

This is to certify that the
dissertation entitled

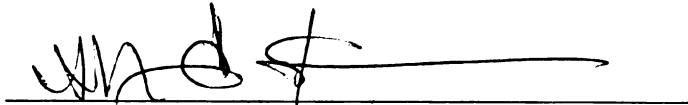
**SUPPORTING MULTICAST IN SCALABLE QOS
FRAMEWORKS**

presented by

BAIJIAN YANG

has been accepted towards fulfillment
of the requirements for the

Ph. D. degree in Computer Science



Major Professor's Signature

12/03/2002

Date

LIBRARY
Michigan State
University

PLACE IN RETURN BOX to remove this checkout from your record.
TO AVOID FINES return on or before date due.
MAY BE RECALLED with earlier due date if requested.

DATE DUE	DATE DUE	DATE DUE

ABSTRACT

SUPPORTING MULTICAST IN SCALABLE QoS FRAMEWORKS

By

Baijian Yang

SUPPORTING MULTICAST IN SCALABLE QoS FRAMEWORKS

Advances in the areas of QoS and multicasting have necessitated the need of integration of these two important Internet. Integration of multicasting and scalable QoS frameworks, such as DiffServ and MPLS, is promising since the underlying QoS support may reduce the complexity to locate a QoS-satisfied multicast tree. In this dissertation, we first identified the problems of provisioning IP multicasting in DiffServ domain, and proposed an efficient DiffServ-Aware Multicasting (DAM) scheme. Next, we have investigated the issue of supporting IP multicast with MPLS-TE. The Edge Router Multicast (ERM) scheme we proposed solved most of the perceived problems without noticeably losing the benefit of multicast. Finally, as the trend of multicast is now moving to the upper layer, we propose a tree-building procedure and Algorithm (TIA) to build a cost-efficient overlay multicast tree. The performance analysis and simulation results show that the proposed schemes provide a good balance among the conflicting requirements. The proposed approach of building multicast tree on top of DiffServ and MPLS-TE is scalable as well as practical.

A DISSERTATION

Submitted to

Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science and Engineering

Michigan State University

2002

ABSTRACT

SUPPORTING MULTICAST IN SCALABLE QoS FRAMEWORKS

By

Baijian Yang

Advances in the areas of QoS and multicasting have necessitated the need of integration of these two important features of Internet. Integration of multicasting and scalable QoS frameworks, such as DiffServ and MPLS, is promising since the underlying QoS support may reduce the complexity to locate a QoS-satisfied multicast tree. In this dissertation, we first identified the problems of provisioning IP multicasting in DiffServ domain, and proposed an efficient DiffServ-Aware Multicasting (DAM) scheme. Next, we have investigated the issues of supporting IP multicast with MPLS-TE. *To my parents* The Edge Router Multicast (ERM) scheme we proposed solved most of the perceived problems without noticeably losing the benefit of multicast. Finally, as the trend of multicast is now moving to the upper layer, we propose a tree-building protocol – Incremental Insertion Algorithm (IIA) to build a cost-efficient overlay multicast tree for DiffServ domains. The performance analysis and simulation results demonstrate that the proposed schemes provide a good balance among several performance metric. In summary, the approach of building multicast tree on top of the existing QoS frameworks is scalable as well as practical.

ACKNOWLEDGMENT

First and foremost, I am very grateful to my advisor, Dr. Prasant Mohapatra, for his continuous support and help on all aspects from technical to material. His guidance helped me overcome many technical obstacles which otherwise would have taken much more efforts. Some solutions presented in this proposal was directly motivated through discussions with him.

I am also very grateful to my co-advisor Dr. Abdol H. Estaharian, who offered me tremendous help. Special thanks to Prof. Lionel Ni for his valuable advice and inspiring comments. I would also like to express my gratitude to the other two committee members, Dr. Matt W. Mutka, and Dr. Zhenfang Zhou for their encouragement and valuable suggestions.

Many of my colleagues have **To my parents** my dissertation. I would like to thank Xiangping Chen, Fugui Wang, Jian Li, and Zhi Li for their helpful discussions.

