

M.Sc. ENGG. THESIS

A NEW APPROACH FOR SELECTING  
AGGREGATED MULTICAST TREES TO  
REDUCE FORWARDING STATES

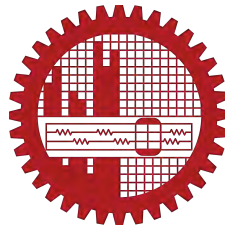
by

Masud Rana (0412052008 P)

Submitted to

Department of Computer Science and Engineering

(In partial fulfillment of the requirements for the degree of  
Master of Science in Computer Science and Engineering)



Department of Computer Science and Engineering

Bangladesh University of Engineering and Technology (BUET)

Dhaka 1000

August 27, 2018

*Dedicated to my loving parents*

## AUTHOR'S CONTACT

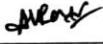
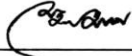

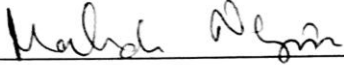

---

Masud Rana

Email: [masud.rana@dutchbanglabank.com](mailto:masud.rana@dutchbanglabank.com)

The thesis titled "A NEW APPROACH FOR SELECTING AGGREGATED MULTI-CAST TREES TO REDUCE FORWARDING STATES", submitted by Masud Rana, Roll No. 0412052008 P, Session April 2012, to the Department of Computer Science & Engineering, Bangladesh University of Engineering & Technology, has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Master of Science in Computer Science & Engineering and approved as to its style and contents. Examination held on August 27, 2018.

## Board of Examiners

1.   
\_\_\_\_\_  
Dr. A. B. M. Alim Al Islam  
Associate Professor  
Department of Computer Science & Engineering  
Bangladesh University of Engineering & Technology, Dhaka.  
Chairman  
(Supervisor)
2.   
\_\_\_\_\_  
Prof. Dr. Md. Mostofa Akbar  
Head and Professor  
Department of Computer Science & Engineering  
Bangladesh University of Engineering & Technology, Dhaka.  
Member  
(Ex-Officio)
3.   
\_\_\_\_\_  
Prof. Dr. M. Sohel Rahman  
Professor  
Department of Computer Science & Engineering  
Bangladesh University of Engineering & Technology, Dhaka.  
Member
3.   
\_\_\_\_\_  
Prof. Dr. Mahmuda Naznin  
Professor  
Department of Computer Science & Engineering  
Bangladesh University of Engineering & Technology, Dhaka.  
Member
4.   
\_\_\_\_\_  
Dr. Lutfa Akter  
Associate Professor  
Department of Electrical and Electronic & Engineering  
Bangladesh University of Engineering & Technology, Dhaka.  
Member  
(External)

## Candidate's Declaration

This is hereby declared that the work titled "A NEW APPROACH FOR SELECTING AGGREGATED MULTICAST TREES TO REDUCE FORWARDING STATES", is the outcome of research carried out by me under the supervision of Dr. A. B. M. Alim Al Islam, in the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka 1000. It is also declared that this thesis or any part of it has not been submitted elsewhere for the award of any degree or diploma.



---

Masud Rana

Candidate

# Acknowledgment

First of all, I would like to express my heart-felt gratitude to my supervisor, Dr. A. B. M. Alim Al Islam, for his constant supervision of this work. He helped me a lot in shaping, deciding steps of my work, and providing infrastructural supports.

I would also want to thank the honorable members of my thesis committee: Prof. Dr. Md. Mostofa Akbar, Prof. Dr. M. Sohel Rahman, Prof. Dr. Mahmuda Naznin, and Dr. Lutfa Akter for their encouragements, insightful comments, and valuable suggestions.

I am also grateful to all honorable teachers of the department for their comments and suggestions. I would like to give special thanks to Prof. Dr. M. Kaykobad and Mohammad Saifur Rahman for their valuable suggestions during my thesis work.

Last but not the least, I remain ever grateful to my beloved parents, wife and family, for their inspirations behind every success of mine.

# Abstract

Multicast is an efficient method for sending data to multiple destinations in a single transmission. Conventional multicast routing often suffers from scalability issues. Here, the number of forwarding states maintained in the Network layer generally increases with the number of concurrently active multicast groups. Consequently, large-size routing tables get generated, which in turn causes degradation in router performance. There exist only a limited number of studies on aggregated multicast to address this issue. The existing studies such as AM (Aggregated Multicast), STA (Scalable Tree Aggregation for Multicast), and STS (Shared-Tree Selection for Aggregated Multicast) attempt to perform tree aggregation, however, still retain high number of forwarding states in the routers. Besides, they mostly ignore the effect of tree aggregation ratio in terms of network performance metrics (for example delay and throughput). Hence, the research on aggregated multicast is still at an elementary stage. To this extent, in this study, we propose a novel aggregated multicast approach to reduce Network layer forwarding states. Our approach proposes new methods for selecting aggregated multicast trees, refining search range, and replacing stale trees in process of the aggregation. We show the effect of our proposed tree aggregation in terms of network performance metrics (delay and throughput). We also show that performance of our approach is better than previous approaches by comparing different performance metrics such as the number of trees, the number of forwarding states, delay, and throughput. Ns-3 simulation results confirm that our approach can reduce up to 92% forwarding states and 34% delay compared to conventional multicast, and reduce up to 88% forwarding states and 29% delay compared to the STS method.

# Acronyms List

AM = Aggregated Multicast

STA = Scalable Tree Aggregation

STS = Shared Tree Selection Method

CBT = Core Based Tree

MPLS = Multi protocol level switching

PIM-SM = Protocol Independent Multicast- Sparse Mode

PIM-DM = Protocol Independent Multicast- Dense Mode

LSR = Level Switching Router

# Contents

<i>Board of Examiners</i>	ii
<i>Candidate's Declaration</i>	iii
<i>Acknowledgment</i>	iv
<i>Abstract</i>	v
<i>Acronyms List</i>	vi
<b>1 Introduction</b>	<b>1</b>
1.1 Existing Research Studies and Their Limitations . . . . .	3
1.2 Our Proposed Approach . . . . .	3
1.3 Performance Metrics for Multicast . . . . .	4
1.4 Our Contributions . . . . .	4
1.5 Outline of Our Thesis . . . . .	5
<b>2 Related Work</b>	<b>6</b>
2.1 Protocol Independent Multicast - Sparse Mode (PIM-SM) . . . . .	6
2.2 Core-Based Tree (CBT) . . . . .	8
2.3 MPLS Multicast Tree (MMT) . . . . .	9
2.4 Aggregated Multicast (AM) . . . . .	10
2.5 Scalable Tree Aggregation for Multicast . . . . .	12
2.6 Shared-Tree Selection Method for Aggregated Multicast . . . . .	12
2.7 Limitations of the Existing Studies . . . . .	13



<b>3</b>	<b>Proposed Mechanism for Selecting Aggregated Multicast Trees</b>	<b>14</b>
3.1	Overview on Proposed Aggregated Multicast . . . . .	14
3.2	Refining Search Range . . . . .	15
3.3	New Consideration over Overlapping Degree . . . . .	17
3.4	Replacement Method for Stale Aggregated Tree . . . . .	19
3.5	Overall Proposed Aggregation Approach . . . . .	21
<b>4</b>	<b>Experimental Evaluation</b>	<b>24</b>
4.1	Simulation Settings . . . . .	24
4.2	Performance Comparison with Existing Proposed Protocols . . . . .	26
4.2.1	Performance in Terms of The Number of Trees . . . . .	26
4.2.2	Performance in Terms of The Network Layer Forwarding State . . . . .	28
4.2.3	Performance in Terms of Delay . . . . .	30
4.2.4	Performance in Terms of Throughput . . . . .	33
4.2.5	Summary of Performance Evaluation . . . . .	35
4.3	Performance Analysis of Proposed Refinement on Search Range . . . . .	39
4.4	Performance Analysis of Proposed Consideration of Overlapping Degree . . . . .	42
4.5	Performance analysis of Proposed Replacement Method . . . . .	45
4.6	Variation in Simulation Time . . . . .	48
4.7	Variation in Data Rate . . . . .	51
4.8	Variation in Bandwidth . . . . .	53
4.9	Variation in Coincidence Degree (P) . . . . .	56
4.10	Variation in Overlapping Degree ( $\gamma$ ) . . . . .	58
4.11	Effects of Different Iteration . . . . .	60
4.12	Variation in Relative Numbers of Large and Small Groups . . . . .	62
4.13	Simulation Findings . . . . .	64
<b>5</b>	<b>Conclusion and Future Work</b>	<b>66</b>

# List of Figures

1.1	Applications of multicasting . . . . .	2
2.1	PIM-SM multicast . . . . .	8
2.2	Core-based tree [7] . . . . .	9
2.3	MPLS Multicast Tree [2] . . . . .	10
2.4	Aggregated multicast tree . . . . .	11
3.1	Multicast tree for newly arrived multicast group $G_0$ . . . . .	16
3.2	Trees in existing tree set T . . . . .	16
3.3	Tree $T_0$ is selected for newly arrived multicast group $G_0$ . . . . .	17
3.4	Multicast tree for newly arrived multicast group $G_0$ . . . . .	18
3.5	$T_0$ is a tree in the current tree set T . . . . .	18
3.6	Tree aggregation as per the proposed overlapping mechanism, as destination nodes of newly arrived multicast group $G_0$ overlap with that of tree $T_0$ . . . . .	18
3.7	Multicast tree for a newly arrived multicast group $G_2$ . . . . .	19
3.8	Trees in the current tree set T . . . . .	20
3.9	Trees of the tree set T after applying proposed replacement method . . . . .	21
3.10	Flow chart of our proposed aggregation multicast method . . . . .	23
4.1	Backbone networks adopted in our simulation . . . . .	25
4.2	Number of trees in Abilene network . . . . .	26
4.3	Number of trees in NSFNET network . . . . .	27
4.4	Number of trees in ARPANET network . . . . .	27
4.5	Number of trees in vBNS network . . . . .	27
4.6	Number of trees in Random network . . . . .	28

4.7	Number of forwarding states in Abilene network . . . . .	29
4.8	Number of forwarding states in NSFNET network . . . . .	29
4.9	Number of forwarding states in ARPANET network . . . . .	29
4.10	Number of forwarding states in vBNS network . . . . .	30
4.11	Number of forwarding states in Random network . . . . .	30
4.12	Delay in Abilene network . . . . .	31
4.13	Delay in NSFNET network . . . . .	32
4.14	Delay in ARPANET network . . . . .	32
4.15	Delay in vBNS network . . . . .	33
4.16	Delay in Random network . . . . .	33
4.17	Throughput in Abilene network . . . . .	34
4.18	Throughput in NSFNET network . . . . .	34
4.19	Throughput in ARPANET network . . . . .	34
4.20	Throughput in vBNS network . . . . .	35
4.21	Throughput in Random network . . . . .	35
4.22	Number of trees for proposed search range . . . . .	40
4.23	Number of forwarding states for proposed search range . . . . .	40
4.24	Delay for proposed search range . . . . .	41
4.25	Throughput for proposed search range . . . . .	41
4.26	Number of trees for proposed overlapping degree . . . . .	43
4.27	Number of forwarding states for proposed overlapping degree . . . . .	43
4.28	Delay for proposed overlapping degree . . . . .	44
4.29	Throughput for proposed overlapping degree . . . . .	44
4.30	Number of trees for proposed replacement method . . . . .	46
4.31	Number of forwarding states for proposed replacement method . . . . .	46
4.32	Delay for proposed replacement method . . . . .	47
4.33	Throughput for proposed replacement method . . . . .	47
4.34	Effect of variation in simulation time on the number of trees . . . . .	49
4.35	Effect of variation in simulation time on the number of forwarding states . . . . .	49
4.36	Effect of variation in simulation time on delay . . . . .	50

4.37	Effect of variation in simulation time on throughput . . . . .	50
4.38	Effect of variation in data rate on the number of trees . . . . .	51
4.39	Effect of variation in data rate on the number of forwarding states . . . . .	52
4.40	Effect of variation in data rate on delay . . . . .	52
4.41	Effect of variation in data rate on throughput . . . . .	53
4.42	Effect of variation in bandwidth on the number of trees . . . . .	54
4.43	Effect of variation in bandwidth on the number of forwarding states . . . . .	54
4.44	Effect of variation in bandwidth on delay . . . . .	55
4.45	Effect of variation in bandwidth on throughput . . . . .	55
4.46	Effect of variation in coincidence degree on the number of trees . . . . .	56
4.47	Effect of variation in coincidence degree on the number of forwarding states . . . . .	57
4.48	Effect of variation in coincidence degree on delay . . . . .	57
4.49	Effect of variation in coincidence degree on throughput . . . . .	58
4.50	Effect of variation in overlapping degree on the number of trees . . . . .	58
4.51	Effect of variation in overlapping degree on the number of forwarding states . . . . .	59
4.52	Effect of variation in overlapping degree on delay . . . . .	59
4.53	Effect of variation in overlapping degree on throughput . . . . .	60
4.54	Effect of different iterations on the number of trees . . . . .	60
4.55	Effect of different iterations on the number of forwarding states . . . . .	61
4.56	Effect of different iterations on delay . . . . .	61
4.57	Effect of different iterations on throughput . . . . .	61
4.58	Effect of variation in relative numbers of large and small groups on the number of trees . . . . .	63
4.59	Effect of variation in relative numbers of large and small groups on the number of forwarding states . . . . .	63
4.60	Effect of variation in relative numbers of large and small groups on delay . . . . .	64
4.61	Effect of variation in relative numbers of large and small groups on throughput . . . . .	64

# List of Tables

4.1	Simulation parameters . . . . .	25
4.2	Performance comparison over conventional method in Abilene network . . . . .	36
4.3	Performance comparison over conventional method in NSFNET network . . . . .	36
4.4	Performance comparison over conventional method in ARPANET network . . . . .	36
4.5	Performance comparison over conventional method in vBNS network . . . . .	37
4.6	Performance comparison over conventional method in Random network . . . . .	37
4.7	Summary of performance comparison over conventional method . . . . .	37
4.8	Performance comparison over STS [9] method in Abilene network . . . . .	37
4.9	Performance comparison over STS [9] method in NSFNET network . . . . .	38
4.10	Performance comparison over STS [9] method in ARPANET network . . . . .	38
4.11	Performance comparison over STS [9] method in vBNS network . . . . .	38
4.12	Performance comparison over STS [9] method in Random network . . . . .	38
4.13	Summary of performance comparison over STS method . . . . .	39
4.14	Percentages of improvement using only our proposed refined search range over conventional method . . . . .	41
4.15	Percentages of improvement using only our proposed refined search range over STS method . . . . .	42
4.16	Percentages of improvement using only our proposed refined overlapping degree over conventional method . . . . .	44
4.17	Percentages of improvement using only our proposed refined overlapping degree over STS method . . . . .	45
4.18	Percentages of improvement using only our proposed replacement method over conventional method . . . . .	47

4.19 Percentages of improvement using only our proposed replacement method over STS method . . . . . 48

4.20 Performance improvement summary of our different proposed methods over conventional method . . . . . 48

4.21 Performance improvement summary of our different proposed methods over STS method . . . . . 48

4.22 Summary of average percentage of improvement . . . . . 65

# Chapter 1

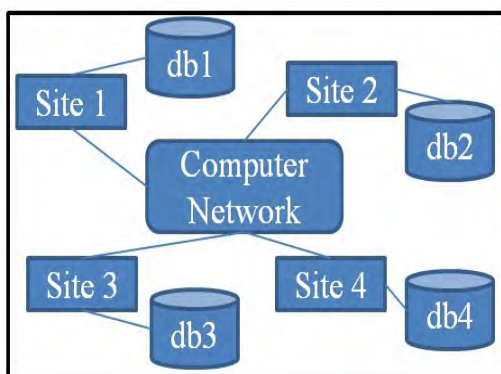
## Introduction

Multicast is an efficient mechanism for delivery of data from one or multiple sources to a set of destinations (receivers) identified by a multicast group of channel [1]. Multicasting is a useful operation for supporting several evolving applications such as WWW (World Wide Web), video/audio on-demand services, distributed database systems, information dissemination, distance learning, teleconferencing, etc. Fig. 1.1 shows some applications of multicasting covering distributed database system, teleconferencing, information dissemination, and distance learning using multicast.

Multicast routing protocols make logical multicast trees from source(s) to destination(s). To deliver packets from source(s) to destination(s), multicast trees can get constructed based on different mechanisms. For example, in case of Source Specific Multicast Tree for a single source, a multicast tree gets generated having its root at its source and having all destinations as leaves [2]. Besides, in case of Shared Multicast Tree for multiple sources, a tree gets generated having its root at a Rendezvous Point and having all destinations as its leaves [3]. For a shared multicast tree, packets are first sent to the rendezvous point from source(s) and then get delivered to destinations following the shared multicast tree.

In conventional multicast, every router on a multicast tree contains forwarding states in the Network layer for routing, and each forwarding state contains child routers on the multicast tree. Such conventional multicast often suffers from the scalability problem, as the number

of forwarding states maintained in a router increases linearly with the number of multicast groups passing through it. Various approaches have been proposed to solve this multicast scalability problem. Examples include aggregation based schemes [4], MPLS (Multiprotocol Label Switching)[5], tunnel based schemes [6], etc. Aggregated multicast attempts to reduce the number of trees by forcing multiple groups to share the same tree within a domain, where the groups are said to be aggregated to this tree [4]. However, the scalability problem still remains. Besides, existing aggregated approaches mostly analyze only tree aggregation ratio while performing the aggregation, retaining the effect of tree aggregation ratio in terms of network performance metrics such as delay and throughput yet to be focused.



(a) Distributed database system



(b) Teleconferencing



(c) Information dissemination



(d) Distance learning

Figure 1.1: Applications of multicasting



## 1.1 Existing Research Studies and Their Limitations

Conventional multicast suffers from scalability problem. Here, the number of forwarding states maintained in the Network layer generally increases with an increase in the number of concurrently active multicast groups. Consequently, large-size routing tables get generated causing degradation in router performance. Several methods such as PIM-SM [3], AM [4], MPLS [5], MMT [6], CBT [7], STA [8], STS [9], etc., have been proposed in the literature to reduce the number of forwarding states. PIM-SM method builds unidirectional shared trees rooted at a rendezvous point (RP) per group and optionally creates a shortest path tree per source. CBT protocol is a multicast routing architecture that builds a single delivery tree per group and the tree has its root at the core router. MMT approach is proposed for scalable multicast in MPLS networks in which packets are transported from one router to the next by MPLS label switching. STA reduces the number of trees by allowing several groups to be aggregated to the same tree. STS proposes to select a smaller size Shared Tree for a multicast session. In addition to such generalized approaches, few specialized multicast approaches have recently been proposed for specific types of networks such as content-centric networks [10], software defined networks [11], etc. However, till now, there exist only a limited number of studies specifically focusing on aggregated multicast to the best of our knowledge. The existing aggregated approaches namely AM [4], STA [8], and STS [9] attempt to perform tree aggregation in different methods, however, still retain high number of forwarding states. Besides, they mostly ignore the effect of tree aggregation ratio in terms of network performance metrics such as delay and throughput.

## 1.2 Our Proposed Approach

In this work, our goal is to present a solution for the scalability problem by reducing the number of forwarding states. To do so, we propose a new approach for selecting aggregated multicast trees to reduce the number of forwarding states. Here, we present three different heuristics. In our proposed approach, first, we refine search range, which is used over the existing tree list while attempting aggregation with a new multicast session. Next, we redefine the notion of overlapping degree to improve the criteria of selection of an existing tree for aggregation.

Finally, we introduce a new method called replacement method for dropping stale aggregated tree. The proposed replacement method helps us to keep the size of routing table limited. We analyze of performance of enabling all the heuristics using ns-3 simulation in terms of different performance metrics.

### 1.3 Performance Metrics for Multicast

There exist several performance metrics such as the number of trees, the number of forwarding states, delay, and throughput for tree aggregation algorithms in the literature. In traditional IP multicast, the number of trees is equal to the number of concurrent groups (each group is assigned its own tree). On the other hand, the notion of tree aggregation attempts to reduce the number of trees. Here, routing protocols maintain each tree by periodically sending control messages. Therefore, the less number of trees in a domain we have, the less control overhead will be incurred [8]. Besides, the number of forwarding entries depends on the number of trees. A large number of forwarding entries in a router slows down IP lookup, which degrades performance [8]. Accordingly, the number of trees and the number of forwarding states serve as important performance metrics. Besides, delay and throughput are other noteworthy network performance metrics for tree aggregation.

