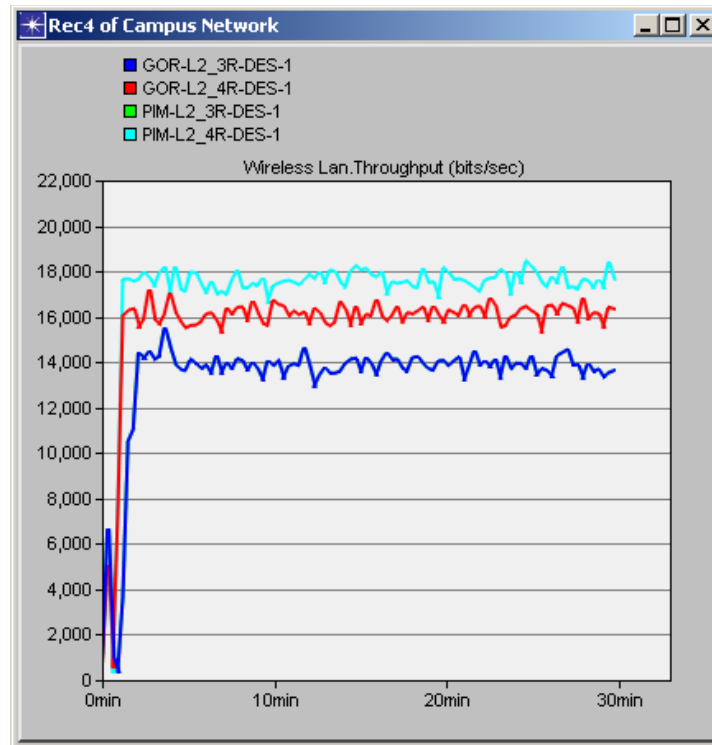


Figure 48: Rec4 throughput (bits/sec)

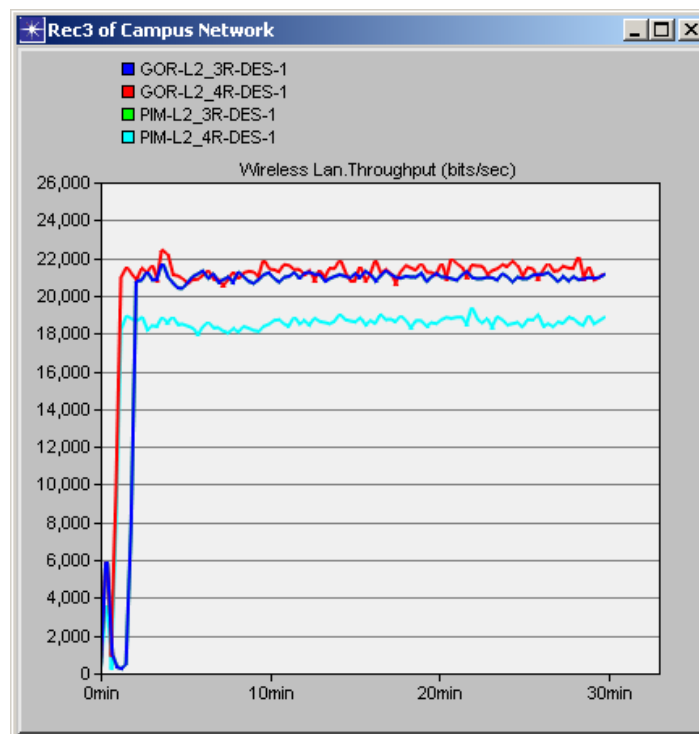


Figure 49: Rec3 throughput (bits/sec)