ACKNOWLEDGMENT

First and foremost, I am very grateful to my advisor, Dr. Prasant Mohapatra, for his continuous support and help on all aspects from technical to material. His guidance helped me overcome many technical obstacles which otherwise would have taken much more efforts. Some solutions presented in this proposal was directly motivated through discussions with him.

I am also very grateful to my co-advisor Dr. Abdol H. Esfahanian, who offered me tremendous help. Special thanks to Prof. Lionel Ni for his valuable advice and inspiring comments. I would also like to express my gratitude to the other two committee members, Dr. Matt W. Mutka, and Dr. Zhenfang Zhou for their encouragement and valuable suggestions.

Many of my colleagues have contributions to my dissertation. I would like to thank Xiangping Chen, Fugui Wang, Jian Li, and Zhi Li for their helpful discussions.

1	INTRODUCTION	1
1.1	Basic Concepts	2
1.2	Multicast	2
1.2.1	Multicast	2
1.2.2	QoS Techniques	6
1.2.3	QoS Techniques	6
1.3	Multi-Protocol Label Switching (MPLS)	10
1.3.1	Multi-Protocol Label Switching (MPLS)	10
1.3.2	Research Goals	12
2	RELATED WORK	14
2.1	QoS-aware Route-based Multicast	14
2.1.1	QoS-aware Route-based Multicast	14
2.2	NFS Problems	15
2.2.1	NFS Problems	15
2.2.2	Marking Problem	18
2.3	Supporting IP Multicast in MPLS Domains	19
2.4	Supporting Overlay Multicast in DiffServ Domains	23
3	DIFFSERV AWARE MULTICASTING (DAM)	26
3.1	DiffServ and Multicast Model	26
3.2	Weighted Traffic Conditioning (WTC) Model	28
3.3	Receiver-Initiated Marking (RIM) Scheme	30
3.4	Heterogeneous DSCP Encapsulation (HDE)	32
3.5	DiffServ Aware Multicasting (DAM) technique	33
3.6	Implementation Issues	40
3.6.1	WTC	40
3.6.2	Marking	42
3.7	Performance Analysis	43
3.7.1	Qualitative Analysis	43

3.7.2	Quantitative	TABLE OF CONTENTS	44
3.7.3	Simulations Results		49
3.8	Summary		53
LIST OF TABLES			VIII
EDGE ROUTER MULTICASTING WITH MPLS-TE			53
4.1	Basics of Edge Router Multicasting in MPLS (ERM)		55
LIST OF FIGURES			IX
4.1.2	ERM Basics		57
1	INTRODUCTION		1
1.1	Motivation		1
1.2	Basic Concepts		2
4.3	1.2.1 IP Multicast		2
	1.2.2 Overlay Multicast		4
	1.2.3 QoS Techniques		6
	1.2.4 Differentiated Services (DiffServ)		9
4.4	1.2.5 Multi-Protocol Label Switching (MPLS)		10
1.3	Research Goals		12
5	SUPPORTING OVERLAY MULTICAST IN DIFFSERV DOMAINS		72
2	RELATED WORK		14
2.1	QoS-aware-route-based Multicast		14
2.2	Provisioning Multicasting in DiffServ Domains		15
5.2	2.2.1 NRS Problems		15
	2.2.2 Marking Problem		18
2.3	Supporting IP Multicast in MPLS Domains		19
2.4	Supporting Overlay Multicast in DiffServ Domains		23
5.2.3	Partition Repair		30
5.3	Simulations Results		31
3	DIFFSERV AWARE MULTICASTING (DAM)		26
3.1	DiffServ and Multicast Model		26
3.2	Weighted Traffic Conditioning (WTC) Model		28
3.3	Receiver-Initiated Marking (RIM) Scheme		30
3.4	Heterogeneous DSCP Encapsulation (HDE)		32
3.5	DiffServ Aware Multicasting (DAM) technique		33
6	3.6 Implementation Issues		40
6.1	3.6.1 WTC		40
6.2	3.6.2 Marking		42
3.7	Performance Analysis		43
3.7.1	Qualitative Analysis		43