### 1.4 Our Contributions

Based on our work on aggregated multicast to reduce forwarding states considering the above-mentioned performance metrics, we make the following set of contributions in this study.

- We propose a new approach to solve multicast scalability problem considering different relevant performance metrics (the number of trees, the number of forwarding states, end-to-end delay, and throughput). The objective of our approach includes reduction of end-to-end delay by reducing the number of forwarding states, which is often ignored in contemporary studies.
- In our approach, we propose a new searching criteria for picking the existing candidate trees with which aggregation of a new multicast group is attempted. We also introduce

a notion of overlapping ratio for better aggregation and a new replacement method to drop a stale tree when the maximum number of stored aggregation trees goes beyond the storing capacity after inclusion of a newly-formed tree.

- We use network simulator ns-3 to experimentally evaluate performances of our proposed approach and other existing approaches in diversified settings. We perform simulation on different benchmark network topologies such as Abilene, NSFNET, ARPANET, and vBNS along with a random topology. Comparative analysis over all the experimental results demonstrate that our proposed approach can provide significant performance improvement in most of the cases compared to the existing approaches.

## 1.5 Outline of Our Thesis

The rest of this study is organized in the following way: In Chapter 2, we will present background of our research and related research studies. After that, in Chapter 3, we propose the methodology that we use to solve the scalability problem. In Chapter 4, we elaborate our simulation and its results. Finally, we conclude the study pointing its future work in Chapter 5.

# Chapter 2

## Related Work

Different solutions have been proposed to solve the scalability problem of multicast by reducing the number of forwarding states. Example solutions include PIM-SM (Protocol Independent Multicast Sparse Mode), CBT (Core Based Tree), MMT (MPLS Multicast Tree), AM (Aggregated Multicast), STA (Scalable Tree Aggregation for Multicast), STS (Shared-Tree Selection Method for Aggregated Multicast), etc. Besides, few specialized multicast approaches have recently been proposed for specific types of networks such as content-centric networks [10], software defined networks [11], etc. In this chapter, we elaborate the existing studies along with pointing their limitations.

### **2.1 Protocol Independent Multicast - Sparse Mode (PIM-SM)**

PIM-SM (Protocol Independent Multicast - Sparse Mode) is a protocol for efficiently routing Internet Protocol (IP) packets to multicast groups. The protocol is named as protocol independent because it is not dependent on any particular routing protocol for topology discovery, and as sparse mode because it is suitable for groups where a very low percentage of the nodes (and their routers) will subscribe to the multicast session. PIM-SM explicitly builds unidirectional shared trees rooted at a rendezvous point (RP) per group and optionally creates a shortest path tree per source.

There are several methods to select the rendezvous point. In the study presented in [12], authors propose a multicast RP selection algorithm based on Tabu search. The authors in [13] further present multicast protection trees that provide instantaneous failure recovery from single node failures. In PIM-SM method, routers explicitly join and leave the multicast group using PIM protocol messages. PIM protocol messages can be of two different types-Join and Prune messages [14].

Several studies have analyzed different aspects of PIM-SM. For example, the study presented in [15] analyzes fault tolerance of the PIM-SM IP multicast routing protocol. Another study in [16] analyzes the performance of video conferencing in unicast and multicast communication using PIM-SM.

Fig. 2.1 demonstrates the PIM-SM method. In this figure, R2 serves as a rendezvous point. During a data transmission, the source sends data to the rendezvous point (R2) and then get the data delivered to receiver(s). The rendezvous point maintains multicast Join and Prune messages. After receiving a join message from a router, the rendezvous point informs other routers about it. Similarly, after receiving a prune message from a router, the rendezvous point informs other routers about it. Besides, the rendezvous point can be used for access control scheme in PIM-SM multicast [17].

The main limitation of PIM-SM is that it maintains individual multicast tree for each individual multicast group. Individual multicast tree for each multicast group results in more forwarding states with an increase in the number of multicast groups. Thus, the problem of scalability remains. Besides, PIM-SM always forces sending data packets to RP rather than sending to receivers, which is often not optimal.

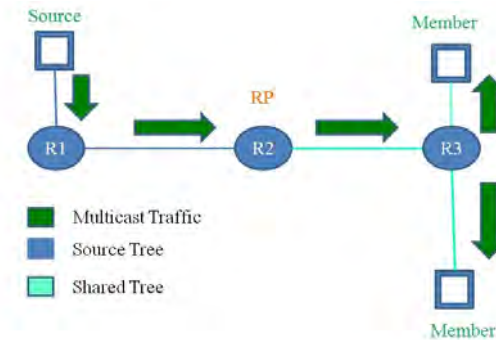


Figure 2.1: PIM-SM multicast

## 2.2 Core-Based Tree (CBT)

Core-Based Tree (CBT) proposed by Ballardie et al, is a mechanism for making IP multicast scalable by constructing a tree of routers. Here, multiple cores can be placed in a CBT [7]. CBT can be rooted at a core router [18]. Locations of the core routers are statically configured. The study in [19] proposes a heuristic algorithm for selecting the core routers. In [20], authors describe core placement mechanism for the core based tree. In [21], authors propose adaptive core multicast routing protocol. Load balancing and anomaly detection techniques for multi-core based tree are described in [22] [23]. The study presented in [24] designs and evaluates a reliable multicast scheme for core-based multicast trees. Another study [25] presents a technique for compression of core-based trees. Fault recovery and fault-tolerant mechanisms for core-based trees are presented in [26] [27].

All these studies are based on the classical CBT. Fig. 2.2 demonstrates an overview of the classical CBT. Here, during a data transmission, source node sends data to the core router and then get the sent data delivered to receiver(s). For the purpose of data delivering, the core also maintains multicast Join and Prune messages.

CBT maintains individual multicast tree for each individual multicast group. Individual multicast tree for each multicast group results in more forwarding states with an increase in the number of multicast groups. CBT does not solve the problem of how to map a group

address to a single core. In addition, a good core placement is a difficult problem. Without good core placement, CBT trees can be quite inefficient. Therefore, CBT is unlikely to be used as a global multicast routing protocol.

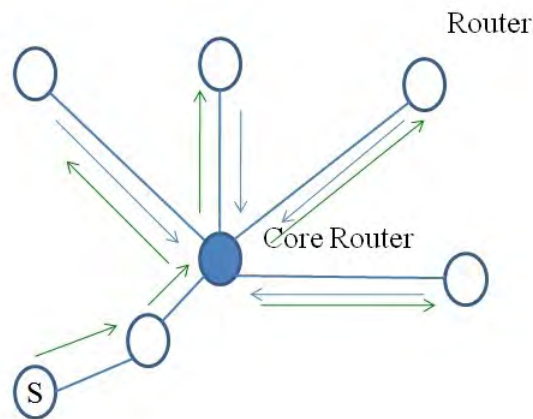


Figure 2.2: Core-based tree [7]

## 2.3 MPLS Multicast Tree (MMT)

The key idea of MMT approach is that, instead of having multicast forwarding states in *all* routers under a tree for each individual multicast session in the core network (backbone), one can have multicast forwarding states only in branching routers (BRs) of the tree [6]. MPLS Multicast Tree (MMT) solution is proposed for scalable multicast in MPLS networks in which packets are transported from one BR to the next-hop BR by MPLS label switching [28]. MPLS uses a signaling protocol such as resource reservation protocol [29] or label distribution protocol [30] to set up label switching protocol (LSP). Here, multiple copies of multicast packets get generated only in the BRs. Possibility of tunneling between BRs has been studied in [31-32]. Besides, MPLS has been revisited in several research studies [33, 34, 35, 36, 37]. In these studies, different aspects are addressed together with MPLS including traffic engineering [34], forwarding state control [36], scalable QoS multicast provisioning [44], etc.

Fig. 2.3 presents an overview of MPLS Multicast Tree (MMT). In this figure, S router is the

source router, and R1 and R2 routers are the receivers. Network Information Manager System (NIMS) keeps all necessary information on LSP. The NIMS gets informed directly of any change in topology of the network (LSP or routers failure) and of any change of membership of a group destination [6]. The NIMS informs all BRs about this change [6]. MMT maintains individual multicast tree for each individual multicast group. Individual multicast tree for each multicast group results in more forwarding states with an increase in the number of multicast groups. Thus, MMT reduces only non-branching forwarding states leaving the problem of having individual multicast tree for each multicast group as it remains in the earlier mentioned existing studies. In road to solve this problem, the notion of aggregation came into light.

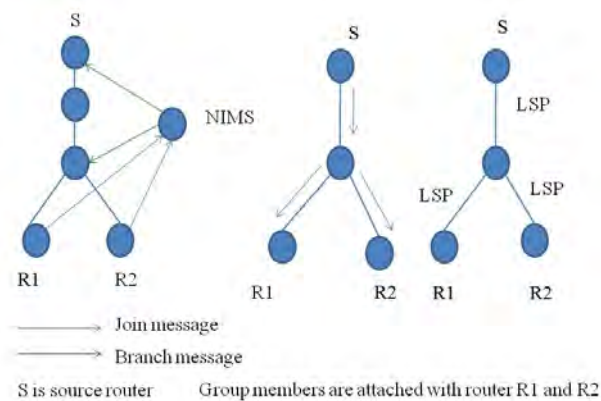


Figure 2.3: MPLS Multicast Tree [2]

## 2.4 Aggregated Multicast (AM)

The existing studies on Aggregated Multicast deal with aggregation of multicast groups. The key idea of aggregated multicast is that, instead of constructing a tree for each individual multicast session in the core network (backbone), one can have multiple multicast sessions sharing a single aggregated tree to reduce multicast states and correspondingly to limit tree maintenance overhead at the network core [4]. AM algorithm evaluates all the trees in the current tree set  $T$  every time while attempting an aggregation of newly arrived multicast group. Several studies, [39-46] have revised aggregated multicast where different aspects are addressed together with AM including control message aggregation [40], aggregated multicast



based on tree splitting [40], distributed multicast based tree aggregation [41], QoS scalable tree aggregation [44], etc. The study in [38] proposes aggregation of control packets for multicast and other hierarchical network protocols to reduce delay. Another study in [39] proposes a scheme, called AMBTS, which uses the concept of aggregated multicast and employs tree splitting before aggregating. QoS Scalable Tree Aggregation (Q-STA) [44] proposes to allow several groups to share the same tree. Q-STA accepts much more groups and performs faster aggregations than previous algorithms.

Fig 2.4 shows an overview of aggregated multicast (AM). In the figure 2.4, Session 1 has destination nodes 1, 2, 3, and 4, Session 2 has destination nodes 2, 3, and 4, and Session 3 has destination nodes 2, 3, 4, and 5. Based on AM, Session 1, 2, and 3 can share a single tree and the shared tree has destination nodes 1, 2, 3, 4, and 5. In such a case, certain amount of bandwidth gets wasted to deliver data to nodes that are not involved for a certain group. Nonetheless, existing aggregated multicast approaches still generates more forwarding states with an increase in the number of multicast groups. Moreover, existing studies often ignore analyzing network-level performance metrics such as delay and throughput.

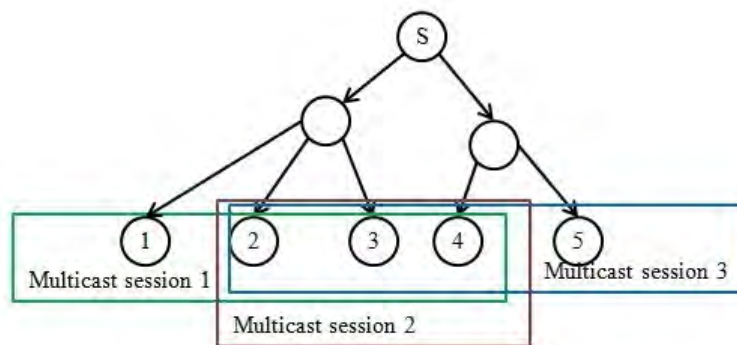


Figure 2.4: Aggregated multicast tree

## 2.5 Scalable Tree Aggregation for Multicast

The study in [8] proposes an algorithm called STA (Scalable Tree Aggregation for Multicast), which reduces the number of trees by allowing several groups to be aggregated under the same tree. The main idea of STA is to partition the trees in a multicast tree set  $T$  considering their cost, which refers to sum of valuations of the edges of a tree. STA reduces the number of evaluated trees for an aggregation of newly arrived multicast group, where as the previous algorithms evaluate all the trees in the current tree set  $T$  every time [8]. Thus, STA selects a bigger Shared Tree that may contain some unnecessary nodes in addition to all destination nodes in the new multicast group [9]. In this case, packets are sent to non-member nodes for the multicast group, which causes unnecessary increase in network traffic [9]. Outcome of STA is similar to that shown and explained in Fig. 2.4. Here, the existing tree to be adopted for aggregation is selected based on tree cost.

## 2.6 Shared-Tree Selection Method for Aggregated Multicast

The study presented in [9] demonstrates Shared-Tree Selection (STS) Method for Aggregated Multicast. In this method, it is possible to select a smaller size Shared-Tree compared to the multicast group [9]. STS proposes a notion overlapping degree to select a smaller size Shared-Tree. Outcome of STS is similar to that presented and explained in Fig. 2.4. Here, similar to the earlier cases, STS generates more forwarding states with an increase in the multicast groups. Besides, STS ignores analyzing network-level performance metrics such as delay and throughput.

## 2.7 Limitations of the Existing Studies

Initial approaches such as PIM-SM and CBT maintain individual multicast trees for each multicast group. Such approaches result in more forwarding state, with an increase in the number of multicast groups. As a remedy, aggregate multicast approaches propose that multiple multicast groups can share a single aggregated tree to reduce tree maintenance overhead at the network core. However, the existing aggregated approaches still retain a high number of forwarding states. Besides, they mostly ignore the effect of tree aggregation ratio in terms of network performance metrics (for example delay and throughput). In this study, we attempt to focus on these issues.

# Chapter 3

## Proposed Mechanism for Selecting Aggregated Multicast Trees

In order to overcome limitations of existing aggregated multicast approaches, we propose a novel aggregated multicast approach. In this section, we present our proposed aggregated multicast approach. In our approach, we refine search range over the existing tree list while attempting aggregation with a new tree and redefine the notion of overlapping degree to improve the criteria of selecting an existing tree for aggregation. We also propose a new method for dropping stale aggregated tree. We elaborate each of these proposals next, after providing an overview on the whole aggregated multicast process.

### 3.1 Overview on Proposed Aggregated Multicast

As mentioned earlier, existing AM approaches generate more forwarding states with an increase in the number of multicast groups. Therefore, our goal is to solve this scalability problem by reducing the number of forwarding states. To do so, first, our proposed mechanism constructs a tree for newly arrived multicast group. Then, we search for a possible aggregation based on search range and overlapping degree as defined in Equation 3.1 and 3.2 respectively. If the newly arrived multicast session tree matches with any tree from the current tree set, then the newly arrived multicast session tree gets aggregated with the existing tree. If there is no tree in the current tree set that matches with the newly arrived multicast session, then a new tree

for the newly arrived multicast session is constructed and registered in the current tree set. Next, replacement method may be required during new tree registration if the size of all trees exceeds maximum storage capacity while attempting a new aggregation and registration. In our approach, we replace the *least-recently aggregated* tree with the new tree during the registration process. We elaborate whole aggregation multicast processes of the first case (the newly arrived multicast session tree matches with any tree from the current tree set). The second case (there is no tree in the current tree set that matches with the newly arrived multicast session) later in this section.

## 3.2 Refining Search Range

In our proposed approach, we refine search range used over the existing tree list while attempting aggregation with a new multicast session. In our refined approach, searching occurs only on non-smaller trees in the current tree set compared to the tree representing new multicast session for aggregating the new multicast session. If any non-smaller tree of the current tree set matches with a newly arrived multicast session tree, then the new tree gets aggregated with the matched existing tree. Otherwise, the newly arrived multicast session tree gets registered as a new tree in the current tree set. Note that the existing approaches consider both smaller and non-smaller trees of the current tree set for the purpose of aggregation. However, in case of smaller trees, all routers need to update routing information again. Therefore, in our case, we consider only non-smaller trees.

Equation 3.1 defines our proposed search range as follows:

$$|N_{GT}| \leq |N_{ST}| \leq \frac{|N_{GT}|}{P} \quad (3.1)$$

Here,  $|N_{GT}|$  is the number of nodes in destination node set of new multicast group tree,  $|N_{ST}|$  is the number of nodes in destination node set of the tree under construction from existing tree set, and  $P$  is coincidence degree.

After arriving a new multicast session, searching occurs for a tree from the current tree set for aggregation. If a tree of the current tree set matches with new multicast session

based on Equation 3.1, then we perform overlapping degree calculation. We present detail of overlapping degree calculation later. Otherwise, a new tree gets constructed and the newly constructed multicast tree gets registered in the current tree set. Next, we present an example of utilizing our refined search range.

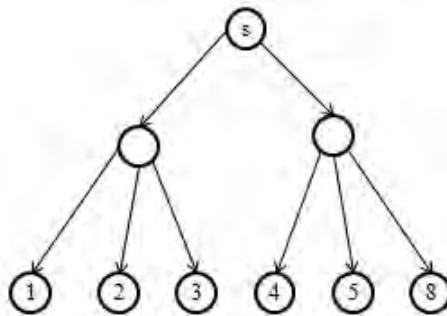


Figure 3.1: Multicast tree for newly arrived multicast group  $G_0$

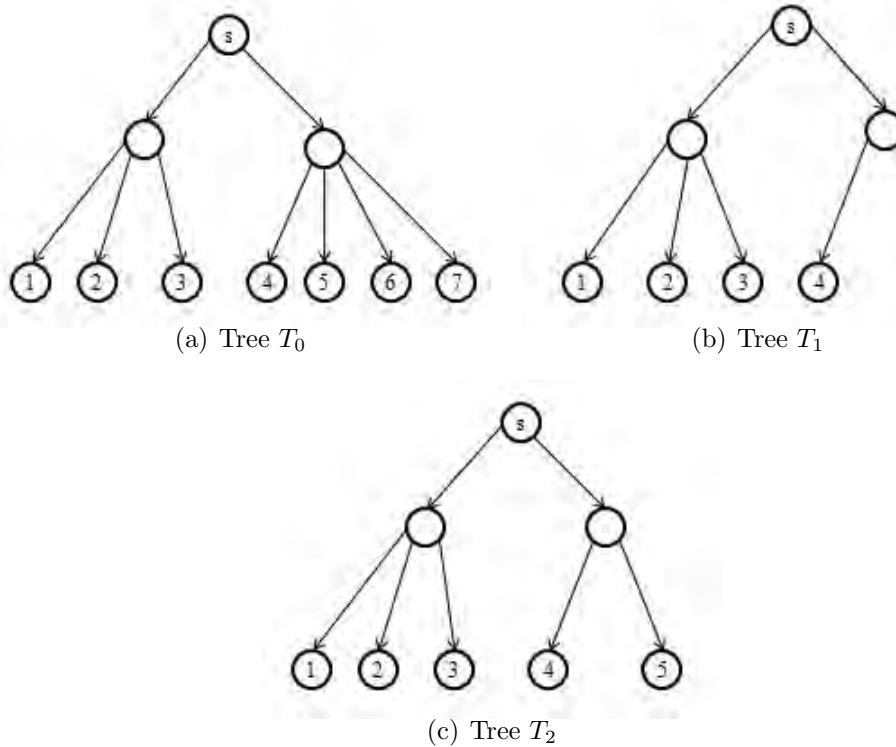


Figure 3.2: Trees in existing tree set  $T$

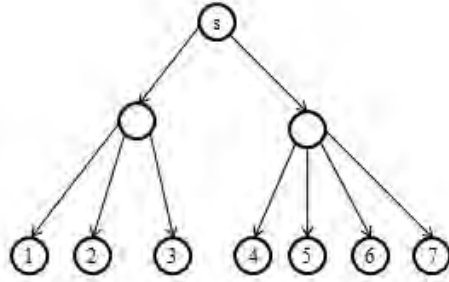


Figure 3.3: Tree  $T_0$  is selected for newly arrived multicast group  $G_0$

Let a new multicast group  $G_0(s, 1, 2, 3, 4, 5, 8)$  has destination nodes 1, 2, 3, 4, 5, and 8 as shown in Fig. 3.1. Besides, Fig. 3.2 shows trees in the existing tree set  $T$  under consideration for aggregation. We present outcome of our proposed refined searching technique in Fig. 3.3. Here, searching occurs only for non-smaller trees of the existing tree set. In the existing tree set, only non-smaller tree  $T_0$  has destination nodes 1, 2, 3, 4, 5, 6, and 7. Therefore, we perform our aggregation with the tree. Note that, here, we consider the coincidence degree,  $P$  as 0.8. Tree  $T_0$  of the current tree set gets considered for the new multicast group  $G_0$  for its aggregation as  $6 \leq |N_{T_0}| \leq \frac{6}{0.8}$ .