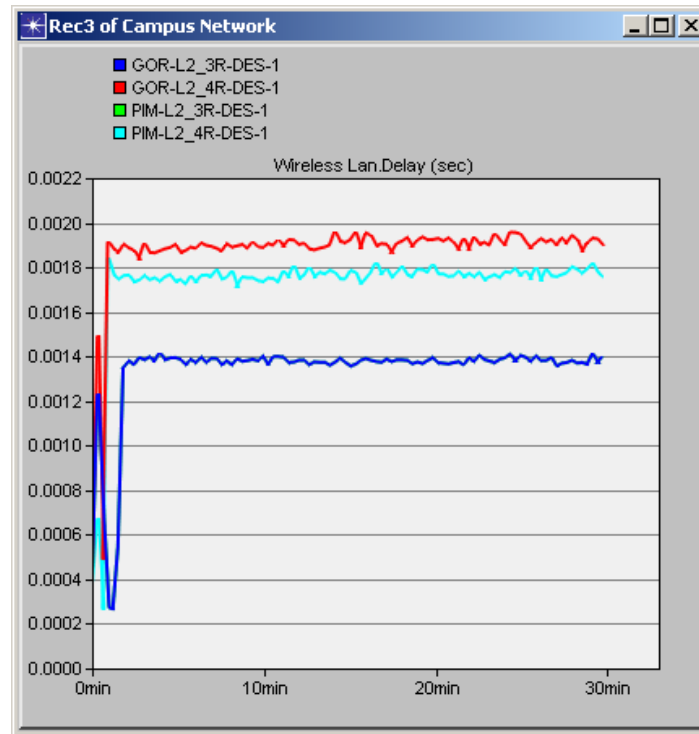


Figure 50: Rec3 delay (Sec)