BIBLIO	3.7.2	Quantitative Analysis	44
	3.7.3	Simulations Results	49
	3.8	Summary	53
4		EDGE ROUTER MULTICASTING WITH MPLS-TE	55
	4.1	Basics of Edge Router Multicasting in MPLS (ERM)	55
		4.1.1 Motivations	55
		4.1.2 ERM Basics	57
	4.2	Extension to ERM Routing	59
		4.2.1 Basic Characteristics	60
		4.2.2 ERM2 Illustration	61
	4.3	Performance Analysis	63
		4.3.1 Relative Tree Cost	65
		4.3.2 Link Stress	67
		4.3.3 Relative Delay	67
	4.4	Summary	69
5		SUPPORTING OVERLAY MULTICAST IN DIFFSERV DOMAINS	72
	5.1	Network Models and Design Issues	72
		5.1.1 DiffServ Architecture and Its Features	72
		5.1.2 Design Issues	73
		5.1.3 Performance Metrics	74
	5.2	Group Management and Incremental Insertion Protocol	75
		5.2.1 Group Management Overview	76
		5.2.2 Incremental Insertion Algorithm (IIA)	77
		5.2.3 Partition Repair	80
	5.3	Simulations Results	81
		5.3.1 Tree Cost	82
		5.3.2 Link Changes	83
		5.3.3 Accumulative Relative Delay Penalties (RDP)	85
	5.4	Summary	88
6		CONCLUDING REMARKS	89
	6.1	Conclusions	89
	6.2	Future Research	90

LIST OF TABLES

3.1 An Example of Multicast Flow Weight Look-up Table.	41
3.2 Notations.	45
4.1 Edge Probability of Selected Flat Random Graph Models.	64
2.2 Illustration of the Unfairness Marking Problem	20
5.1 Statistic of 100 Runs (100 member, Waxman2 mode)	82
3.1 DiffServ Multicasting Model.	26
3.2 Counting A Multicast Flow As Multiple Unicast Flows.	29
3.3 Illustration of Heterogeneous DSCP Encapsulation	34
3.4 Three Cases of Joining Multicast Tree in A DiffServ Domain.	35
3.5 Logical View of Traffic Conditioner with WTC Component.	41
3.6 Token Bucket Marking implementation at the Edge Routers	42
3.7 Performance of EF at A DiffServ Edge Router ($a=0.1, b=2$).	46
3.8 Average BB Signaling of DAM.	48
3.9 Relative BB Signaling Cost of DAM.	49
3.10 Network Topology of the Simulation.	50
3.11 EF Multicast Results	51
3.12 AF Multicast Results.	52
3.13 AF Fairness Results	53
4.1 ERMP Illustration	56
4.2 Multicast Routing Table at An Edge LSRs	56
4.3 ERMP2 Join Example	62
4.4 State Machine of ERMP2 Edge Routers	63
4.5 Relative Tree Cost, Locality Model	65
4.6 Relative Tree Cost, Waxman1 Model	65
4.7 Relative Tree Cost, Waxman2 Model	66
4.8 Link Stress, Locality Model	68
4.9 Link Stress, Waxman1 Model	68
4.10 Link Stress, Waxman2 Model	69
4.11 Relative Delay, Locality Model	70
4.12 Relative Delay, Waxman1 Model	70
4.13 Relative Delay, Waxman2 Model	71
5.1 Logical View of Tree Topology for (S,G)	77
5.2 Illustration of Leaf-join and Insertion	78
5.3 On-line Mode and Off-line Mode Repair	81