### 3.3 New Consideration over Overlapping Degree

To improve the criteria of selection of an existing tree for aggregation ratio, we propose a notion of overlapping degree. To calculate overlapping degree, we consider overlapping of destination nodes of newly arrived multicast session tree with the destination nodes of the trees from existing tree set under consideration for aggregation. We use Equation 3.2 for the purpose of calculating overlapping degree. In our study, we set a threshold on the overlapping degree as the 0.85. If the overlapping degree of a tree in the search range becomes equal or greater than the threshold, then the newly arrived multicast session tree gets aggregated with the tree from the current tree set. Otherwise, we construct a new multicast tree and register it in the current tree set. Equation realizing the notion of overlapping degree is as follows:

$$\gamma = \frac{|N_{ST} \cap N_{GT}|}{|N_{ST}|} \quad (3.2)$$

Where,  $N_{GT}$  is the set consisting destination nodes of the newly arrived multicast session tree and  $N_{ST}$  is the set consisting destination nodes of the tree from the current tree set under consideration for aggregation. Next, we present an example of utilizing the notion of overlapping degree.

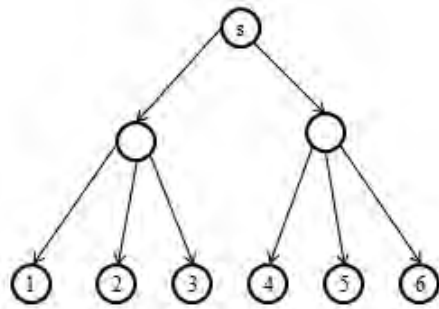


Figure 3.4: Multicast tree for newly arrived multicast group  $G_0$

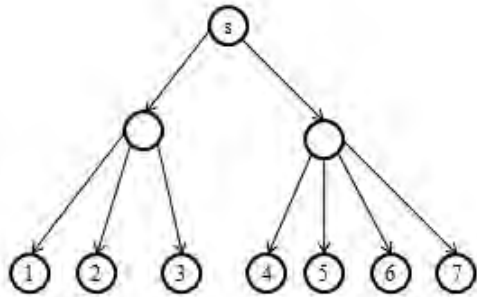


Figure 3.5:  $T_0$  is a tree in the current tree set  $T$

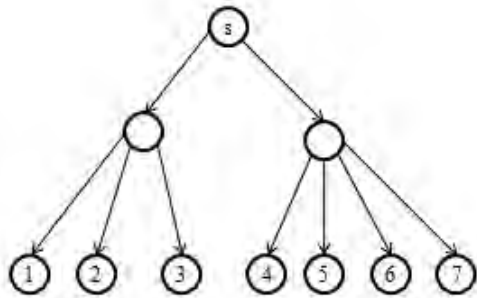


Figure 3.6: Tree aggregation as per the proposed overlapping mechanism, as destination nodes of newly arrived multicast group  $G_0$  overlap with that of tree  $T_0$



Let a new multicast group  $G_1(s, 1, 2, 3, 4, 5, 6)$  has destination nodes 1, 2, 3, 4, 5, and 6. Besides, tree  $T_0$  from the current tree set (shown in Fig. 3.5) has destination nodes 1, 2, 3, 4, 5, 6, and 7. Accordingly, the tree  $T_0$  of the current tree set  $T$  can get selected for aggregation as per our notion of overlapping degree. Fig. 3.6 presents outcome of such aggregation where destination nodes of group  $G_1$  substantially overlap ( $\gamma \geq 0.85$ ) with destination nodes of tree  $T_0$  in the current tree set.

### 3.4 Replacement Method for Stale Aggregated Tree

It is infeasible to keep infinite entries of shared-trees in the routing table considering limitation of storage capacity. To solve this issue, in our proposed approach, we introduce a new replacement method. Here, we use *least-recently aggregated* metric rather than the classically used notion of *least-frequently used*. In this replacement method, when the number of trees registered in tree set exceeds maximum capacity for storing the tree set, the replacement method deletes or drops the least-recently aggregated tree. By using the replacement method, the number of trees registered in tree set does not exceed maximum capacity for storing the tree set.

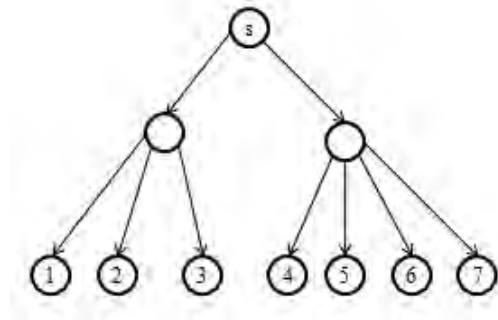


Figure 3.7: Multicast tree for a newly arrived multicast group  $G_2$

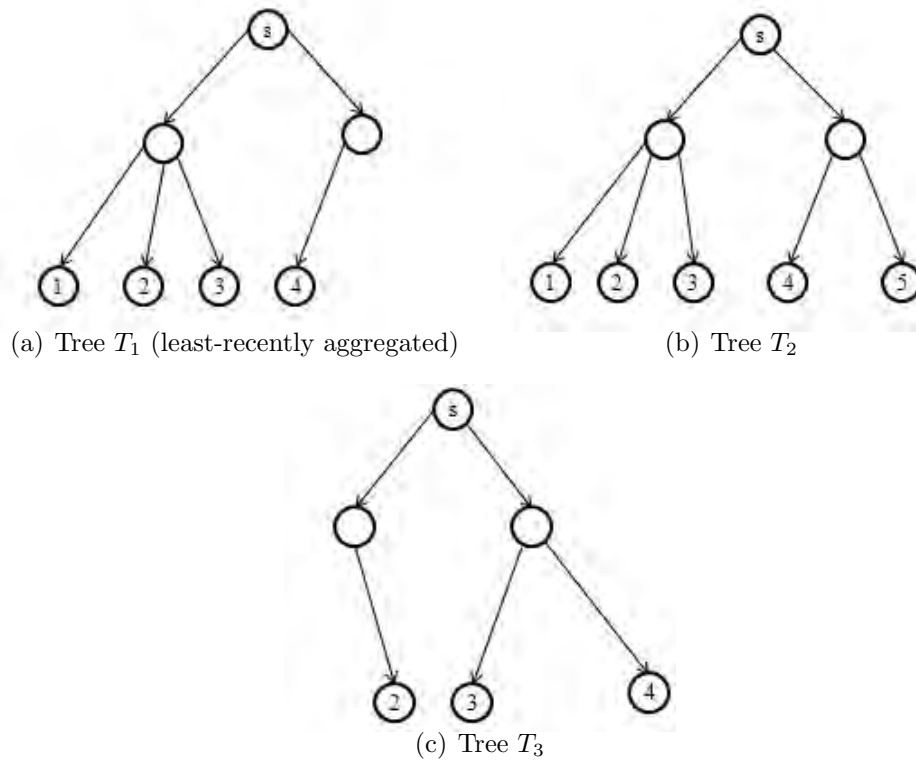


Figure 3.8: Trees in the current tree set  $T$

As an example, let, multicast group  $G_2(s, 1, 2, 3, 4, 5, 6, 7)$  has destination nodes 1, 2, 3, 4, 5, 6, and 7 as shown in Fig. 3.7. Fig. 3.8 shows trees in the current tree set  $T$  along with the least-recently aggregated tree (Fig. 3.8(a)). We present outcome of our proposed replacement method in Fig. 3.9, where the earlier tree of Fig. 3.9(a) gets replaced with the newly arrived multicast session tree  $T_G$  of  $G_2$ .

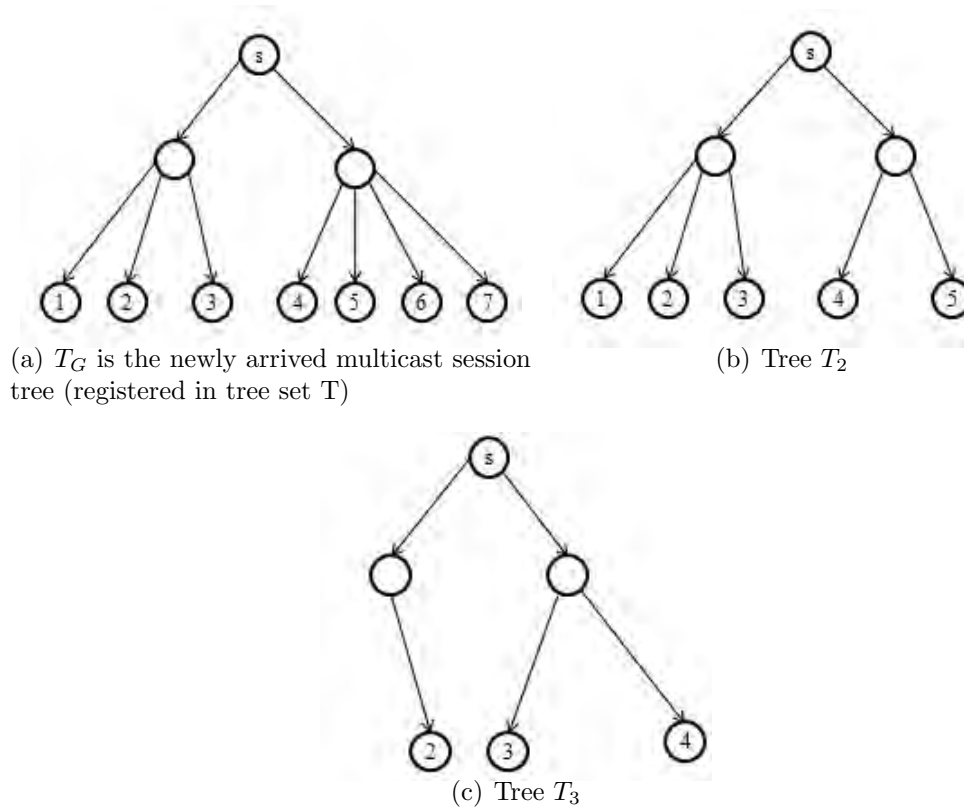


Figure 3.9: Trees of the tree set T after applying proposed replacement method

### 3.5 Overall Proposed Aggregation Approach

In a nutshell, we present our overall aggregation approach below:

(1) When a new multicast session arrives, trees of the existing tree set are searched for possible aggregation based on our refined search range and our introduced notion of overlapping degree as defined in Equation 3.1 and 3.2 respectively. We set overlapping degree as  $\gamma \geq 0.85$  and threshold on coincident degree as  $p = 0.8$  in our study.

(2) If any tree from the current tree set matches with the newly arrived multicast session, then the newly arrived multicast session tree gets aggregated with the existing tree.

(3) If there is no tree in the current tree set that matches as per search range and overlapping degree with the newly arrived multicast session, then a new tree for the newly arrived multicast session is constructed and registered in the current tree set.

(4) Tree set size should not increase infinitely. Therefore, replacement may be required during new tree registration if the size of all trees exceeds maximum allowable storage capacity while attempting a new registration. In our approach, we replace the least-recently aggregated tree with the new tree during the registration process.

We present steps in our proposed aggregation approach in Fig. 3.10 along with its algorithms in Algorithm 1, 2, and 3.

---

**Algorithm 1** Algorithm for Tree Aggregation

---

```

1: procedure TREEAGGREGATION
2:    $T \leftarrow$  tree set
3:    $k \leftarrow$  number of trees in current tree set
4:    $T_G \leftarrow$  newly arrived multicast group tree
5:    $i \leftarrow 0$ 
6:    $found \leftarrow false$ 
7:    $N_{GT} \leftarrow$  destination node sets of newly arrived multicast group tree
8:    $N_{ST} \leftarrow$  destination node sets of the first tree in the current tree set
9:    $P \leftarrow$  coincident degree
10:   $\gamma \leftarrow$  overlapping degree
11:  while  $i < k$  do
12:    if  $N_{GT} \leq N_{ST}$  and  $N_{ST} \leq \frac{N_{GT}}{P}$  then  $\gamma = \frac{|N_{ST} \cap N_{GT}|}{|N_{ST}|}$ 
13:    if  $\gamma \geq 0.85$  then
14:      aggregate  $T_G$  with  $T_i$ 
15:       $found \leftarrow true$ 
16:    if  $i = k-1$  and  $found = false$  then
17:       $treeReplacement()$ 
18:     $i \leftarrow i+1$ 

```

---



---

**Algorithm 2** Algorithm for Tree Replacement

---

```

1: procedure TREEREPLACEMENT
2:    $T \leftarrow$  tree set
3:    $T_G \leftarrow$  newly arrived multicast group tree
4:   if  $T_G$  does not get aggregated in the tree set  $T$  then
5:      $T_{leastAgg} \leftarrow getLeastRecentlyAggregatedTree()$ 
6:      $T_{leastAgg} \leftarrow T_G$ 

```

---

**Algorithm 3** Algorithm for Finding Least Recently Aggregated Tree

```

1: procedure GETLEASTRECENTLYAGGREGATEDTREE
2:    $k \leftarrow$  number of trees in current tree set
3:    $i \leftarrow 0$ 
4:    $T \leftarrow$  tree set
5:    $j \leftarrow$  aggregation frequency of trees in current tree set
6:    $T_G \leftarrow$  newly arrived multicast group tree
7:    $T_{leastAgg} \leftarrow$  least recently aggregated tree
8:    $T_{leastAgg} \leftarrow T[0][j]$ 
9:   if  $T_G$  does not exist in the tree set  $T$  then
10:    while  $i < k$  do
11:      if  $T[i][j] < T_{leastAgg}$  then
12:         $T_{leastAgg} \leftarrow T[i][j]$ 

```

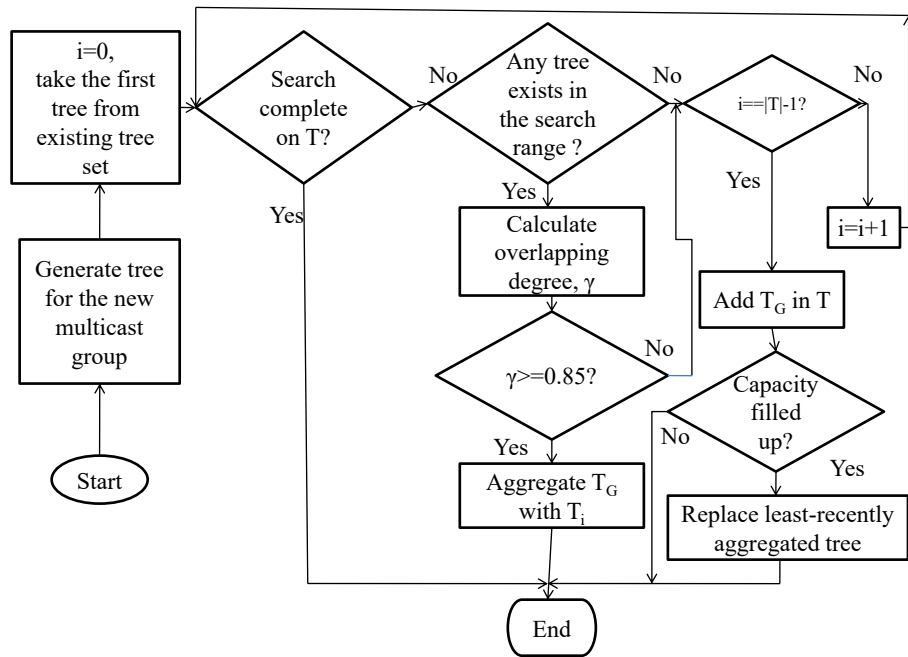


Figure 3.10: Flow chart of our proposed aggregation multicast method

# Chapter 4

## Experimental Evaluation

In this section, we compare performance of our proposed aggregation approach with that of conventional multicast and recently proposed STS [9]. We adapt STS in the comparison, as this serves as the best-known alternative in our case to the best of our knowledge. In our performance comparison, we perform ns-3 simulation to measure various performance metric.

### 4.1 Simulation Settings

We perform simulation on different benchmark network topologies such as Abilene [49], NSFNET [50], ARPANET [51], and vBNS [52] along with a random topology. Fig. 4.1 shows Abilene, NSFNET, ARPANET, vBNS, and a random network topology. In our simulation, to mimic real network setting, we adopt Ethernet, CSMA, OSPF, and UDP as the protocols in Physical layer, MAC layer, Network layer, and transport layer respectively. We vary the number of multicast groups over 100 - 500. We also vary the simulation time over 10 - 100 sec. We set packet size as 1500 bytes and channel delay between two nodes as 100us. We separately vary bandwidth over 100 - 500 Mbps. We separately create point-to-point link setup created between two nodes. The simulation setting is shown in Table 4.1. With these settings of parameters, we execute the simulation. For performance evaluation we compare the number of trees required for multicast, the number of forwarding states generated in the Network layer, delay, and throughput.

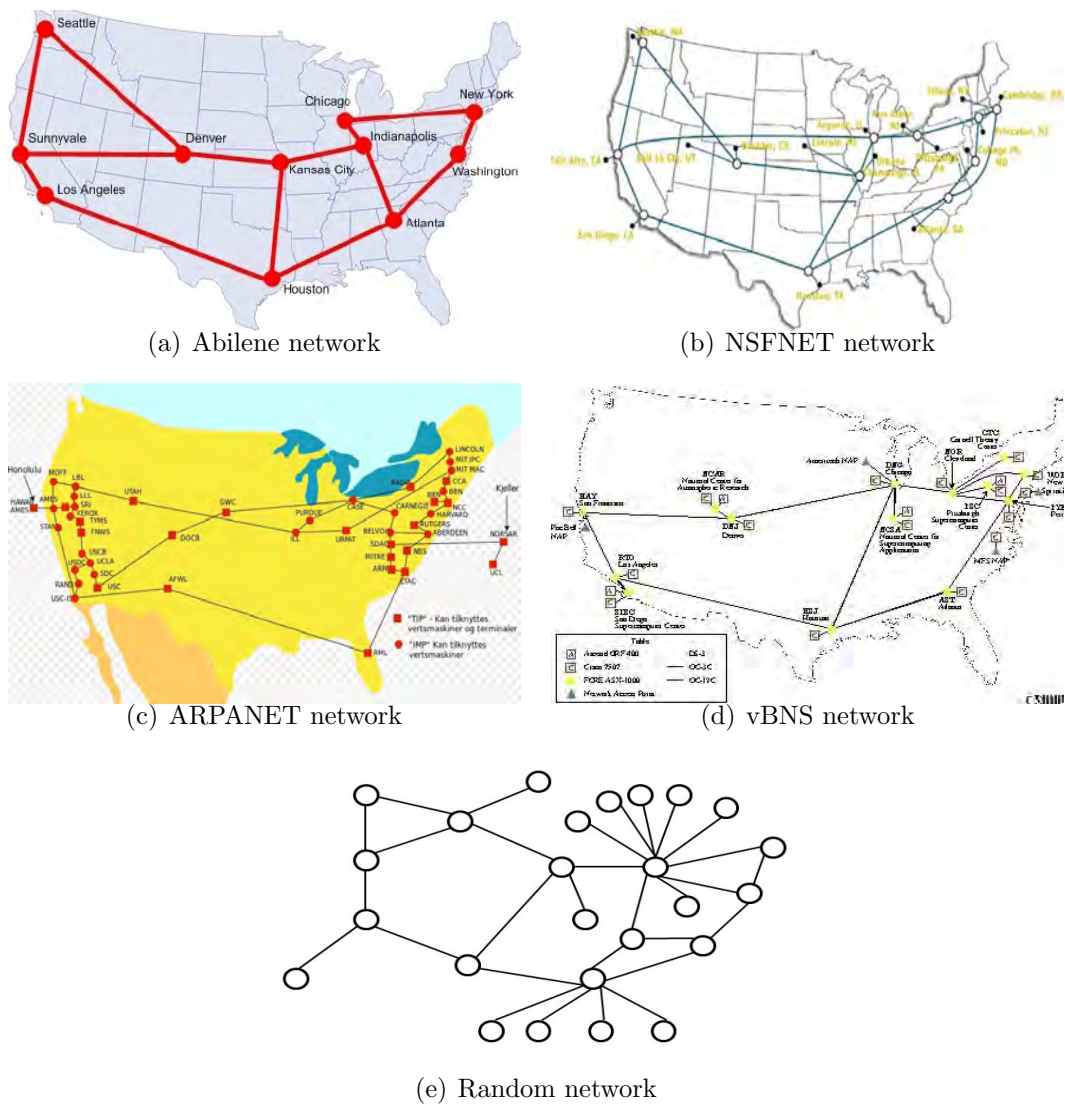


Figure 4.1: Backbone networks adopted in our simulation

Table 4.1: Simulation parameters

Parameter	Value(s)
Packet size (bytes)	1500
Data transmission rate (Mbps)	1
Bandwidth (Mbps)	100 - 500
Number of multicast groups	100 - 500
Physical layer	Ethernet
MAC layer protocol	CSMA
Network layer protocol	OSPF
Transport layer protocol	UDP
Simulation time (seconds)	10 - 100

## 4.2 Performance Comparison with Existing Proposed Protocols

We compare performance of our proposed approach with a recent existing approach proposed in [9] along with the conventional multicast approach. The conventional multicast approach constructs an individual multicast tree for a new multicast session [9]. Besides, the approach presented in [9] constructs aggregation tree to improve scalability. We implement both the multicast approaches along with our proposed one in ns-3 simulator. Our simulation results demonstrate that the proposed approach exhibits significantly better performance over the other two alternative approaches. In the results, we evaluate performance of the approaches in term of different metrics such as the number of trees, the number of forwarding states, delay, and throughput. We elaborate all of these simulation results in detail in the following subsections.