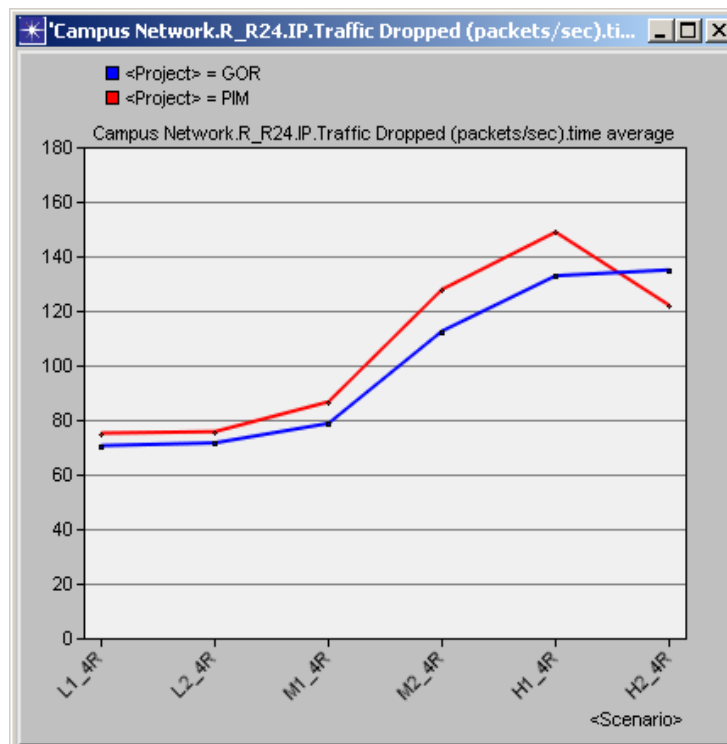


Figure 51: Average traffic dropped by R_R24

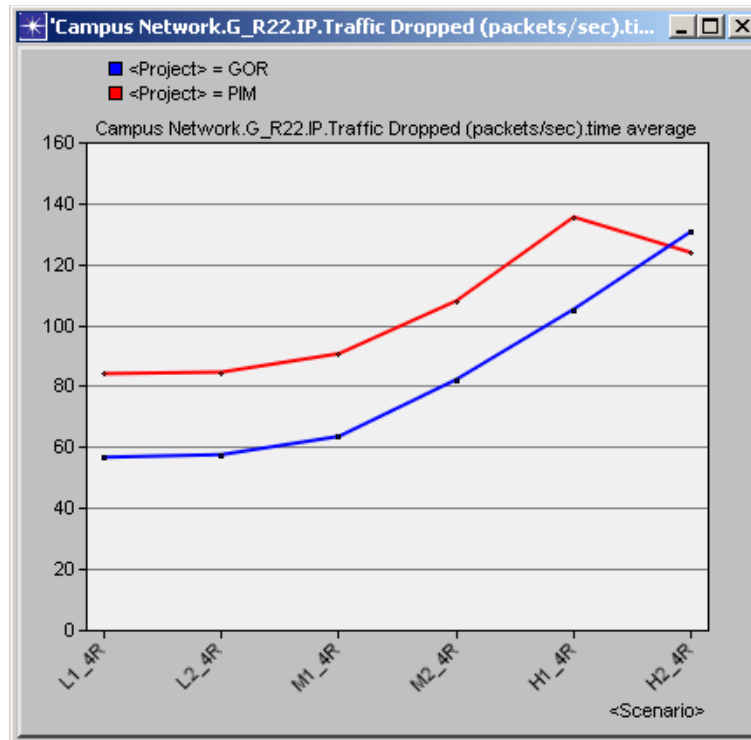


Figure 52: Average traffic dropped by G_R22

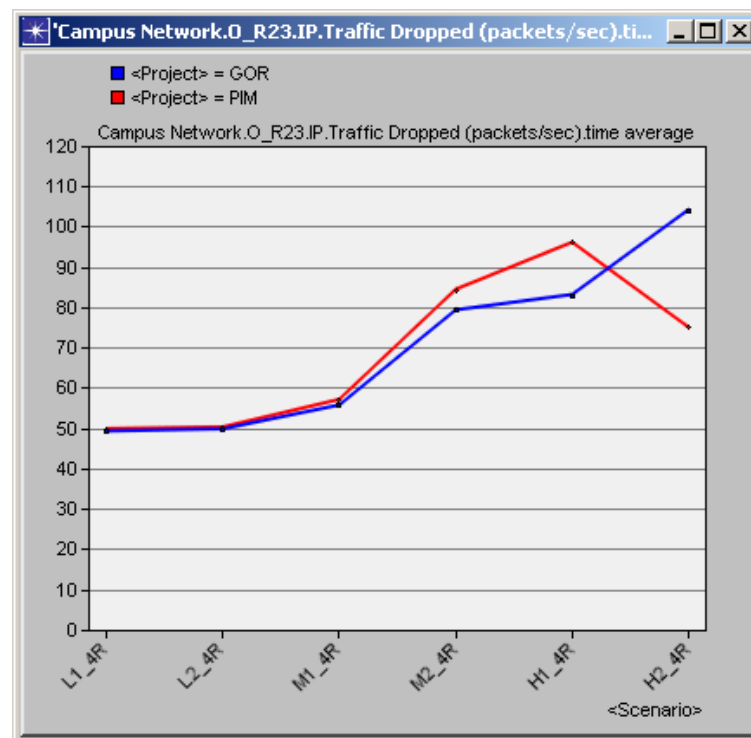