5.4	Tree Cost Comparison, LIST OF FIGURES	63
5.5	Tree Cost Comparison, Waxman1 Model	64
5.6	Tree Cost Comparison, Waxman2 Model	64
1.1	Illustration of Unicast, IP Multicast and Overlay Multicast	5
1.2	MPLS Illustration	10
2.1	Illustration of the NRS-Problem	16
2.2	Illustration of the Unfairness Marking Problem	20
3.1	DiffServ Multicasting Model	26
3.2	Counting A Multicast Flow As Multiple Unicast Flows	29
3.3	Illustration of Heterogeneous DSCP Encapsulation	34
3.4	Three Cases of Joining Multicast Tree in A DiffServ Domain	35
3.5	Logical View of Traffic Conditioner with WTC Component	41
3.6	Token Bucket Marking Implementation at the Edge Routers	42
3.7	Performance of EF at A DiffServ Edge Router ($a=0.1, b=2$)	46
3.8	Average BB Signaling of DAM	48
3.9	Relative BB Signaling Cost of DAM	49
3.10	Network Topology of the Simulation	50
3.11	EF Multicast Results	51
3.12	AF Multicast Results	52
3.13	AF Fairness Results	53
4.1	ERMP Illustration	56
4.2	Multicast Routing Table at An Edge LSRs	58
4.3	ERMP2 Join Example	62
4.4	State Machine of ERMP2 Edge Routers	63
4.5	Relative Tree Cost, Locality Model	65
4.6	Relative Tree Cost, Waxman1 Model	66
4.7	Relative Tree Cost, Waxman2 Model	66
4.8	Link Stress, Locality Model	68
4.9	Link Stress, Waxman1 Model	68
4.10	Link Stress, Waxman2 Model	69
4.11	Relative Delay, Locality Model	70
4.12	Relative Delay, Waxman1 Model	70
4.13	Relative Delay, Waxman2 Model	71
5.1	Logical View of Tree Topology for (S,G)	77
5.2	Illustration of Leaf-join and Insertion	78
5.3	On-line Mode and Off-line Mode Repair	81

5.4	Tree Cost Comparison, Locality Model	83
5.5	Tree Cost Comparison, Waxman1 Model	84
5.6	Tree Cost Comparison, Waxman2 Model	84
5.7	Link Changes, Locality Model	85
5.8	Link Changes, Waxman1 Model	86
5.9	Link Changes, Waxman2 Model	86
5.10	Relative Delay Penalties, Locality Model	87
5.11	Relative Delay Penalties, Waxman1 Model	87
5.12	Relative Delay Penalties, Waxman2 Model	88

capacity and the support for Quality of Service (QoS). Demand for extra capacity is compelled by the trend that the capacity of the current generation Internet is likely to get outgrown by the bandwidth-consuming network traffic such as transmission of continuous media, interactive games, and the evolving peer-to-peer information sharing applications. The issue of QoS support is aggressively driven by the evolving applications, and the transformation of Internet into a commercial application environment. The current version of Internet Protocol (IP) only offers 'best-effort' service. As a consequence, crucial applications find themselves contending for limited resources without any priority.

Simply increasing the network capacity through advanced technology may not be the best solution. Historically, the users have always managed to consume the entire system capacity soon after it was enlarged [1]. Multicasting techniques [2, 12, 13] are designed for networking applications working in group communication pattern. They can reduce the bandwidth consumption and relieves the network stress at the source of data by sharing network resources. Therefore, multicasting is one of the attractive solutions for the capacity shortage problem.

The notation of Quality-of-Service (QoS) has been proposed to define a col-

CHAPTER 1 INTRODUCTION

1.1 Motivation

The next generation Internet needs the support of two important aspects in addition to all the features of the current generation Internet. These aspects are: additional capacity and the support for Quality of Service (QoS). Demand for extra capacity is compelled by the trend that the capacity of the current generation Internet is likely to get outgrown by the bandwidth-consuming network traffic such as transmission of continuous media, interactive games, and the evolving peer-to-peer information sharing applications. The issue of QoS support is aggressively driven by the evolving applications, and the transformation of Internet into a commercial application environment. The current version of Internet Protocol (IP) only offers 'best-effort' service. As a consequence, crucial applications find themselves contending for limited resources without any priority.

Simply increasing the network capacity through advanced technology may not be the best solution. Historically, the users have always managed to consume the entire system capacity soon after it was enlarged [1]. Multicasting techniques [2, 12, 13] are designed for networking applications working in group communication pattern. They can reduce the bandwidth consumption and relieve the network stress at the source of data by sharing network resources. Therefore, multicasting is one of the attractive solutions for the capacity shortage problem.