### 4.2.1 Performance in Terms of The Number of Trees

The number of trees measures a baseline efficiency of a tree aggregation approach [10]. The number of trees generally increases with an increase in the number of multicast groups. Fig. 4.2 - 4.6 show the increase in the number of trees against the number of multicast groups for the various network topologies under consideration. The figures show that our proposed approach results in substantially smaller number of trees compared to the other two alternatives.

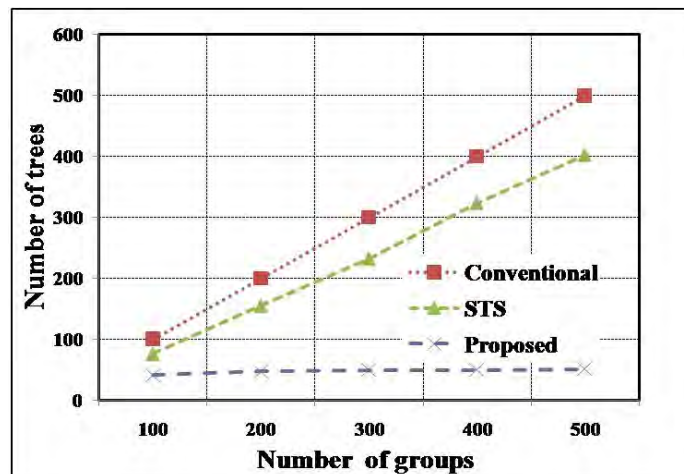


Figure 4.2: Number of trees in Abilene network



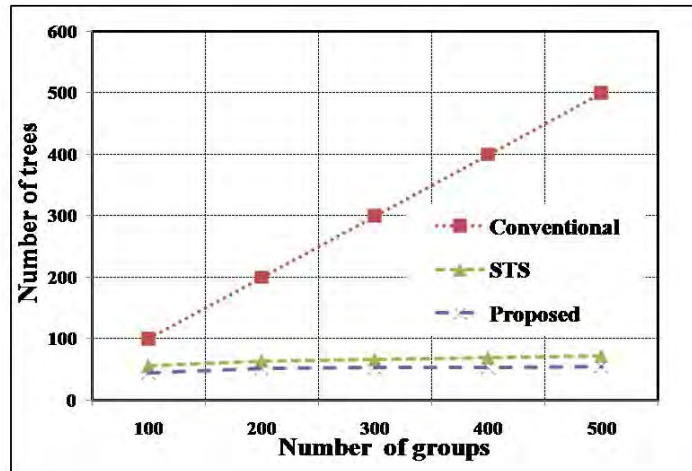


Figure 4.3: Number of trees in NSFNET network

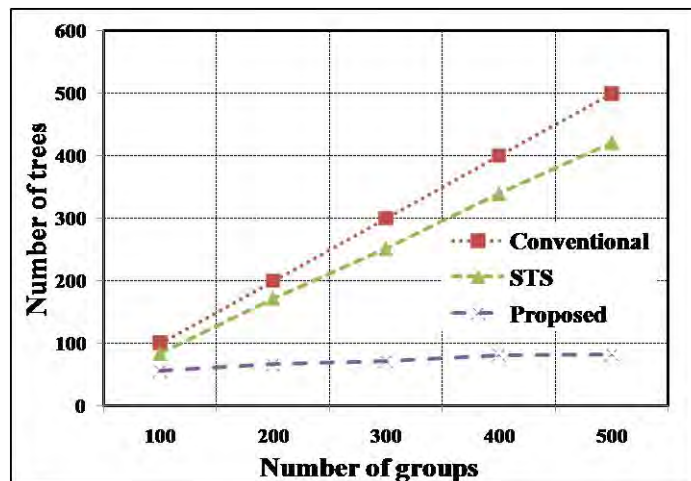


Figure 4.4: Number of trees in ARPANET network

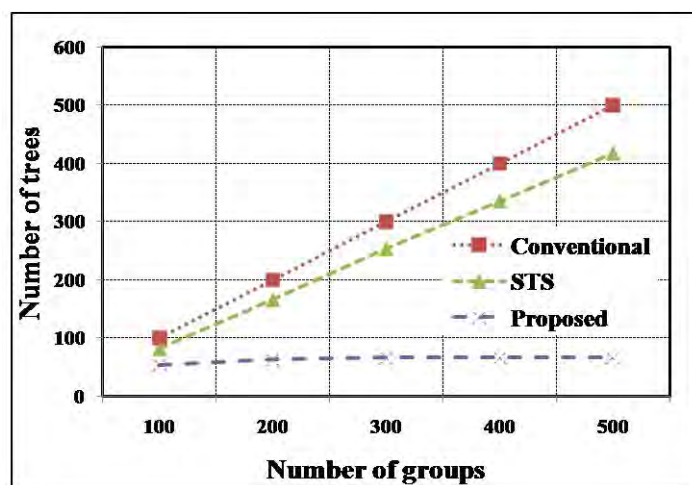


Figure 4.5: Number of trees in vBNS network

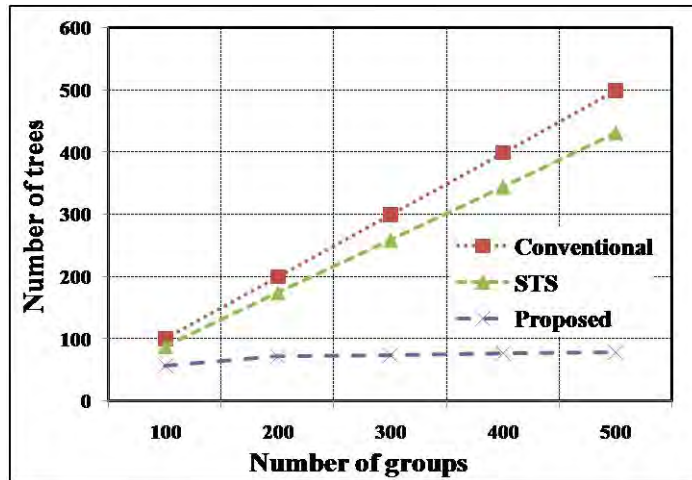


Figure 4.6: Number of trees in Random network

## 4.2.2 Performance in Terms of The Network Layer Forwarding State

The number of forwarding states increases with an increase in the number of multicast groups [47]. We calculate the number of forwarding states using Equation 4.1.

$$\text{Number of forwarding states} = \sum_T n_T \quad (4.1)$$

Where,  $T$  is a tree in the current tree set and  $n_T$  is the number of routers in tree  $T$ .

A forwarding state requires a router to store a routing table entry. When a host has a packet to send or when a router receives a packet that need to be forwarded, the router looks at this table entry to find a route to the final destination [48]. Fig. 4.7 - 4.11 show the increase in the number of forwarding states with an increase in the number of multicast groups for the different network topologies under consideration. The figures show that our proposed approach results in substantially smaller number of forwarding states compared to the other alternatives.

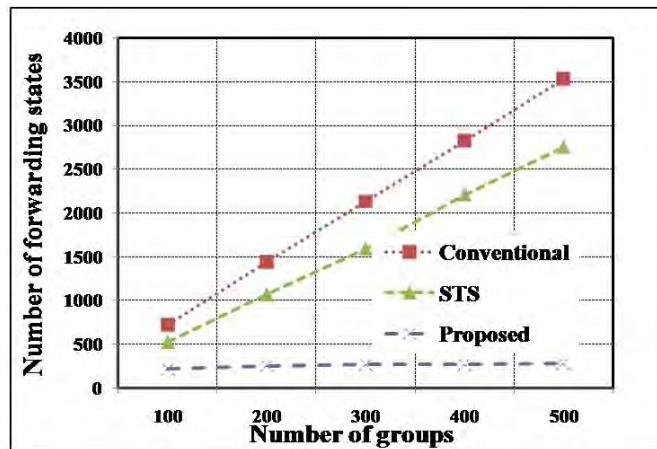


Figure 4.7: Number of forwarding states in Abilene network

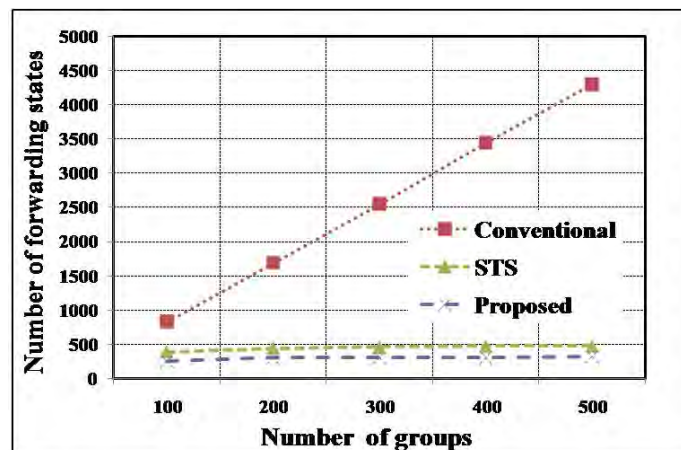


Figure 4.8: Number of forwarding states in NSFNET network

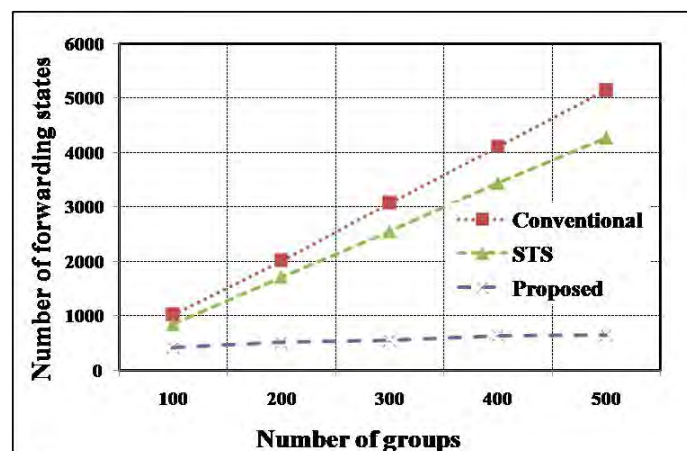


Figure 4.9: Number of forwarding states in ARPANET network

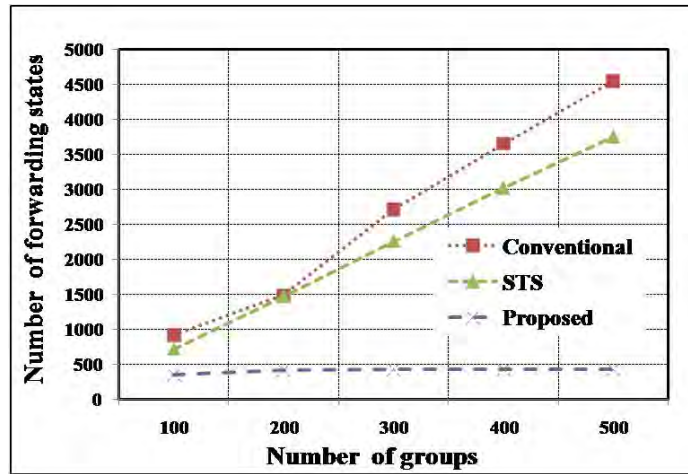


Figure 4.10: Number of forwarding states in vBNS network

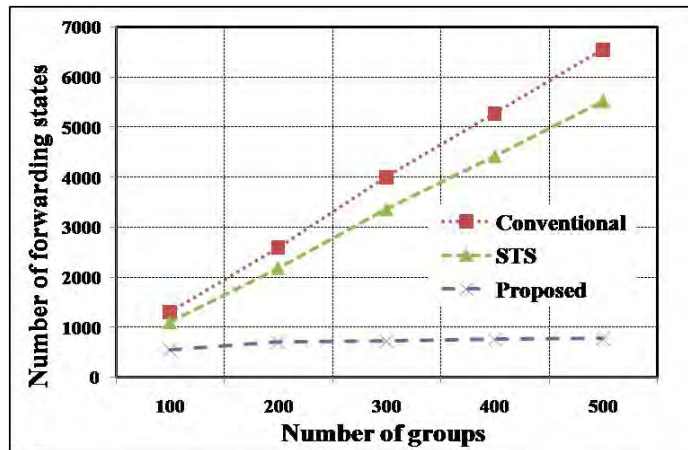


Figure 4.11: Number of forwarding states in Random network

### 4.2.3 Performance in Terms of Delay

Delay is one the most important performance metrics considered while evaluating network operations. For calculating delay, we consider the following three major components:

- Processing time,
- Transmission time, and
- Queuing time.

Processing time ( $p_t$ ): Processing time refers to the time duration required by a router to process the packet headers. Processing time is a key component in delay. During the data

packet processing, routers check the packets for its next destination (s). The processing time can be calculated as follow:

$$p_t = n_T \times t_p \quad [2] \quad (4.2)$$

where,  $n_T$  is the number of routers in tree T and  $t_p$  is the time to traverse the tree T.

Transmission time ( $t_t$ ): Transmission time is the amount of time required to push all the packets bits into the transmission medium. In other words, this is the time caused by the data rate of the link.

$$t_t = \frac{\text{Packet Size}}{\text{Bandwidth}} \quad (4.3)$$

Queuing time ( $q_t$ ): Queuing time refers to the time spent in a queue waiting for being executed. Thus, combining all the three types of delays, we calculate end-to-end delay as follows:

$$N_d = p_t + t_t + q_t \quad (4.4)$$

Here, we obtain the  $q_t$  from simulation log and calculate the other two components using Eq. 4.2 - 4.4. We present the end-to-end delays for all the three approaches in Fig. 4.12 - 4.16 present delays for the approaches in all the network topologies under consideration. The figures demonstrate that our proposed approach results in substantially lower delay compared to other two alternatives.

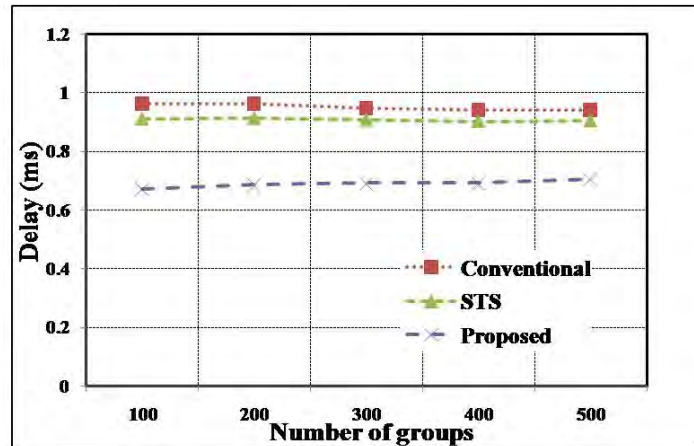


Figure 4.12: Delay in Abilene network

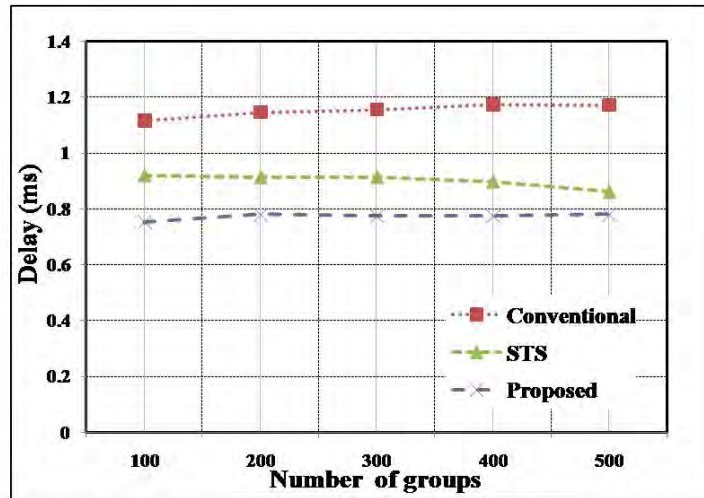


Figure 4.13: Delay in NSFNET network

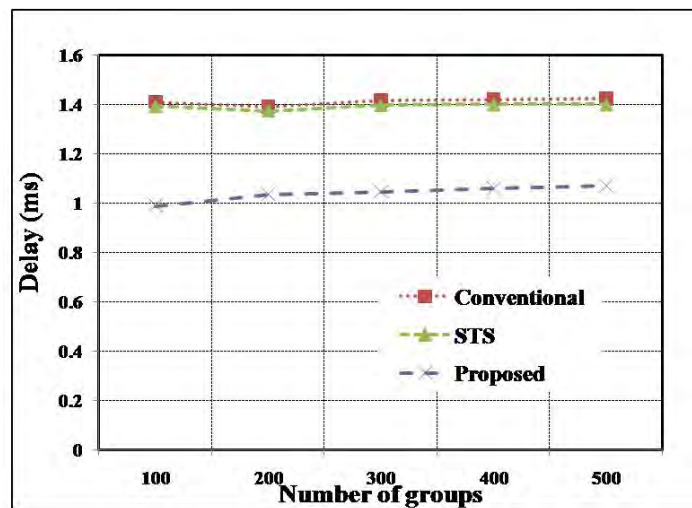


Figure 4.14: Delay in ARPANET network

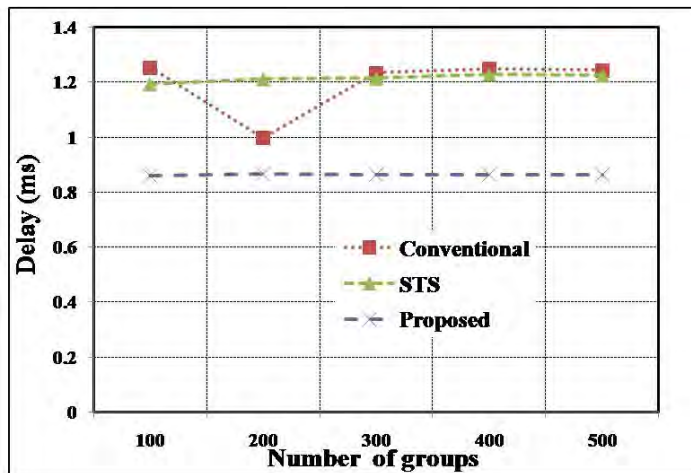


Figure 4.15: Delay in vBNS network

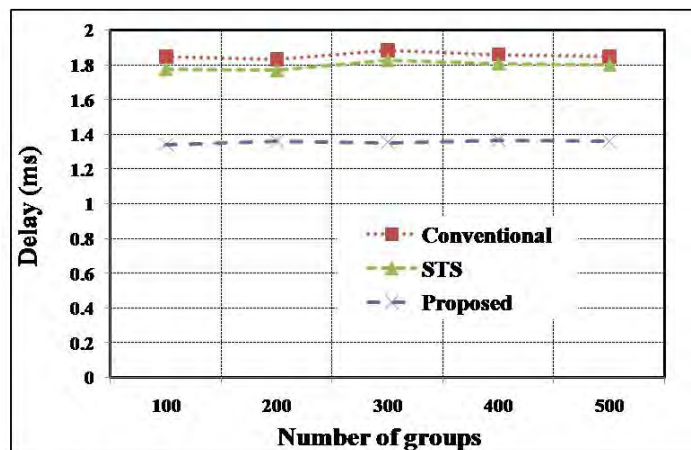


Figure 4.16: Delay in Random network

#### 4.2.4 Performance in Terms of Throughput

We measure network throughput for all the three approaches. Fig. 4.17 - 4.21 show the throughput in all network topologies under consideration. The figures show that our approach results in marginally increased or similar throughput compared to other two approaches.