Figure 53: Average traffic dropped by O_R23

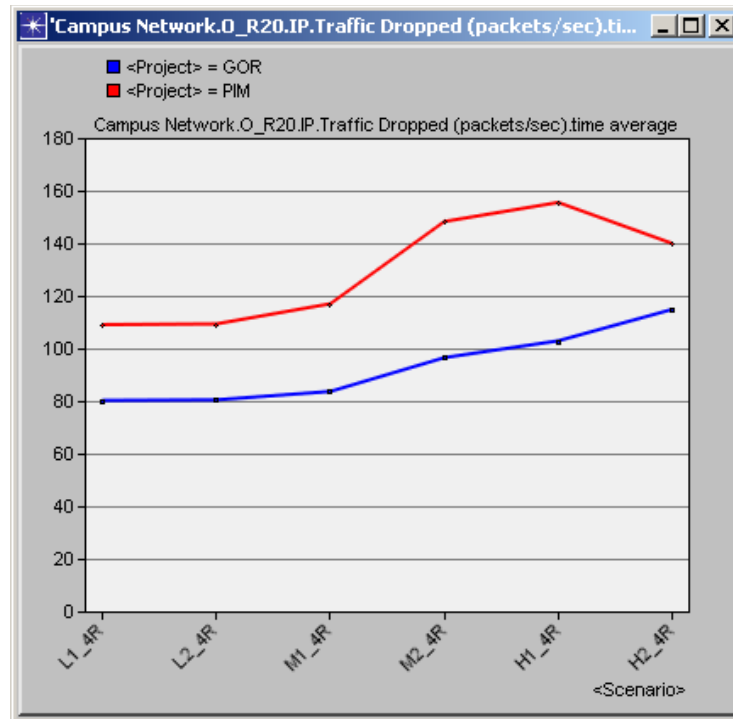


Figure 54: Average traffic dropped by O_R20

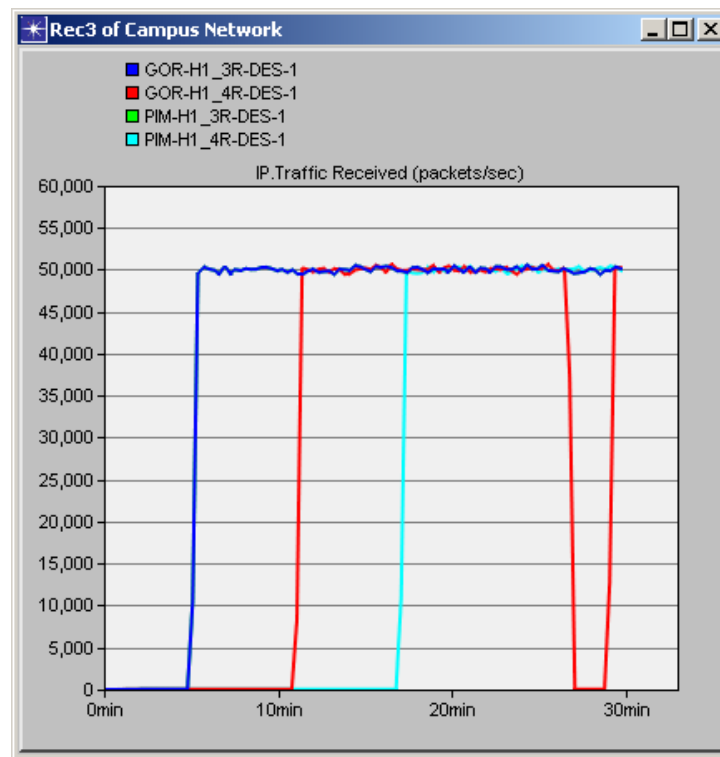


Figure 55: Rec3 traffic received (Packets/sec)