The notation of Quality-of-Service (QoS) has been proposed to define a col-

lection of technologies, which allow network applications to request and receive a predictable or guaranteed service in terms of throughput capacity (bandwidth), propagation latency (delay), latency variations (jitter), loss rate, etc.. The QoS techniques do not create any additional bandwidth. Rather, they manage network resources based on the requirements of the applications and networking management policies. To enable QoS support, the cooperation of the all network layers from bottom-to-top is required in addition to all the network elements from end-to-end.

Due to their extra overheads of building and maintaining the delivery trees, multicasting techniques are most suitable for the bandwidth consuming applications which work in group communication models, such as video/audio conferences, distant education, and information dissemination. These applications usually need QoS support. Thus multicasting techniques are often associated with QoS problems. Therefore, schemes that provision QoS in multicasting techniques should be proposed and studied. In this dissertation, we are particularly interested in studying how to support multicasting in scalable QoS frameworks, such as DiffServ[14] and MPLS-TE [40].

1.2 Basic Concepts

1.2.1 IP Multicast

IP multicast is an extension to the standard IP network layer protocol [3]. It provides an efficient mechanism for one-to-many or many-to-many communication. That

is, only one copy of a multicast message will be transmitted over any link in the network, and the replication of the message will be made only where paths diverge at a router.

Fundamentally, IP multicast can be characterized as a receiver-based host group model. To be able to receive datagram from a specific multicast group, which can be identified by a unique IP address, a host must explicitly send a join message to the multicast capable routers to which it is directly attached.

From networking point of view, three issues need to be addressed to support IP multicast:

- **Multicast Address:** IP multicast uses class D Internet Protocol address, which begins with '1110' and ranges from 224.0.0.0 to 239.255.255.255. Each IP multicast group must acquire a class D IP address as the group identity before transmitting packets.
- **Host Group Management:** The native IP multicast allows a host to join/leave a multicast group freely. Internet Group Management Protocol (IGMP) was proposed to manage the multicast group memberships. One multicast router per subnet periodically sends an IGMP query message to the local hosts to get updated group memberships information.
- **Multicast Delivery Tree:** The most efficient way to forward multicast packets is to build and maintain a multicast delivery tree along the networks that multicast group members span. In IP multicast, this task is handled by the multicast routing protocols.

Based on the group density, there are two types of IP multicast routing protocols: Dense Mode (DM) and Sparse Mode (SM). DM multicast protocols include DVMRP [53], MOSPF [54] and PIM-DM [56]. In DM, multicast tree is traffic driven. The delivery tree will only be built once the multicast packets flow into the network. Due to the high group density, these types of routing protocols normally adopt flood/prune techniques to set up delivery trees. SM multicast protocols consist of CBT [55], PIM-SM [57] and etc. Unlike DM, SM multicast delivery tree is control driven. The tree structure is usually built prior to the arrival of multicast traffic by explicit joining to the Rendezvous Point (RP). Furthermore, IP multicast routing protocols can also be classified into source-based tree and group-shared tree in terms of the tree structure. The former intends to build a delivery tree for each multicast source, while the latter tempts to set up a single shared tree for a specific multicast group. In spite of the fact that group-shared tree approach scales better, it may not be applicable to DM multicast because a feasible tree may not exist when users have QoS requirements.

1.2.2 Overlay Multicast

Although it has been widely accepted that the network layer is the appropriate place to efficiently support multicast services, the deployment of IP multicast still advances in a slow pace even after a decade of efforts. A number of reasons can explain this situation. First, IP multicast requires routers to maintain per group state, which is fundamentally at odds with the conventional stateless internet infrastructure, where intelligence is pushed to the end systems at the edge of the

networks to keep core nodes simple, fast, and stateless. Due to this architectural conflict, costly upgrades are necessary to enable multicast in the routers. Second, IP multicast routing look-up entries are hard to be aggregated. Traditional unicast IP addresses not only reveal the identities of the end systems, but also imply their 'location' information from their network and sub-network IDs. However, A multi-cast IP address is only associated with a certain multicast group whose members could scatter arbitrarily and dynamically in the networks. Third, limited IP multicast address space also hinders its deployment. It suggests that IP multicast addresses must be assigned cautiously to avoid naming conflicts. Furthermore, IP multicast lacks flow control and authentication mechanisms, which further slow down its process of being commercially deployed.