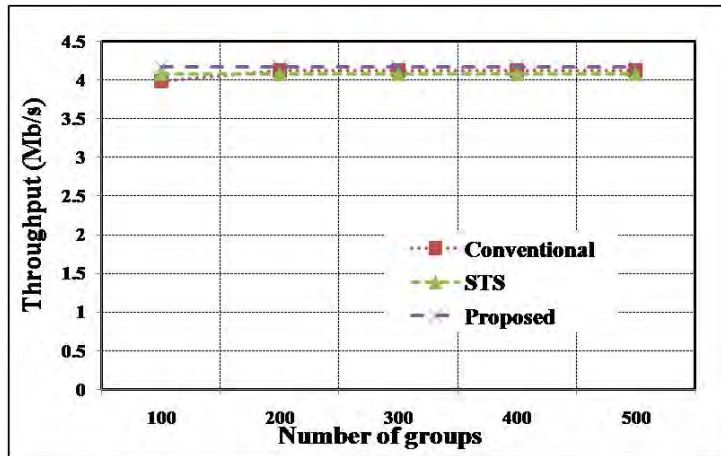


Figure 4.17: Throughput in Abilene network

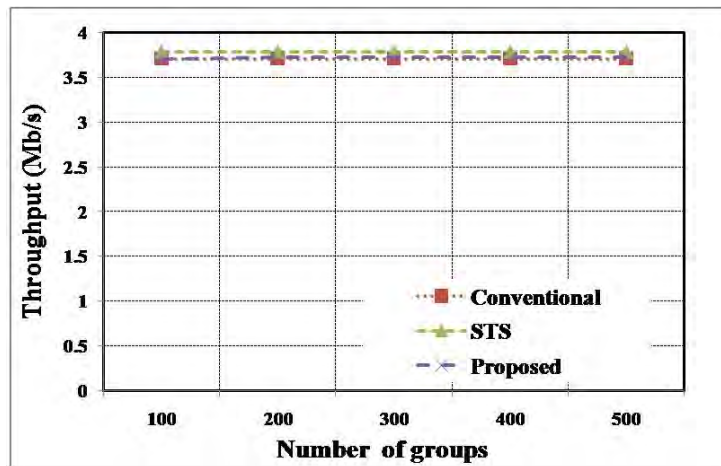


Figure 4.18: Throughput in NSFNET network

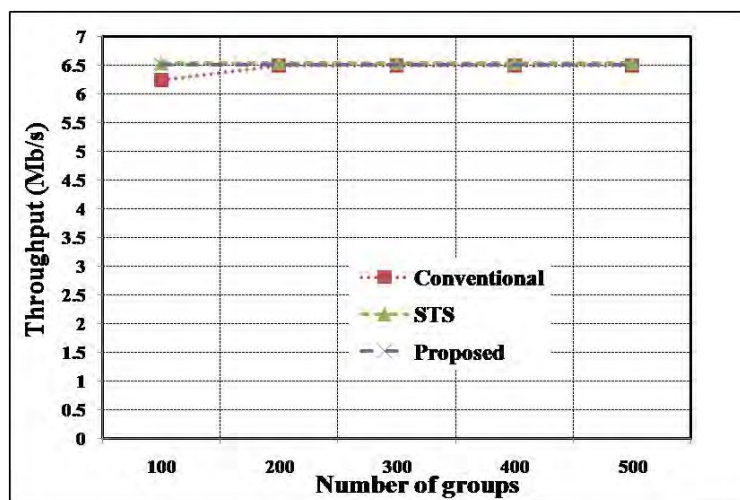


Figure 4.19: Throughput in ARPANET network



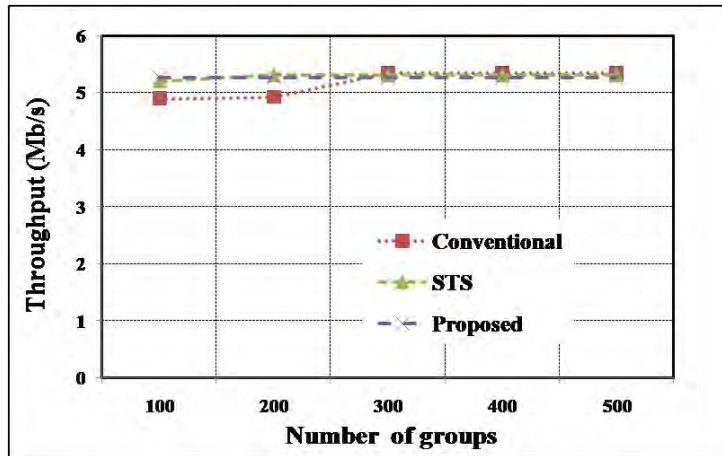


Figure 4.20: Throughput in vBNS network

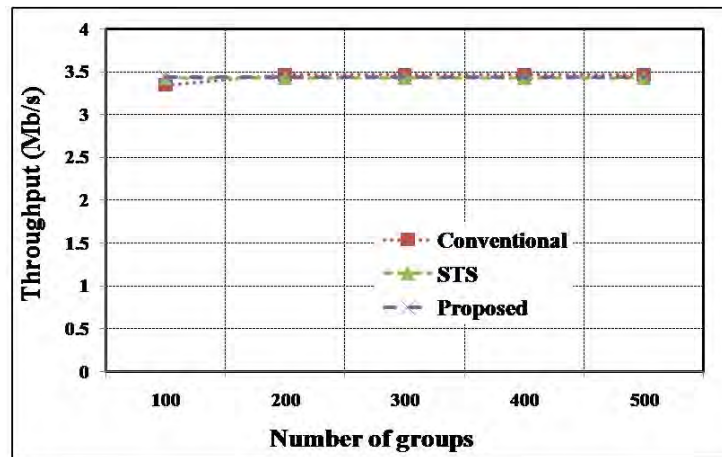


Figure 4.21: Throughput in Random network

### 4.2.5 Summary of Performance Evaluation

We summarize percentages of improvement with respect to different performance metrics using our approach compared to the conventional approach in Table 4.2 - 4.6 and compared to STS in Table 4.7 - 4.11. The tables demonstrate substantial extent of improvement using our approach compared to both other alternatives in term of the number of trees, the number of forwarding states, and delay while having comparable throughput through all the approaches. Next, we study individual impact of all the aspects of our proposed approach. In the subsequent studies, we perform simulation only over Abilene network topology.

Performance and comparison Table given below:

Table 4.2: Performance comparison over conventional method in Abilene network

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	59	70	30	4
200	76	82	28	1
300	83	87	27	1
400	87	90	26	1
500	89	92	25	1

Table 4.3: Performance comparison over conventional method in NSFNET network

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	56	68	32	0
200	74	81	31	0
300	82	87	32	0
400	87	91	34	0
500	89	92	33	0

Table 4.4: Performance comparison over conventional method in ARPANET network

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	44	59	29	4
200	67	74	25	0
300	76	82	26	0
400	79	84	25	0
500	83	87	24	0

Table 4.5: Performance comparison over conventional method in vBNS network

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	46	61	31	7
200	68	72	13	6
300	78	84	30	-1
400	83	88	30	-1
500	86	90	30	-1

Table 4.6: Performance comparison over conventional method in Random network

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	43	57	27	8
200	64	72	25	0
300	75	81	28	0
400	80	85	26	0
500	84	88	26	0

Table 4.7: Summary of performance comparison over conventional method

Network	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
Abilene	79	84	28	2
NSFNET	78	84	33	0
ARPANET	70	78	26	1
vBNS	73	79	27	2
Random	69	77	27	1
Overall	74	81	28	1

Table 4.8: Performance comparison over STS [9] method in Abilene network

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	26	44	25	1
200	34	48	24	1
300	24	40	22	1
400	30	44	22	1
500	29	40	21	1

Table 4.9: Performance comparison over STS [9] method in NSFNET network

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	21	34	18	-1
200	20	30	14	-1
300	21	31	15	-1
400	24	33	13	-1
500	25	31	9	-1

Table 4.10: Performance comparison over STS [9] method in ARPANET network

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	33	51	29	0
200	61	70	24	0
300	71	78	25	0
400	76	81	4	0
500	80	84	23	0

Table 4.11: Performance comparison over STS [9] method in vBNS network

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	34	50	27	1
200	62	71	28	0
300	74	80	28	0
400	80	85	29	0
500	84	88	29	0

Table 4.12: Performance comparison over STS [9] method in Random network

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	35	50	24	4
200	58	67	22	0
300	71	78	25	0
400	77	82	24	0
500	81	85	24	0

Table 4.13: Summary of performance comparison over STS method

Network	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
Abilene	29	44	23	2
NSFNET	23	32	14	-2
ARPANET	65	73	25	0
vBNS	67	76	29	0
Random	65	73	24	1
Overall	50	60	23	0

### 4.3 Performance Analysis of Proposed Refinement on Search Range

In our proposed approach, we refine search range over the existing tree list while attempting aggregation with a new tree. In our search range, searching occurs only on non-smaller trees in the current tree set. If any non-smaller tree in the existing tree set matches with the new multicast session tree, then the new session tree gets aggregated with the existing tree. We analyze impact solely exhibited by this refinement on search range. Fig. 4.22 - 4.25 illustrate the sole impact of proposed refinement on search range on different performance metrics. Table 12 and 13 show percentages of performance improvement achieved through our proposed refined search range. The results show that the proposed refined search range achieves substantial performance improvement compared to the conventional approach in most of the cases. However, it results in marginal degradation in performance compare to STS.

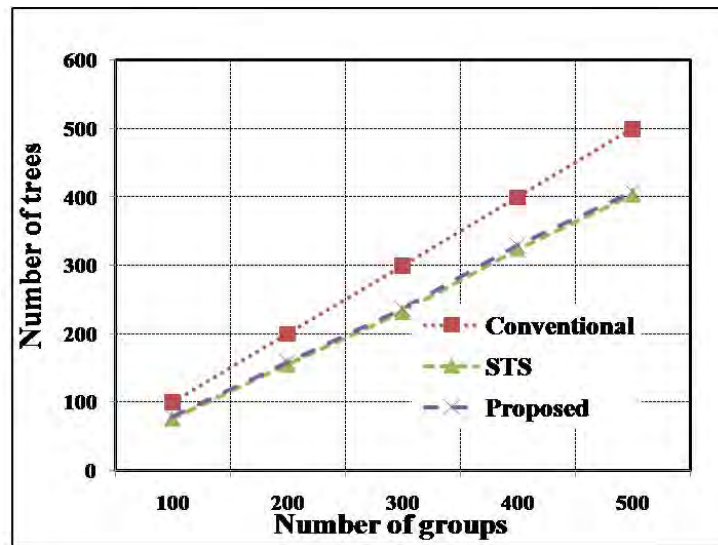


Figure 4.22: Number of trees for proposed search range

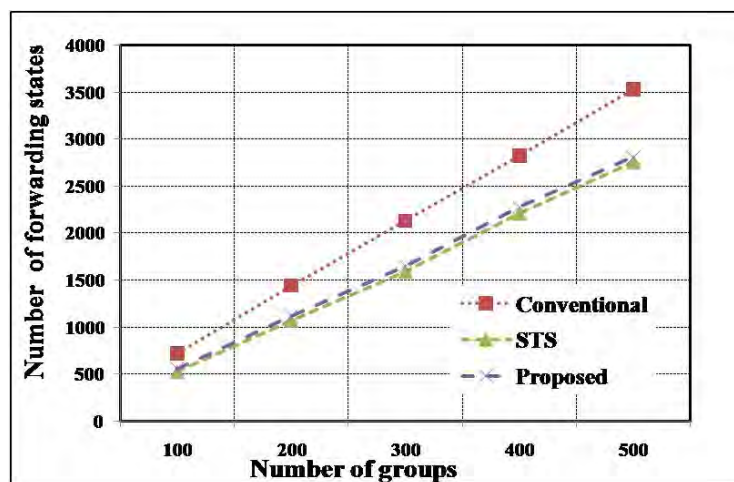


Figure 4.23: Number of forwarding states for proposed search range

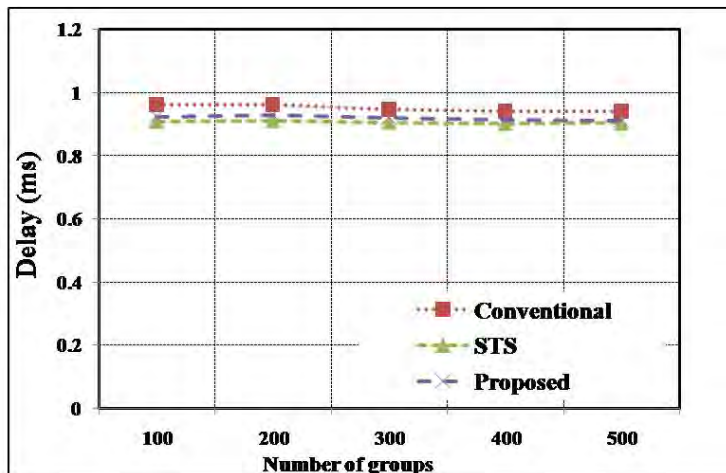


Figure 4.24: Delay for proposed search range

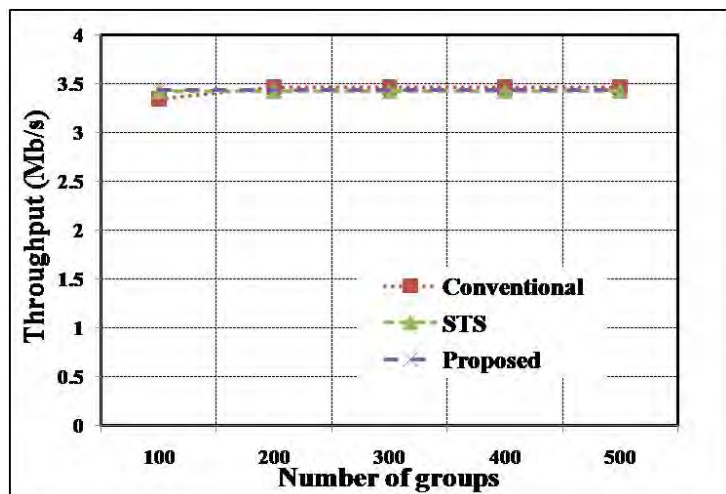


Figure 4.25: Throughput for proposed search range

Table 4.14: Percentages of improvement using only our proposed refined search range over conventional method

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	34	50	27	1
200	62	71	28	0
300	74	80	28	0
400	80	85	29	0
500	84	88	29	0

Table 4.15: Percentages of improvement using only our proposed refined search range over STS method

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	-3	-5	-1	0.4
200	-2	-4	-1	0.4
300	-2	-3	-1	0.4
400	-1	-3	-1	0.4
500	-1	-2	0	0.4

## 4.4 Performance Analysis of Proposed Consideration of Overlapping Degree

In this study, we redefine the notion of overlapping degree to improve the criteria of selection of an existing tree for aggregation based on the overlapping degree. For calculating overlapping degree, we overlap destination nodes of a new multicast session tree with destination nodes of a tree in the existing tree set. In our proposed method, threshold on the overlapping degree is set to 0.85 following the study presented in [9]. If the overlapping degree of an existing tree in the search range is found to be greater than the threshold, then the new multicast session tree gets aggregated with the existing tree. Fig. 4.26 - 4.29 illustrate the sole impact of proposed overlapping degree on different performance metrics. Table 4.14 and 4.15 show percentages of performance improvement achieved solely by our proposed overlapping degree. The results demonstrate that the proposed refinement on overlapping degree results in substantial performance improvement compared to the conventional approach and smaller performance improvement compared to the STS approach.