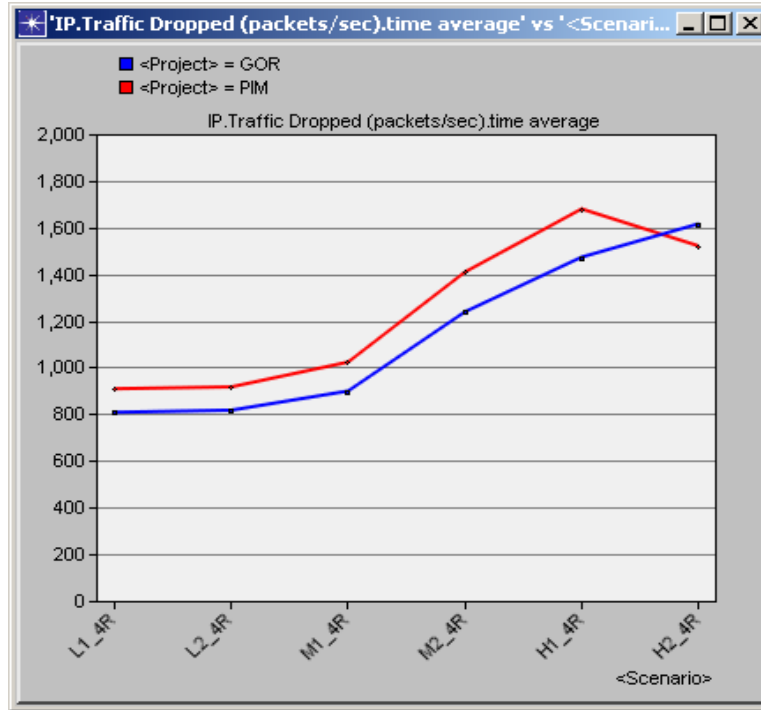


Figure 56: Average traffic dropped by entire nodes

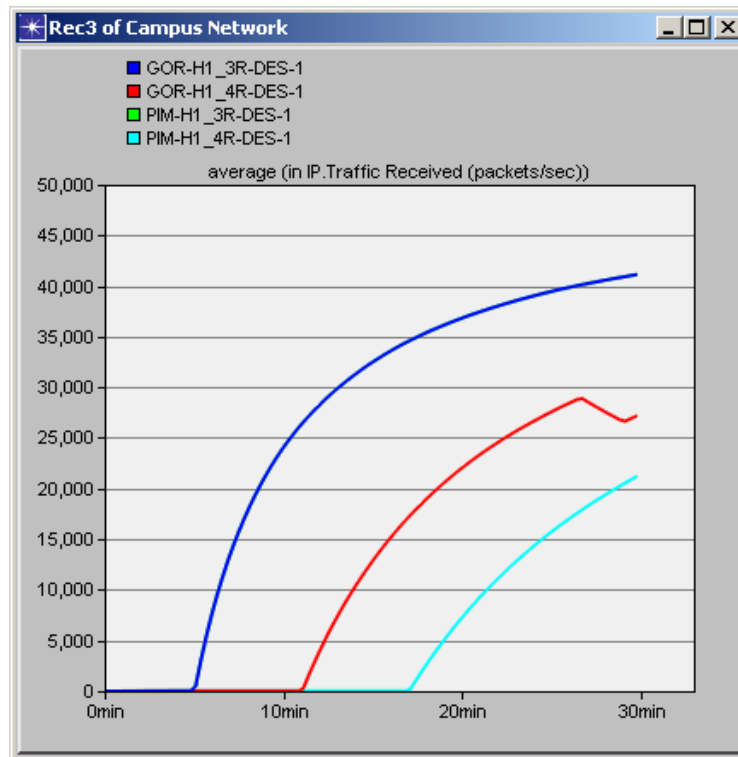


Figure 57: Average traffic received by Rec3 (Packet/Sec)