To get around the inherent difficulties involved in deploying IP multicast, an alternative solution which is termed as *overlay multicasting* or *end-system multicasting* has attracted a lot of attentions. The overlay multicasting approach assumes no multicasting support in the network layer, and constructs a multicast delivery tree in the application layer. An intuitive comparison of IP multicast, multiple unicast and overlay multicast is illustrated in Figure 1.1.

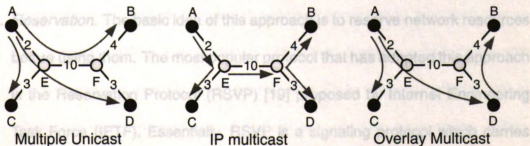


Figure 1.1: Illustration of Unicast, IP Multicast and Overlay Multicast

In Figure 1.1, host A is the sender while host B, C and D are receivers. Node E and F refer routers. For traditional unicast scheme, three identical copies of the data will be sent from A to B, C and D respectively. While using IP multicast approach, only one copy of data presents on each on-tree link, i.e. $A \rightarrow E \rightarrow C$, $E \rightarrow F \rightarrow D$, and $F \rightarrow B$. In Overlay multicast scenario, multicast tree is built at the application layer. Sender A sends two identical copies to C and D. Upon receiving, D makes a copy and forwards it to B through link $D \rightarrow F \rightarrow B$. If we ignore the network layer and the application layer overheads and let the number on each link denote link cost, then the total cost of IP multicast and overlay multicast are 22 and 27, respectively, while that of unicast is 38.

The advantages of overlay multicast includes easier deployment, better scalability and support of higher layer functionalities, such as security and congestion control. On the other hand, it is less efficient in terms of network resource usages and leads to a longer transmission delay.

1.2.3 QoS Techniques

Generally, QoS can be achieved by four methods listed below:

1. *Reservation*. The basic idea of this approach is to reserve network resources before using them. The most popular protocol that has adopted this approach is the Reservation Protocol (RSVP) [19] proposed by Internet Engineering Task Force (IETF). Essentially, RSVP is a signaling protocol which carries the bandwidth reservation along the data path predetermined by the network

routing protocol. It passes the reservation request to all network components in the traffic flow path. To meet the requirement of QoS end-to-end, each hop along the path must grant the reservation and allocate the requested bandwidth. RSVP needs to maintain the correct state information. This is achieved by periodically refreshing the reservation state among a node and its RSVP neighbors. RSVP was originally designed for multicast traffic, but it can also be used for unicast traffic. The major drawback of RSVP is its poor scalability, since the network has to maintain soft-state information for each micro-flow. Recently, RSVP has been modified and extended in several ways to mitigate the scaling problems. For instance, RSVP has been extended to reserve resources for aggregation of flows. There are also several proposals to reduce the overhead of soft-state maintenance [4]

2. *Priority.* Priority-based approach classifies the traffic into different classes with each class of traffic encountering different forwarding treatment. For instance, at each router, higher priority traffic has less queuing delay and drop ratio than lower priority traffic does. By more aggressively sacrificing the QoS of less important network flows, mission critical flows can thus be protected when network congestion occurs. Priority based approach is scalable because traffic is aggregated and classified. It often works together with dynamic queuing mechanisms like Random Early Detection (RED) [31] to manipulate queuing behaviors of each priority of traffic aggregation. Differentiated Services (DiffServ) [14] described in Section 1.2.4 is a good example

of such approach.