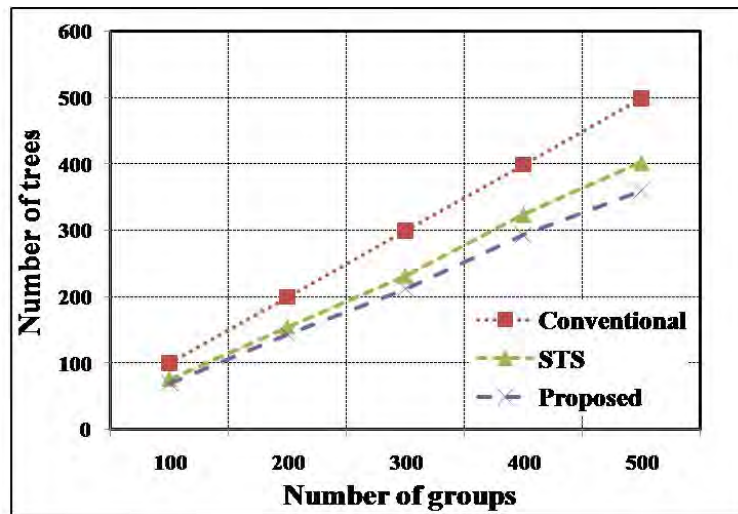


Figure 4.26: Number of trees for proposed overlapping degree

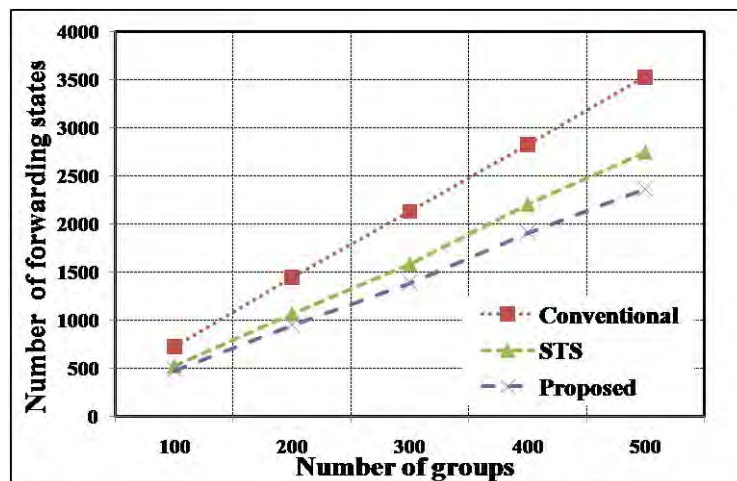


Figure 4.27: Number of forwarding states for proposed overlapping degree

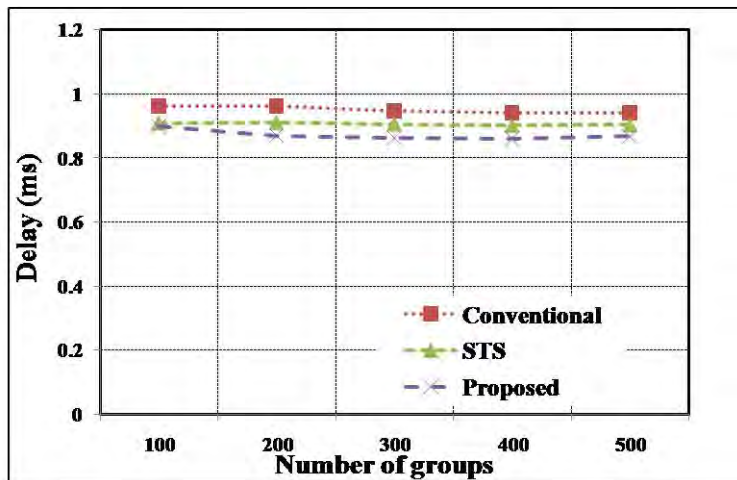


Figure 4.28: Delay for proposed overlapping degree

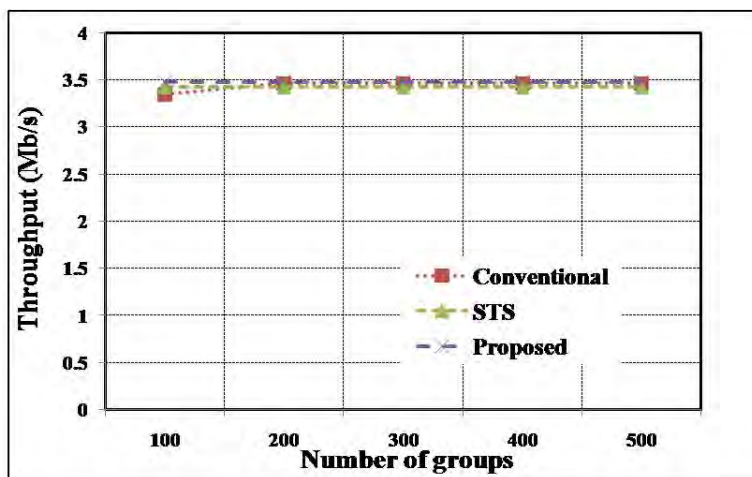


Figure 4.29: Throughput for proposed overlapping degree

Table 4.16: Percentages of improvement using only our proposed refined overlapping degree over conventional method

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	30	33	6	3
200	28	34	9	0
300	29	34	8	0
400	26	32	8	0
500	28	32	7	0

Table 4.17: Percentages of improvement using only our proposed refined overlapping degree over STS method

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	7	8	1	1
200	7	10	4	1
300	8	12	4	1
400	9	13	4	1
500	10	13	4	1

## 4.5 Performance analysis of Proposed Replacement Method

Due to limitation of storage capacity, it is unfeasible to keep infinite number of entries for shared-trees in the routing table. To solve this problem, stale shared trees are generally replaced to remain within a limited storage demand. Here, we propose a new replacement method. We use least recently aggregated metric rather than the classically used notion of least frequently used [9]. In this replacement method, when the number of trees registered in tree set exceeds the permitted size of the tree set as per the storage limitation, the replacement method deletes least recently aggregated tree. Fig. 4.30 - 4.33 show the improvement in performance in terms of different metrics using our proposed replacement method. Table 16 and 17 present corresponding percentages of improvement using the proposed replacement method. The results show that the proposed replacement method achieves substantial performance improvement compared to the conventional approach and STS approach.

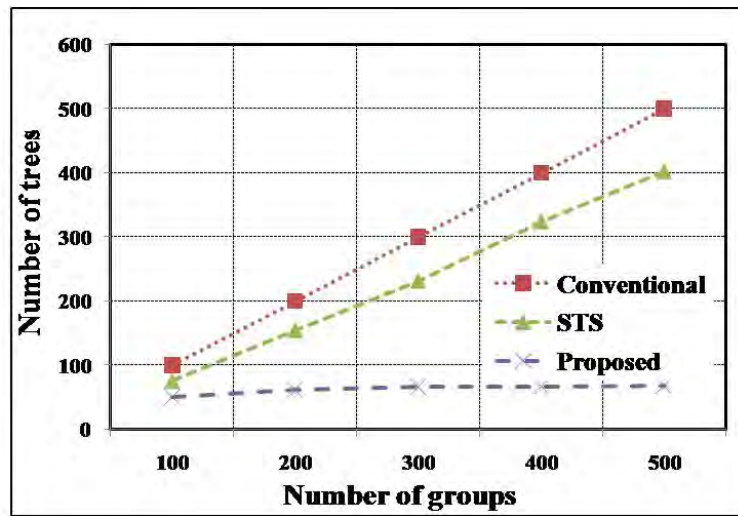


Figure 4.30: Number of trees for proposed replacement method

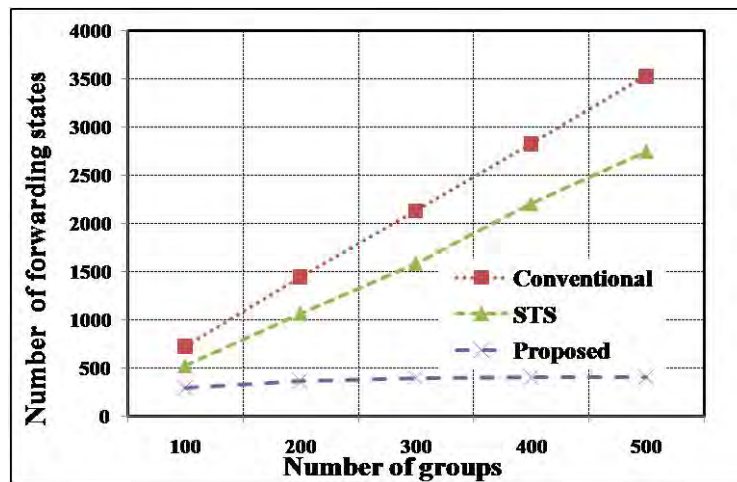


Figure 4.31: Number of forwarding states for proposed replacement method

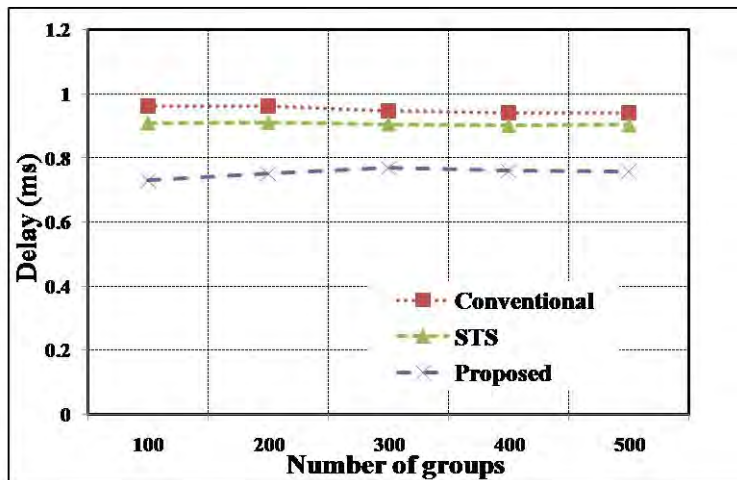


Figure 4.32: Delay for proposed replacement method

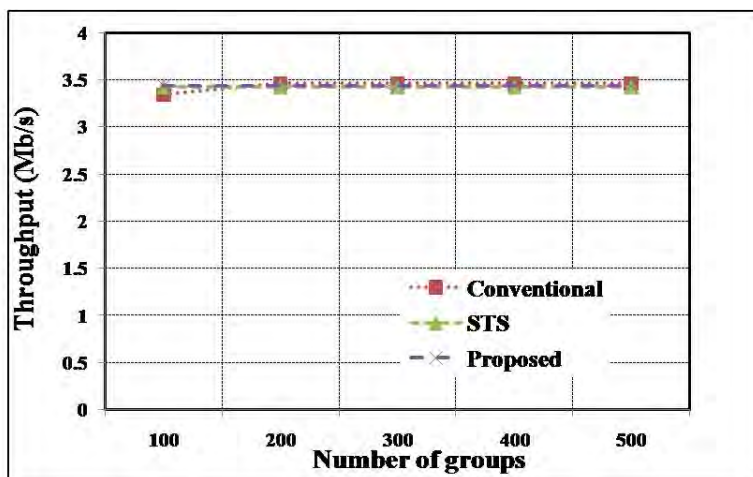


Figure 4.33: Throughput for proposed replacement method

Table 4.18: Percentages of improvement using only our proposed replacement method over conventional method

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	49	59	24	2
200	69	75	21	0
300	77	81	18	0
400	83	85	19	0
500	86	88	19	0

Table 4.19: Percentages of improvement using only our proposed replacement method over STS method

Number of groups	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
100	32	44	19	0
200	60	66	17	0
300	71	75	15	0
400	79	81	15	0
500	82	85	16	0

Table 4.20: Performance improvement summary of our different proposed methods over conventional method

Proposed method	Number of trees	Number of forwarding states	Delay	Throughput
Search Range	19	21	3	0
Overlapping degree	28	33	8	1
Replacement method	72	78	20	0

Table 4.21: Performance improvement summary of our different proposed methods over STS method

Proposed method	Number of trees	Number of forwarding states	Delay	Throughput
Search Range	-2	-3	-1	0
Overlapping degree	8	11	3	1
Replacement method	65	70	16	0

## 4.6 Variation in Simulation Time

We perform our simulation with varying simulation time to analyze sensitivity of our reported results with respect to the simulation time. Here, we vary the simulation time over 5 - 25 sec in the Abilene network. Fig. 4.34 - 4.37 illustrate the impact of variation in simulation time on different performance metrics. The figures show that performance of different metrics remains fixed for different simulation time after 5 sec.

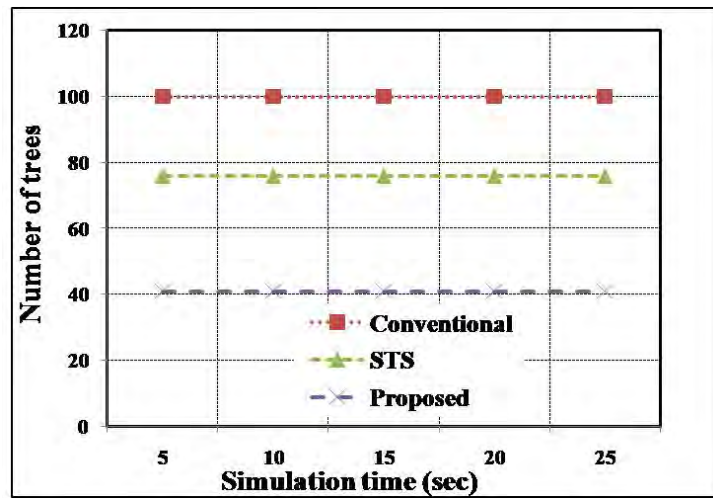


Figure 4.34: Effect of variation in simulation time on the number of trees

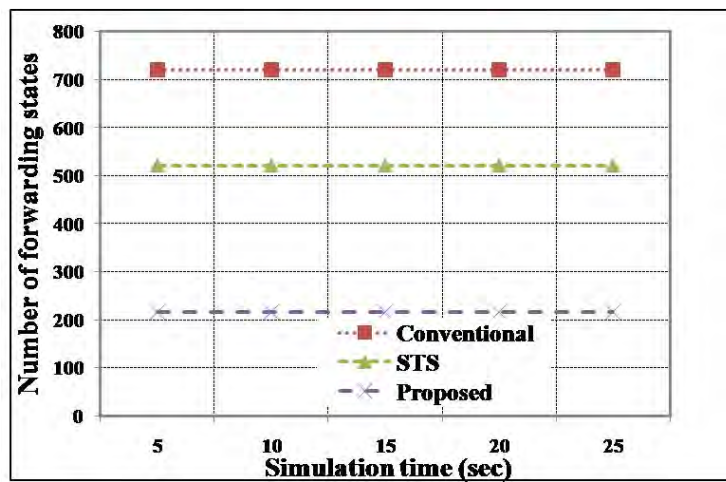


Figure 4.35: Effect of variation in simulation time on the number of forwarding states

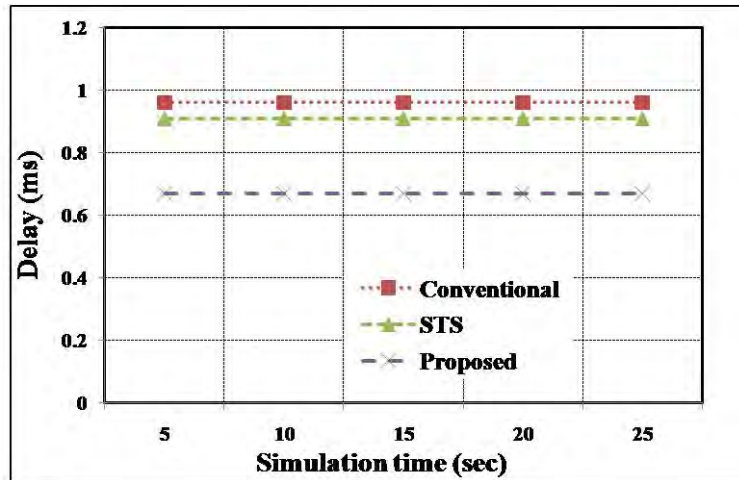


Figure 4.36: Effect of variation in simulation time on delay

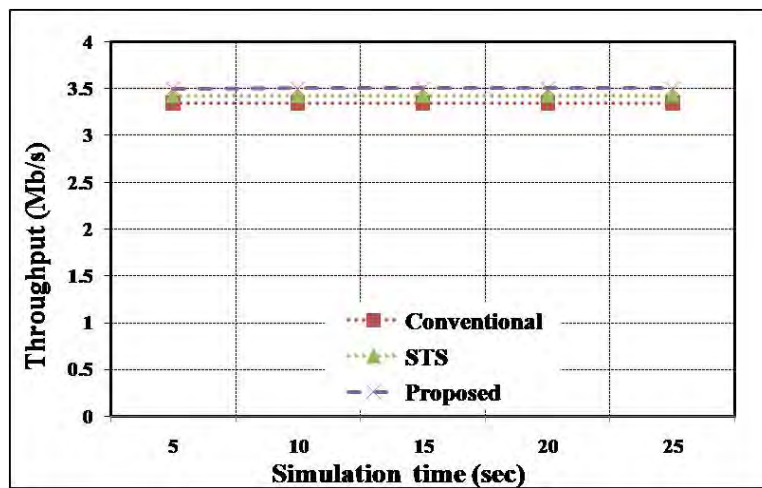


Figure 4.37: Effect of variation in simulation time on throughput



## 4.7 Variation in Data Rate

We further conduct our simulation with varying the data rate to analyze sensitivity of our reported results with respect to the data rate. Here, we vary the data rate (application's rate of data sending) over 1 - 5 Mbps in the Abilene network. Fig. 4.38 - 4.41 illustrate the impact of variation in data rate on different performance metrics. The figures show that the number of trees, the number of forwarding states, and delay remain fixed for different data rates. The only exception is throughput, which gets improved with an increase in data rate. However, here, the noteworthy observation is that the relative values of throughput in all the three mechanisms remain more or less similar irrespective of variation in the data rate. The close appearances of the throughput curves for the three mechanisms in Fig.4.41 justify it. Therefore, the change in data rate does not exhibit any significant sensitivity to our simulation results.

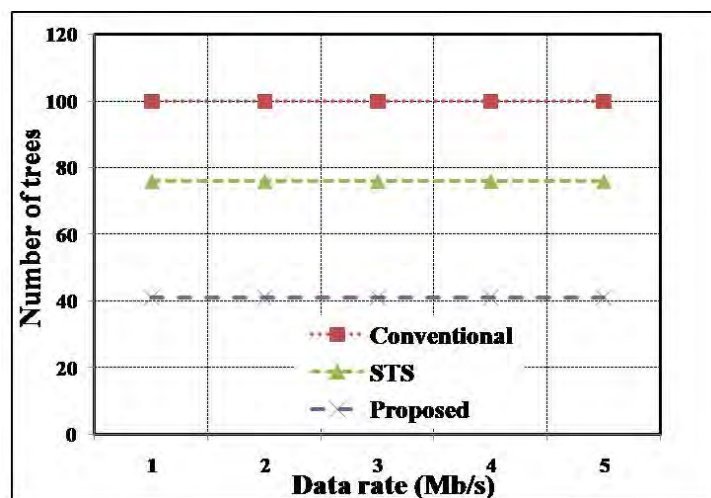


Figure 4.38: Effect of variation in data rate on the number of trees

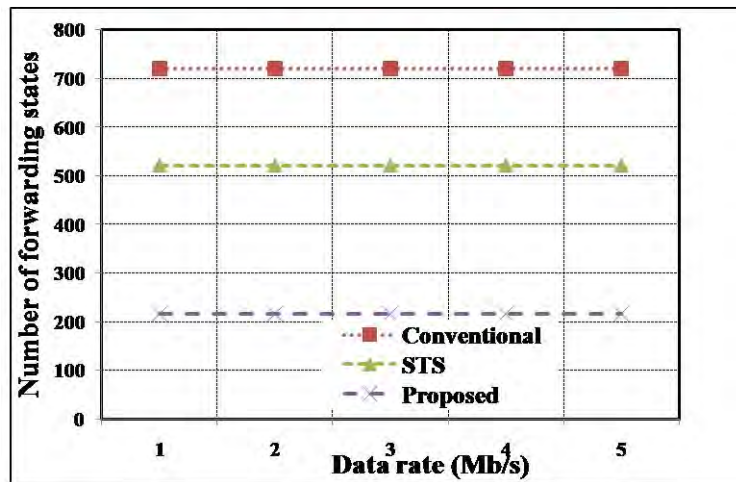


Figure 4.39: Effect of variation in data rate on the number of forwarding states

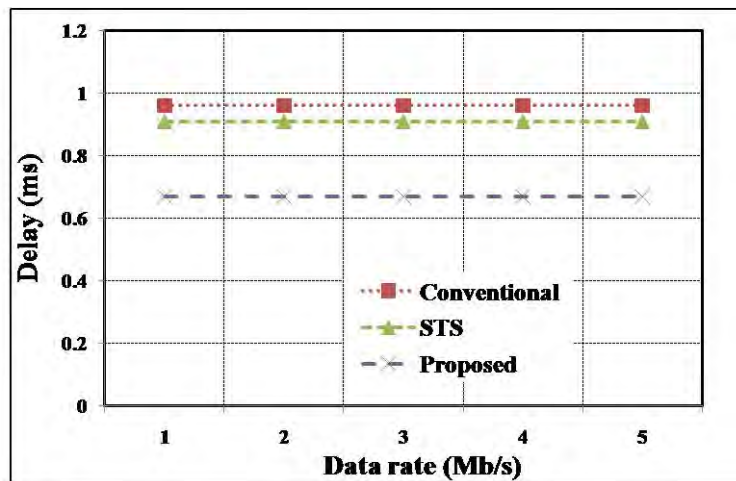


Figure 4.40: Effect of variation in data rate on delay

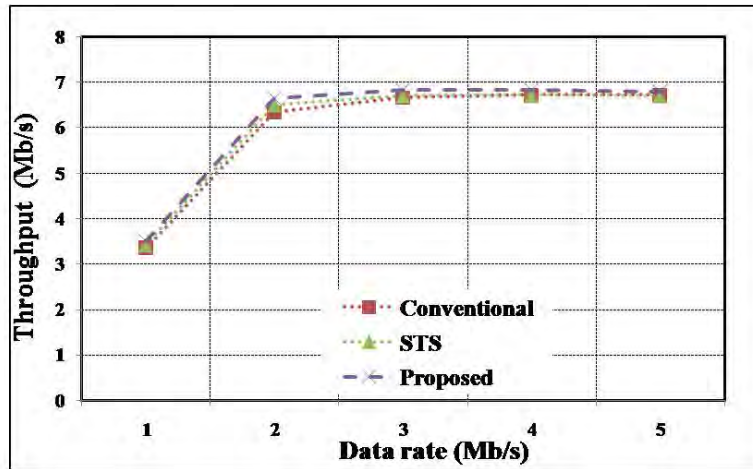


Figure 4.41: Effect of variation in data rate on throughput

## 4.8 Variation in Bandwidth

We execute the simulation with varying bandwidth of communication channel to analyze sensitivity of our reported results with respect to bandwidth. Here, we vary the bandwidth over 100 - 500 Mbps over the Abilene network. Fig. 4.42 - 4.45 demonstrate the impact of variation in bandwidth on different performance metrics. The figures show that the number of trees, the number of forwarding states, and delay remain fixed for different bandwidth. The only exception is throughput, which gets improved with an increase in bandwidth. However, similar to the earlier case, values of throughput in different mechanisms exhibit similar relative values as portrayed by the close appearances of all the throughput curves in Fig.4.45. Thus, the change in bandwidth does not exhibit any significant sensitivity to our simulation results.