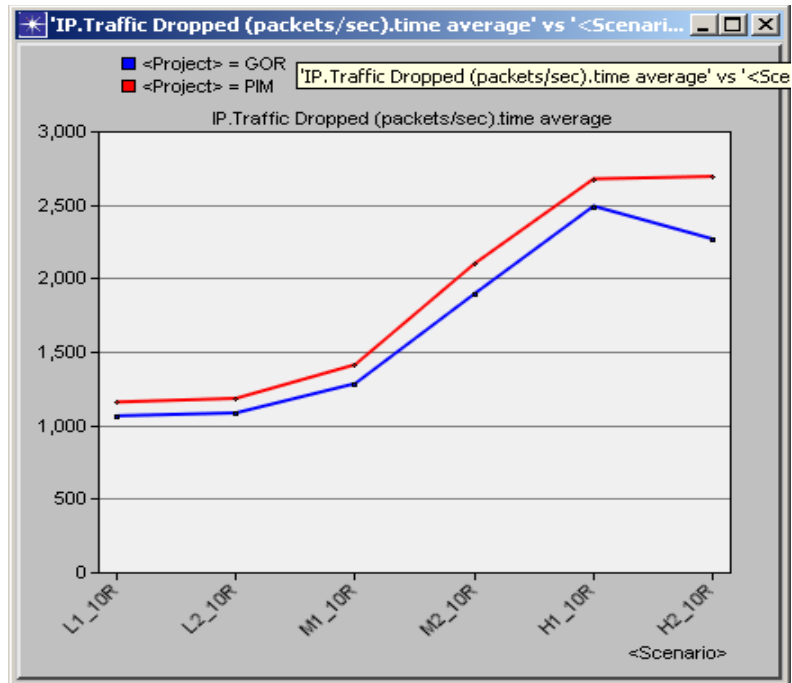


Figure 58: Average traffic dropped by entire nodes

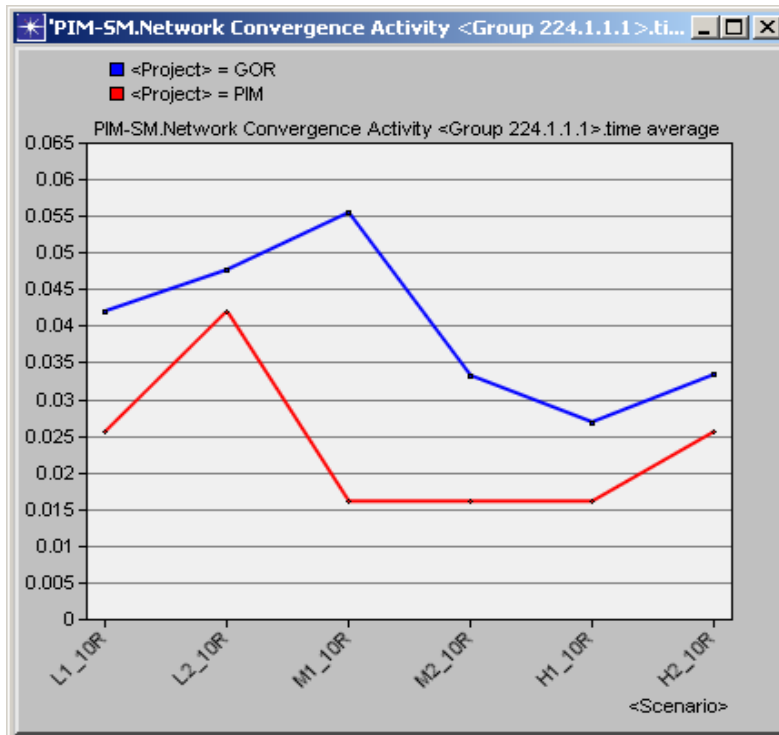


Figure 59: Average convergence activity by entire nodes

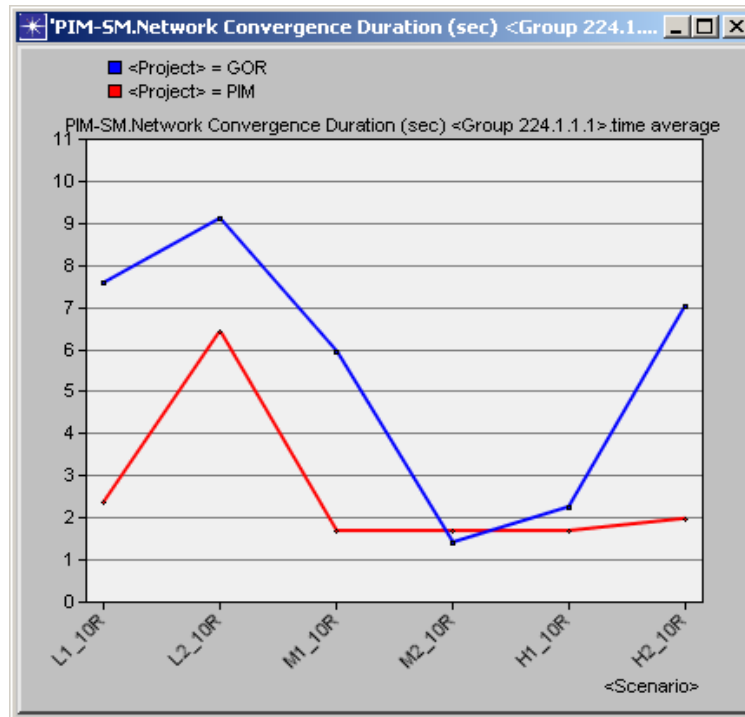


Figure 60: Average convergence duration (Entire nodes)