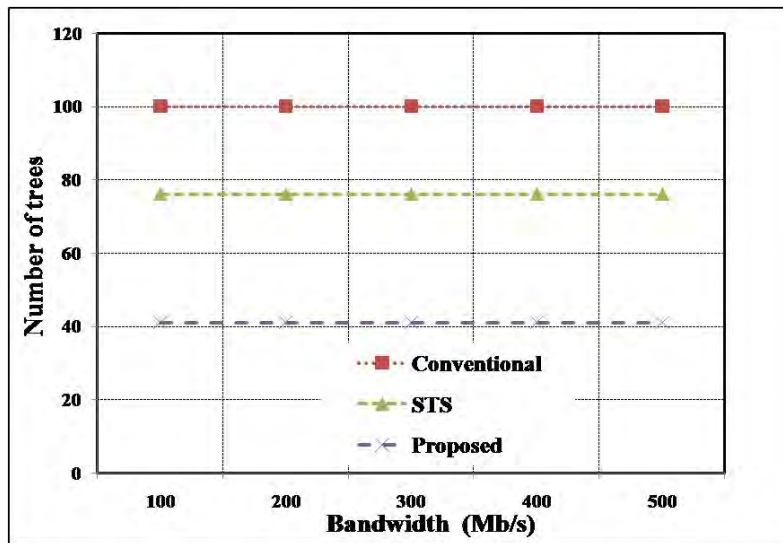


Figure 4.42: Effect of variation in bandwidth on the number of trees

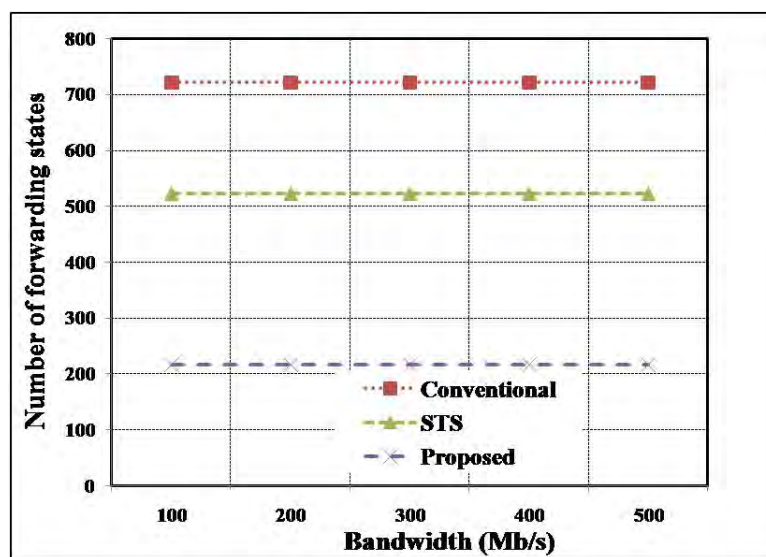


Figure 4.43: Effect of variation in bandwidth on the number of forwarding states

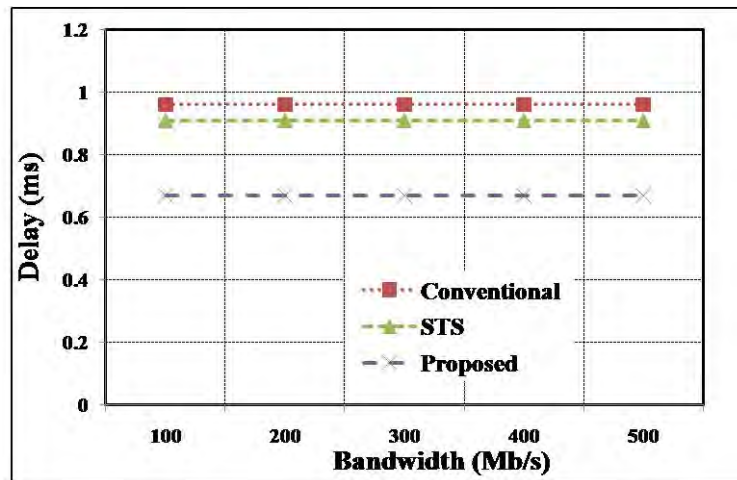


Figure 4.44: Effect of variation in bandwidth on delay

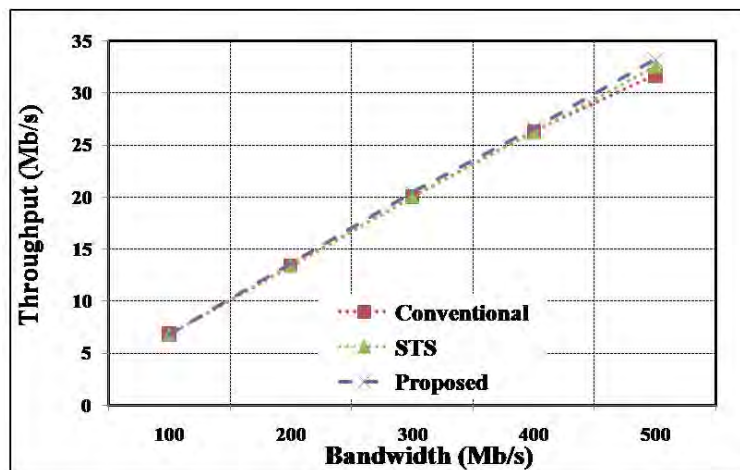


Figure 4.45: Effect of variation in bandwidth on throughput

## 4.9 Variation in Coincidence Degree (P)

We perform our simulation in response to a variation in the values of coincidence degree. We vary the coincidence degree over the range 0.75 - 0.95. Fig. 4.46 - 4.49 demonstrate the impact of variation in coincidence degree on different performance metrics. Here, Fig. 4.46 - 4.47 show that the number of trees and the number of forwarding states increase with an increase in P after the value 0.85 for our proposed approach and STS. However, for the conventional approach, P does not have any impact. Besides, Fig. 4.48 shows that delay substantially decreases with an increase in P after P=0.70. For STS, the impact is reverse -delay increases with an increase in P after P=0.85. Additionally, similar to the previous case, the conventional method does not exhibit any impact here too. Finally, for throughput, there is no noteworthy impact in any of the mechanisms with an increase in P.

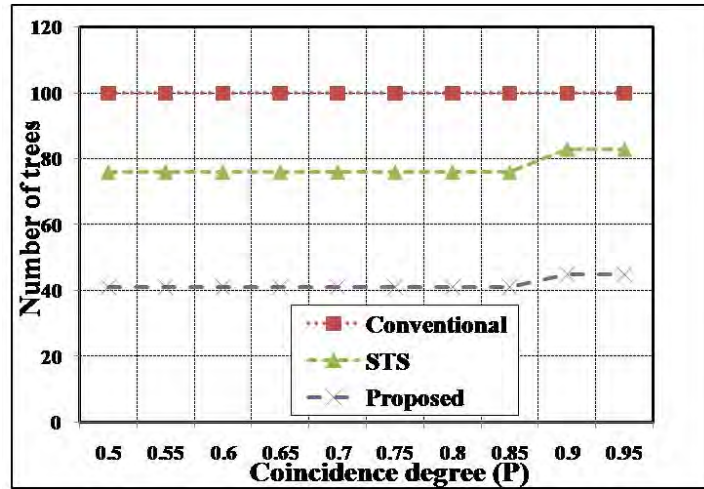


Figure 4.46: Effect of variation in coincidence degree on the number of trees

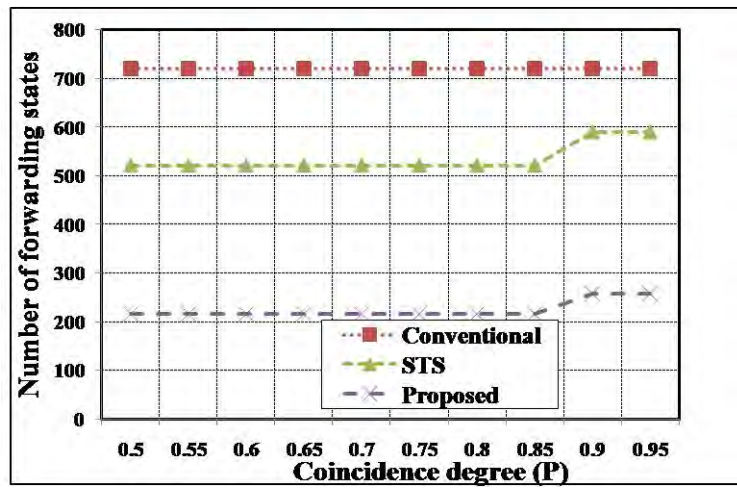


Figure 4.47: Effect of variation in coincidence degree on the number of forwarding states

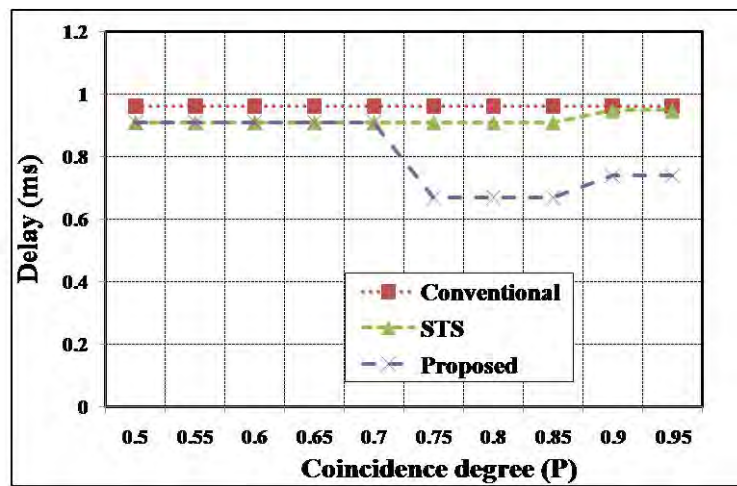


Figure 4.48: Effect of variation in coincidence degree on delay

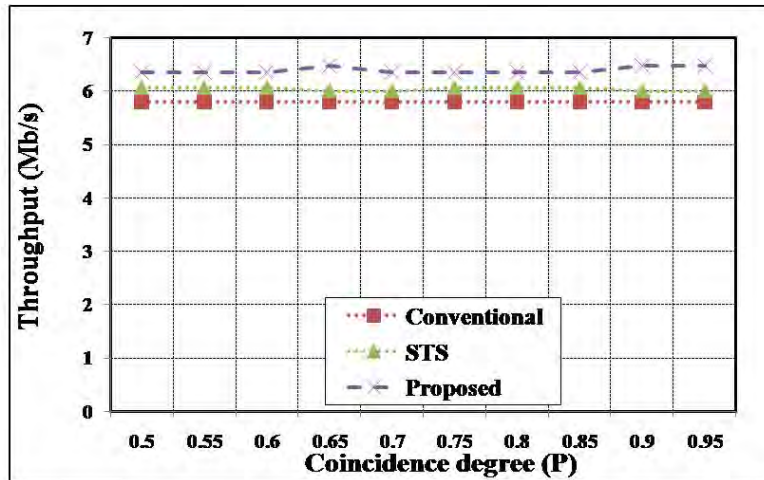


Figure 4.49: Effect of variation in coincidence degree on throughput

### 4.10 Variation in Overlapping Degree ( $\gamma$ )

We perform our simulation on different network topologies with varying values of the overlapping degree. Here, we vary the overlapping degree over the range 0.75 - 0.95. Fig.4.50 - 4.53 demonstrate the impact of variation in overlapping degree on different performance metrics. Fig.4.50 - 4.52 show that the number of trees, the number of forwarding states, and delay mostly increases with an increase in  $\gamma$  for our proposed mechanism and STS. For throughput, there is no such noteworthy trend. Besides, for the conventional approach, performance does not get varied with an increase in  $\gamma$ .

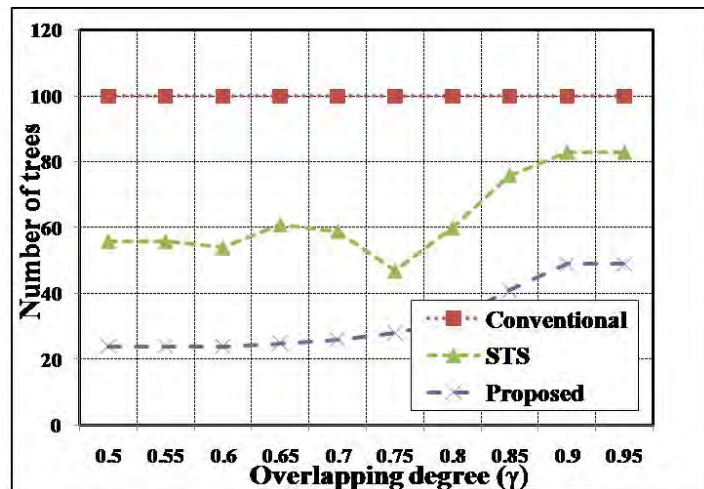


Figure 4.50: Effect of variation in overlapping degree on the number of trees



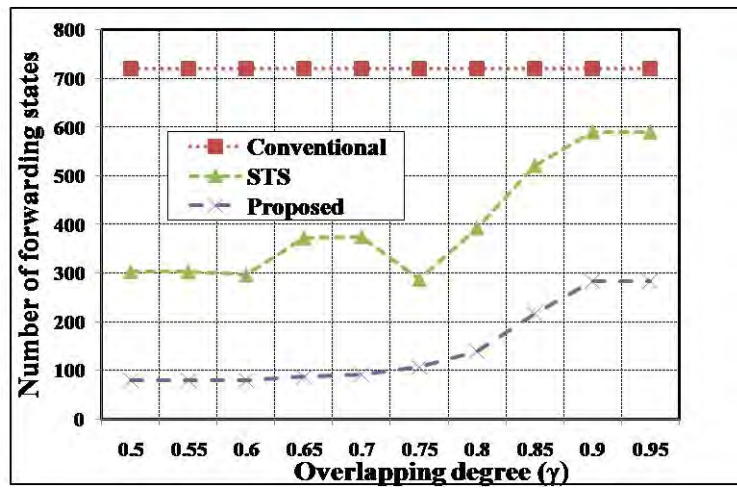


Figure 4.51: Effect of variation in overlapping degree on the number of forwarding states

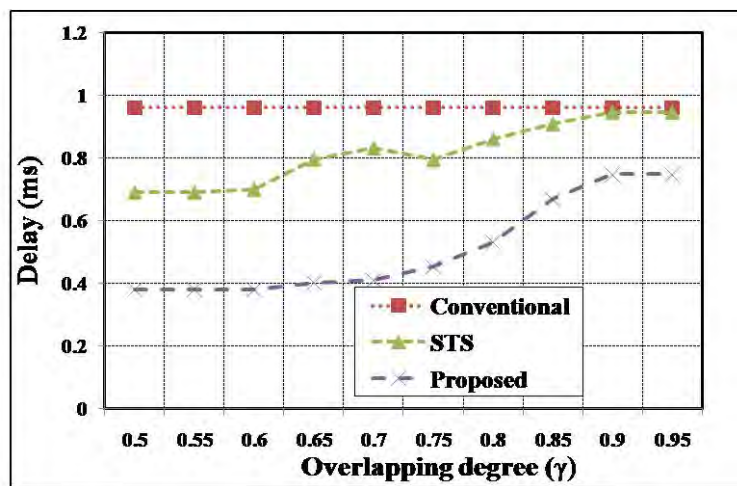


Figure 4.52: Effect of variation in overlapping degree on delay

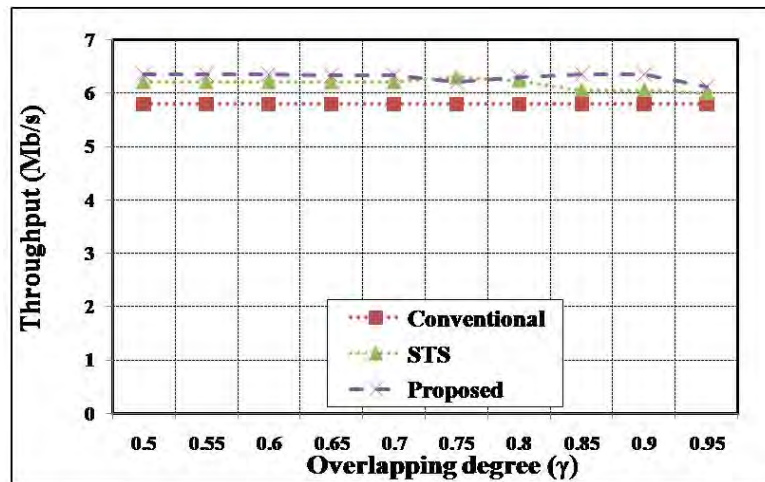


Figure 4.53: Effect of variation in overlapping degree on throughput

## 4.11 Effects of Different Iteration

We perform our simulation for different iterations to analyze sensitivity of our reported results with respect to the iterations. Here, we conduct our simulation for five different iterations over the Abilene network. Fig. 4.54 - 4.57 illustrate the outcomes in different iteration in term of different performance metrics. The figures demonstrate that performances in term of all the metrics remain same in different iterations for all the approaches under consideration alleviating the general need of having several number of iterations for reporting simulation results.

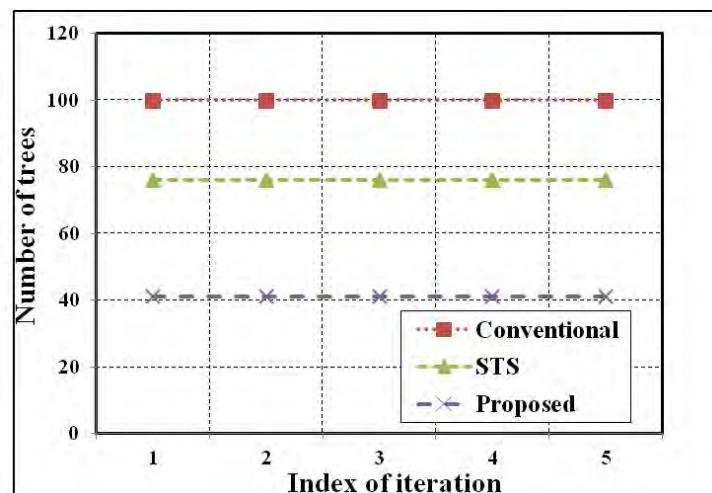


Figure 4.54: Effect of different iterations on the number of trees

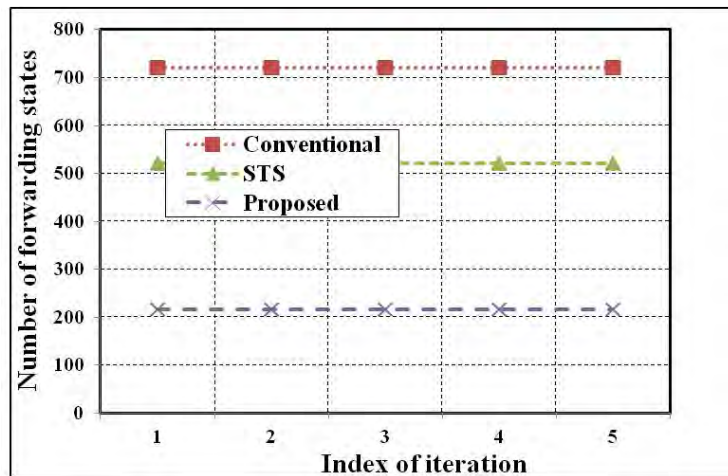


Figure 4.55: Effect of different iterations on the number of forwarding states

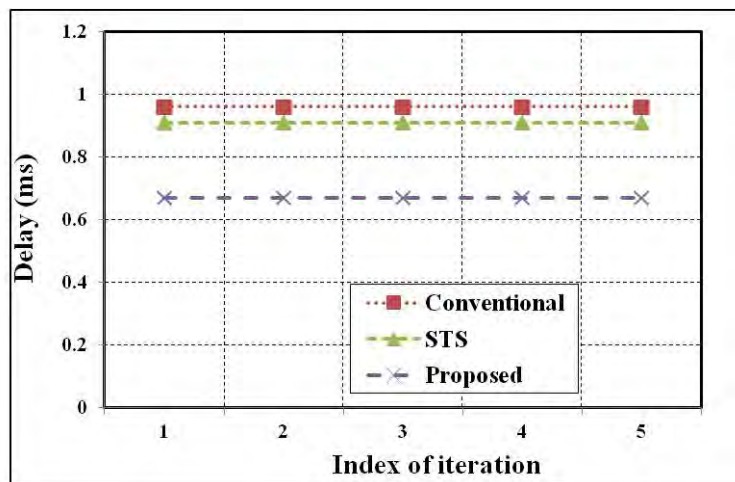


Figure 4.56: Effect of different iterations on delay

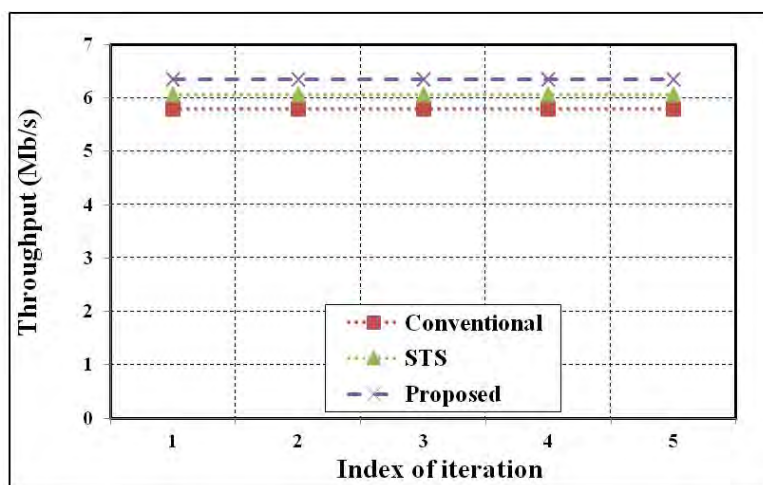


Figure 4.57: Effect of different iterations on throughput

## 4.12 Variation in Relative Numbers of Large and Small Groups

We perform our simulation with varying relative numbers of large and small multicast groups to analyze whether the relative numbers (or ratio, in another way) exhibits any significant impact over the performance achieved through our approach. To analyze the impact, we adopt a few multicast groups large groups while having all other groups as small group. Here, we consider only the case of having substantially more number of small groups, as our proposed approach can be suspected to work worse only in this case for performing the aggregating task.

Fig. 4.58 - 4.61 show the outcomes of having different relative numbers of large and small groups. Here, we take five large groups each having ten members in all cases. Besides, we vary the number of small groups over a range of 95 - 495 with a granularity of 100 keeping the total number of groups (including both large and small groups) over a range of 100 - 500. The small groups have the number of members over a range of 2 - 4.

The figures show that the number of trees, the number of forwarding states, and delay mostly increases with an increase in the number of small groups, i.e., the relative number of small groups. In all these cases, the performances of our proposed approach exhibit better outcomes. On the other hand, in case of throughput, there is marginally performance improvement compared to the conventional method and marginally performance degradation compared to STS method using our proposed approach.

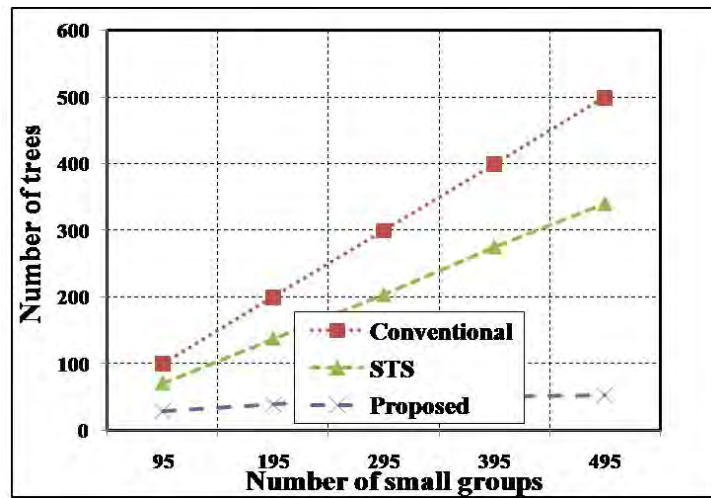


Figure 4.58: Effect of variation in relative numbers of large and small groups on the number of trees

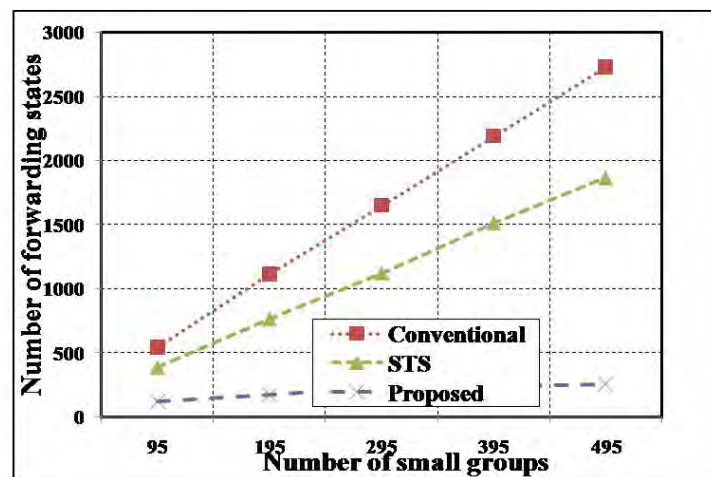


Figure 4.59: Effect of variation in relative numbers of large and small groups on the number of forwarding states

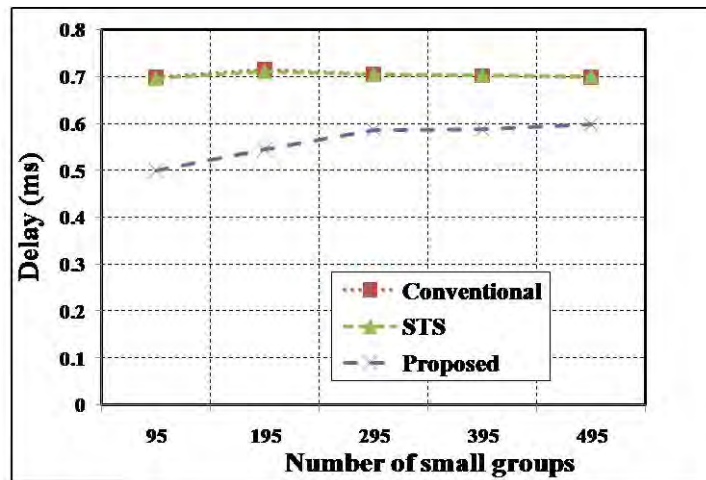


Figure 4.60: Effect of variation in relative numbers of large and small groups on delay

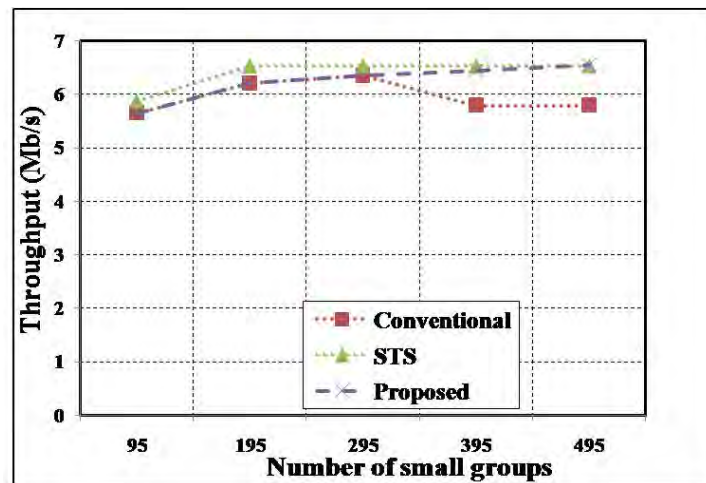


Figure 4.61: Effect of variation in relative numbers of large and small groups on throughput

### 4.13 Simulation Findings

Table 4.2 - 4.11 summarize the percentages of improvement with respect to different network performance metrics such as the number of trees, the number of forwarding states, delay, and throughput using our proposed approach. Here, we summarize the key findings achieved from our simulation study on the different network parameters and network topology.

- Different network performance parameters such as the number of trees, the number of forwarding states, delay, and throughput exhibit variations after adopting different multi-

cast methods. From our simulation, we find that our proposed approach performs better compared to other multicast approaches in response to these parameters.

- Our proposed approach performs better than that of the existing alternative multicast approaches mostly in terms of the number of trees, the number of forwarding states, and delay. For throughput, our proposed approach provides comparable or even marginally better performance compared to the other two alternatives.

Table 4.22: Summary of average percentage of improvement

Method under comparison	% of improvement			
	Number of trees	Number of forwarding states	Delay	Throughput
Conventional	73	80	27	1.5
STS	49	59	21	0

# Chapter 5

## Conclusion and Future Work

Conventional approaches for multicast often fail to address scalability issue. Therefore, in this study, we present a new aggregated multicast approach. Our new approach proposes new aggregated multicast tree selection method, refined search range, and a new notation of overlapping degree for replacing stale trees. The main idea of our approach is to force multiple multicast groups to share single tree based on our refined search range, replacing old trees based on our definition of overlapping degree if needed. Here, our aim is to solve multicast scalability problem by reducing the number of forwarding states in the Network layer and show the effects of tree aggregation in terms of network performance metrics such as delay and throughput. Less number of forwarding state achieved by our approach require less memory to store states information, and less processing time to forward a packet to its next destination.

We implement our proposed approach in network simulator ns-3. Afterwards, we analyze the performance of our proposed approach for different performance metrics and compare with that of the existing alternatives. Our proposed approach exhibit significant performance improvement over existing approaches for different network metrics such as the number of trees, the number of forwarding states, delay, and throughput. Simulation results show that our approach can reduce up to 92% of forwarding states and 34% of delay over conventional multicast, and reduce up to 88% of forwarding states and 29% of delay over STS method.

In future, we plan to perform performance evaluation of our proposed approach using real network settings. Besides, we will modify our aggregated multicast scheme for throughput



improvement, in addition to already - achieved delay improvement.

# Bibliography

- [1] L. Qian, Y. Tang, Y. Wang, B. B. Diab, and W. Olesinski. A new scalable multicast solution in MPLS networks. In Proceedings of the Global Communications Conference. IEEE, 2006.
- [2] S. Bhattacharyya. An overview of source-specific multicast (SSM). RFC 3569, 2003.
- [3] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei. Protocol independent multicast sparse mode (PIM-SM): protocol specification. RFC 2362, 1998.
- [4] A. Fei, J. Cui, M. Gerla, and M. Faloutsos. Aggregated multicast: an approach to reduce multicast state. In Proceedings of the Global Communications Conference. IEEE, 2001.
- [5] A. Boudani and B. Cousin. A new approach to construct multicast trees in MPLS networks. In Proceedings of the ISCC 2002 Seventh International Symposium on Computers and Communications. IEEE, 2002.
- [6] A. Boudani and B. Cousin. Multicast tree in MPLS network. In Proceedings of the Global Communications Conference. IEEE, 2005.
- [7] T. Ballardie, P. Francis, and J. Crowcroft. Core based trees (CBT). In Proceedings of the SIGCOMM 93 Conference, pages 85-95, 1993.
- [8] A. Guitton and J. Moulrierac. Scalable tree aggregation for multicast. ICRN, IRISA, 2005.
- [9] Y. Sekine, T. Mikoshi, and T. Takenaka. Shared-tree selection method for aggregated multicast. Journal of Communication and Computer, 10(4):579-583, 2013.

- 
- [10] G. L. Aceves and M. M. Barijough. Efficient multicasting in content centric networks using locator-based forwarding state. In Proceedings of the 2017 International Conference on Computing, Networking and Communications. IEEE, 2017.
- [11] A. Craig, B. Nandy, I. Lambadaris, and P. Koutsakis. BloomFlow: openflow extensions for memory efficient, scalable multicast with multi-stage bloom filters. *Computer Communications*, 110:83-102, 2017.
- [12] H. Wang, X. Meng, and M. Zhang. Tabu search algorithm for RP selection in PIM-SM multicast routing. *Computer Communications*, 33(1):35-42, 2010.
- [13] A. Sundarrajan and S. Ramasubramanian. Fast rerouting for IP multicast under single node failures. In Proceedings of the Global Communications Conference. IEEE, 2013.
- [14] N. F. Mir, S. M. Musa, R. Torresand, and S. Swamy. Evaluation of PIM and CBT multicast protocols on fault-tolerance. *International Journal of Computing and Network Technology*, 2(2):59-64, 2014.
- [15] G. Lencse and I. Derka. Investigation of the fault tolerance of the PIM-SM IP multicast routing protocol for IPTV purposes. *Infocommunications Journal*, 5(1):21-28, 2013.
- [16] V. Chandrasekar and K. Baskaran. Performance of video conferencing in unicast and multicast communication using protocol independent multicast routing. *International Journal of Computer Science and Telecommunications*, 2(9):34-36, 2011.
- [17] X. Li, H. Zhang, J. Chang, J. Chen, and H. Chao. Multicast access control implementation in PIM-SM. *Wireless Personal Communications*, Springer Science+Business Media, 55(1):35-49, 2009.
- [18] I. S. Khan, A. Tripathi, and A. A. Shaikh. Analysis of performance of core based tree and centralized mode of multicasting routing protocol. *International Journal of Scientific and Research Publications*, 3(5):1-7, 2013.
- [19] M. Kabat, M. Patel, and C. Tripathy. A heuristic algorithm for core selection in multicast routing. *Journal of Computer Science and Technology*, 26(6):954-961, 2011.

- 
- [20] H. Lin and S. Lai. Core placement for the core based tree multicast routing architecture. In Proceedings of the Global Communications Conference. IEEE, 1998.
- [21] S. Park and D. Park. Adaptive core multicast routing protocol. *Wireless Networks*, 10(1):53-60, 2004.
- [22] A. Kulkarni, Y. Pino, M. French, and T. Mohsenin. Real-time anomaly detection framework for many-core router through machine-learning techniques. *ACM Journal on Emerging Technologies in Computing Systems*, 13(1):1-22, 2016.
- [23] K. M. Yu and S. H. Wu. An efficient load balancing multi-core frequent patterns mining algorithm. In Proceedings of the TrustCom, pages 1408-1412. IEEE, 2011.
- [24] Y. Gao, Y. Ge, and J. Hou. Reliable multicasts for core-based multicast trees. In Proceedings of the 2000 International Conference on Network Protocols. IEEE, 2000.
- [25] J. Yanez and V. Chouliaras. A configurable statistical lossless compression core based on variable order Markov modeling and arithmetic coding. *IEEE Transactions on Computers*, 15(11):1345-1359, 2005.
- [26] L. Schwiebert and R. Chintalapati. Improved fault recovery for core based trees. *Computer Communications*, 23(9):816-824, 2000.
- [27] W. Jia, W. Zhao, D. Xuan, and G. Xu. An efficient fault-tolerant multicast routing protocol with core-based tree techniques. *IEEE Transactions on Parallel and Distributed Systems*, 10(10):984-1000, 1999.
- [28] E. Rosen et al.. Multiprotocol label switching architecture. RFC 3031, 2001.
- [29] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow. Extensions to RSVP for LSP tunnels. IETF RFC 3209, 2001.
- [30] L. Andersson, E. I. Minei, and E. B. Thomas. LDP specification. IETF RFC 3036, 2007.
- [31] J. Tian and G. Neufeld. Forwarding state reduction for sparse mode multicast communication. In Proceedings of the IEEE INFOCOM 98, the Conference on Computer Communications. IEEE, 1998.

- 
- [32] I. Stoica, T.S.E. Ng, and Hui Zhang. A recursive unicast approach to multicast. In Proceedings of the IEEE INFOCOM 2000. IEEE, 2000.
- [33] M. Reed, M. A. Naday, and N. Thomos. Stateless multicast switching in software defined networks. In Proceedings of the 2016 IEEE International Conference on Communications (ICC). IEEE, 2016.
- [34] T. Barabas, D. Ionescu, and S. Veres. A traffic engineering algorithm for differentiated multicast services over MPLS networks. In Proceedings of the 2012 7th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI). IEEE, 2012.
- [35] A. Zahemszky, P. Jokela, M. Sarela, S. Ruponen, J. Kempf, and P. Nikander. Multiprotocol stateless switching. In Proceedings of the 2010 INFOCOM IEEE Conference on Computer Communications Workshops. IEEE, 2010.
- [36] G. Fernandez, D. Larrabeiti, and J. A. Fuente. On forwarding state control in VPN multicast based on MPLS multipoint LSPs. In Proceedings of the 2012 IEEE 13th International Conference on High Performance Switching and Routing. IEEE, 2012.
- [37] A. Matrawy, W. Yi, I. Lambadaris, and C. H. Lung. MPLS-based multicast shared trees. In Proceedings of the 4th Annual Communication Networks and Services Research Conference (CNSR'06). IEEE, 2006.
- [38] F. A. Khan, F. R. Wani, and M. A. Chishti. Performance analysis of Internet Protocol version 4 (IPv4) and Internet Protocol version 6 (IPv6) over MPLS. *The International Journal of Computer and Telecommunications Networking*, 2(3):93-101, 2014.
- [39] S. Khanna, J. Naor, and D. Raz. Control message aggregation in group communication protocols. In Proceedings of the International Colloquium on Automata, Languages, and Programming, pages 135-146. Springer, 2002.
- [40] Z. Liu, W. Dou, and Y. Liu. A scheme of aggregated multicast based on tree splitting. In Proceedings of the International Conference on Research in Networking, pages 829-840. Springer, 2004.

- 
- [41] J. Moulrierac and A. Guitton. Distributed multicast tree aggregation. RR-5636, INRIA, 2005.
- [42] J. Cui, D. Maggiorini, J. Kim, K. Boussetta, and M. Gerla. A protocol to improve the state scalability of source specific multicast. In Proceedings of the Global Communications Conference. IEEE, 2002.
- [43] J. Moulrierac, A. Guitton, and M. Molnar. Hierarchical aggregation of multicast trees in large domains. *Journal of Communications (JCN)*, 1(6):33-44, 2006.
- [44] J. Moulrierac and A. Guitton. QoS scalable tree aggregation. In Proceedings of the International Conference on Research in Networking, pages 1405-1408. Springer, 2005.
- [45] N. Ali, A. Belghith, J. Moulrierac, and M. Molnar. QoS multicast aggregation under multiple additive constraints. *Computer Communications*, 31(15):3564-3578, 2008.
- [46] B.R. Badrinath and P. Sudame. Gathercast: the design and implementation of a programmable aggregation mechanism for the Internet. In Proceedings of the Ninth International Conference on Computer Communications and Networks. IEEE, 2000.
- [47] R. C. Chalmers and K. C. Almeroth. Developing a multicast metric. In Proceedings of the Global Communications Conference. IEEE, 2000.
- [48] B. A. Forouzan. *Data Communications and Networking*. McGraw-Hill Higher Education, Fourth Edition, pages 647-699, 2007.
- [49] Abilene network topology, <http://www.av.it.pt/anp/on/refnet2.html>, Last accessed on January 15, 2011.
- [50] NSFNET network topology, <http://www.av.it.pt/anp/on/refnet2.html>, Last accessed on June 18, 2009.
- [51] ARPANET network topology, <https://en.wikipedia.org/wiki/ARPANET>, Last accessed on August 25, 2018.
- [52] vBNS network topology, <http://www.av.it.pt/anp/on/refnet2.html>, Last accessed on June 18, 2009.