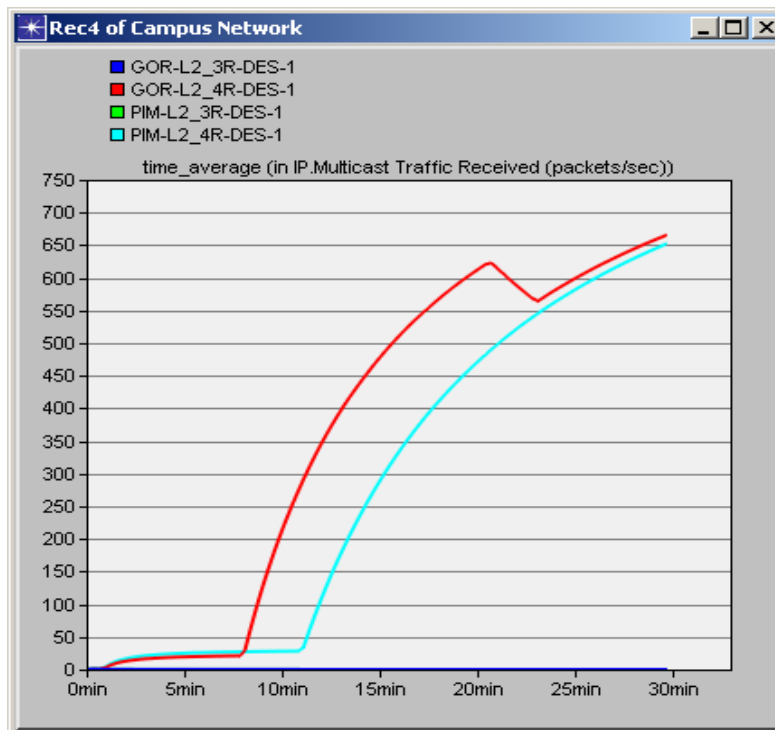


Figure 61: Average traffic received by Rec4 (Packet/Sec)

References

- [1] RFC 3626, “*Optimized Link State Routing Protocol (OLSR)*”, www.ietf.org.
- [2] RFC4601, Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas “*Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification (Revised)*”, www.ietf.org, August 2006.
- [3] A. Laouiti, P. Jacquet, P. Minet, L. Viennot, T. Clausen and C. Adjith, “*Multicast Optimized Link State Routing*,” Report No 4721, INRIA.
- [4] J. Wang, M. Song and Y. Zhao, “*Interference-Aware Multicast in Wireless Mesh Network with Directional Antenna*”, in Proc. Fifth IEEE International Conference on Networking, Architecture, and Storage, 15-17 July 2010.
- [5] I. Tinnirello and G. Bianchi, “*Interference Estimation in IEEE 802.11 Networks*”, IEEE Control Systems Magazine, pp 30-43, Issue 2, Vol. 30, April 2010.
- [6] J.J. Garcia-Luna-Aceves and D. Sampath, “*Efficient Multicast Routing in MANETs Using Prefix Labels*”, in Proc. 18th International Conference on Computer Communications and Networks, Palo Alto, CA, USA, 3-6 Aug. 2009.

- [7] G. Bianchi, “*Performance Analysis of the IEEE 802.11 Distributed Coordination Function*”, Proceeding of IEEE Journal on Selected Areas in Communications, pp 537-547, Vol 18, No. 3, March 2000.
- [8] H. Zhai, X. Chen and Y. Fang, “*How Well Can the IEEE 802.11 Wireless LAN Support Quality of Service?*,” Proceeding of IEEE Transactions on Wireless Communications, pp 3084 – 3094, Vol. 4, No. 6, November 2005.
- [9] M. B. Krishna and M. N. Doja, “*Performance of Structure Based Multicast Routing Protocols in Wireless Ad Hoc Networks*”, in Proc. 10th IEEE International Conference on Computer and Information Technology (CIT 2010), Delhi, India June 29-July 1 2010.
- [10] A. Iyer, C. Rosenberg and A. Karnik, “*What is the Right Model for Wireless Channel Interference?*”, Proceeding of IEEE Transactions on Wireless Communication, pp 2662 – 2671, Vol. 8, No. 5, May 2009.
- [11] D. M. Blough, G. Resta and P. Santi, “*Approximation Algorithm for Wireless Link Scheduling With SINR-Based Interference*”, Proceeding of IEEE/ACM Transactions on Networking, pp 1701 – 1712, Issue 6, Vol. 18, 2010.
- [12] Y. Yu, Y. Zhao and S. Du, “*Providing Interference-aware Quality of Service Support for ODMRP*”, in Proc. Asia-Pacific Conference on Information Processing, Nanjing, China, 18-19 July 2009.
- [13] Z. Jia, R. Gupta, J. Walnard and P. Varaiya, “*Bandwidth Guaranteed Routing for Ad-Hoc Networks with Interference Consideration*”, in Proc. 10th IEEE Symposium on Computers and Communications (ISCC 2005), Berkeley, CA, USA, 27-30 June 2005.
- [14] M.S. Kakkasageri, S.S. Manvi and R.S. Gada, “*QoS Aware Multicast Routing in MANETs*”, in Proc. 10th IEEE Region Conference - TENCON , 19-21 Nov. 2008.
- [15] M. Pandey and D. Zappala, “*Scalable Multicast Routing for Ad Hoc Networks*”, in Proc. 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, Sept. 29 2008-Oct. 2 2008.

- [16] W. Yang and C. Tung, “*An Interference-free Multicast for Mesh Networks with Directional Antennas*”, in Proc. First International Conference on Ubiquitous and Future Networks ICUFN 2009, Tainan, Taiwan, 7-9 June 2009.
- [17] Y. Yu, Y. Zhou and S. Du, “*Impact of Interference on Multicast Throughput over AD Hoc Networks*”, in Proc. Fifth International Joint Conference on INC, IMS and IDC, Nanjing, China, 25-27 Aug. 2009.
- [18] S. Zhao, A. Aggarwal, S. Liu and H. Wu, “*A Secure Routing Protocol in Proactive Security Approach for Mobile Ad-Hoc Networks*”, in Proc. Wireless IEEE Conference on Communications and Networking WCNC 2008, Windsor, ON, Canada, March 31 2008-April 3 2008.
- [19] Y. Yang and J. Wang, “*Design Guideline for Routing Metrics in Multihop Wireless Networks*”, in Proc. IEEE INFOCOM 27th Conference on Computer Communications, Blacksburg, VA, USA, 13-18 April 2008.
- [20] Y. Im, J. Lee, J. Han, S. Lee and T. Kwon, “*FIRM: Flow-based Interference –aware Route Management in Wireless Mesh Networks*”, in Proc. 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops, Seoul, South Korea, 22-26 June 2009.
- [21] S. Lee, Hyukjoon Lee and Hyungkeun Lee, “*A Cross Layer Routing Metric To Recognize Traffic Interference in Wireless Mesh Networks*”, in Proc. IEEE International Conference in Sensor Networks, Ubiquitous and Trustworthy Computing, 7-9 June 2010.
- [22] Y. Jia and Y. Hu, “*Improvement of Wireless Multicast Routing with Link State Based on MAODV*”, in Proc. 4th International Conference on Wireless Communications, Networking and Mobile Computing WiCOM '08, 12-14 Oct. 2008.
- [23] X. Chen, Q. Zhong and D. Liu, “*An Improved MAODV Protocol Based on Mobility Prediction and Self-pruning Flooding*”, in Proc. International Conference on Communication and Mobile Computing, Beijing, China, Jan. 6-8, 2009.

- [24] S. Roy, D. Koutsonikolas, S. Das and Y. C. Hu, “*High-Throughput Multicast Routing Metrics in Wireless Mesh Networks*”, in Proc. 26th IEEE International Conference on Distributed Computing Systems, West Lafayette, IN, USA, 2006.
- [25] D. Koutsonikolas, Y. C. Hu and C. Wang, “*Pacifier: High-Throughput, Reliable Multicast without “Crying Babies” in Wireless Mesh Networks*”, in Proc. IEEE INFOCOM, West Lafayette, IN, USA, April 19-25, 2009.
- [26] P. H. Pathak and R. Dutta, “*A Survey of Network Design Problems and Joint Design Approaches in Wireless Mesh Networks*”, Proceeding of IEEE Communications Surveys & Tutorials, pp 1-33, Issue 99, June 2010.
- [27] L. Junahai, Y. Danxia, X. Liu and F. Mingyu, “*A Survey of Multicast Routing Protocols for Mobile Ad-Hoc Networks*”, Proceeding of IEEE Communications Surveys & Tutorials, pp 78-91, Issue 1, Vol. 11, First Quarter 2009.
- [28] X. Xiang, X. Wang, and Y. Yang, “*Supporting Efficient and Scalable Multicasting over Mobile Ad Hoc Networks*”, Proceeding of IEEE TRANSACTIONS ON MOBILE COMPUTING, VOL. 10, NO. 5, APRIL 2011.
- [29] Robert J. Hall, “*An Improved Geocast for Mobile Ad Hoc Networks*”, Proceeding of IEEE TRANSACTIONS ON MOBILE COMPUTING, VOL. 10, NO. 2, FEBRUARY 2011.
- [30] Y. Yang, J. Wang and R. Kravets, “*Designing Routing Metrics for Mesh Networks*”, in Proc. IEEE Workshop on Wireless Mesh Networks, 2006.
- [31] M. Jaseemuddin, “*Wireless Course notes*”, Ryerson University.
- [32] V. C.M. Leung, “*Mobile Communication Course notes*”, University of British Columbia.