3. *QoS Routing.* In QoS routing, a network is usually represented as a weighted diagram $G=(V,E)$, where V denotes the set of nodes and E denotes the set of links. Each link is associated with the current status of the network. The QoS routing approach amounts to finding the best path (unicast case) or tree (multicast case) with respect to certain QoS constraints. Unlike the two previously mentioned methods which are independent of the routing protocols, QoS routing approaches will determine a feasible path. In order to better utilize network resources while satisfying the QoS requirements, the resulting path located by QoS-routing approach is normally different from the conventional shortest path. The primary concern of this approach is the computational complexity. As summarized in [18], most of the optimization problems for QoS-aware multicast routing are NP-complete.

4. *Internet Traffic Engineering.* Internet traffic engineering is concerned with the issues of performance evaluation and performance optimization of operational IP networks [5]. Unlike QoS routing mentioned above, the goal of traffic engineering is a global optimization problem rather than a per-flow optimization problem. By aggregating traffic flows, overriding the conventional routing, and performing constraints based routing, traffic engineering approach can be used to provide a globally optimized QoS scheme without losing scalability.

It should be noted that the four approaches described above are not exclusive. In practice, all or some of them can be integrated together to provide better performance by eliminating the hop-by-hop signaling and avoiding per micro-flow or per customer state maintenance within the core routers.

1.2.4 Differentiated Services (DiffServ)

1.2.5 Multi-Protocol Label Switching (MPLS)
In DiffServ model, flows entering a network are classified and conditioned at the boundaries of the network and assigned to a set of behavioral aggregates. Each traffic aggregate can be recognized by a DiffServ Code Point (DSCP), which is encoded in the packet header (such as in the TOS bytes in IPv4 header). Within the core of the network, packets are forwarded based on the per hop behaviors (PHBs) associated with the DSCP. A DiffServ domain is defined as a contiguous set of DiffServ-aware nodes with a common service provisioning policy and a set of PHB groups implemented at each node. In order to support differentiated services, Service Level Agreements (SLAs) must be set up between the DiffServ domains and their clients. SLA basically specifies a negotiated service profile between two adjacent DiffServ domains. The resource allocation is taken care of by dedicated nodes in the domain, termed as Bandwidth Brokers (BBs). Current proposal of DiffServ has two basic classes of PHBs: Assured Forwarding (AF) and Expedited Forwarding (EF). AF assigns out-of-profile (more than the SLA) traffic a high drop probability and it is used to support assured services in which the customers are likely to get the negotiated SLA without any hard guarantees. In the absence of network congestion, the nodes can use network resources beyond their pre-negotiated values. EF exercises a strict admission control and drops all out-of-

profile traffic. Since EF guarantees a minimum service rate and has the highest priority, it is used to support premium services. In summary, DiffServ facilitates scalability by eliminating the hop-by-hop signaling and avoiding per micro-flow or per customer state maintenance within the core routers.

1.2.5 Multi-Protocol Label Switching (MPLS)

The fundamental idea of Multi-Protocol Label Switching (MPLS) [36] involves assigning short, fixed length labels to the packets at the ingress point of the network. In ATM environment, the label is encoded in the VCI/VPI field. In IP network, a 32-bit 'shim' header is inserted between the network layer header and the data link layer header. When forwarding packets inside of an MPLS cloud, the MPLS capable router, termed as Label Switching Router (LSR) only examines the label rather than the IP header.

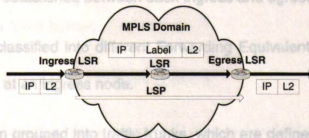


Figure 1.2: MPLS Illustration

As depicted in Figure 1.2, when a packet from a non-MPLS domain arrives at a MPLS domain, an MPLS header will be generated and inserted at the ingress LSR based on the IP header in the packet and local routing information. Within the MPLS domain, the LSR examines the incoming label, looks up the forwarding

table, and replaces it with an outgoing label. Then the packet is switched to the next LSR. Before a packet leaves the MPLS domain, the header is removed. The path between the ingress LSR and egress LSR is called Label Switching Path (LSP), which can be set up using Label Distribution Protocol (LDP) [37], or RSVP [38] etc.

MPLS enriches the classical routing functionality by separating the forwarding components and path controlling components. It allows packets to be forwarded along a pre-configured LSP other than the conventional shortest path, thus provides a means for traffic engineering (MPLS-TE) [40]. Adopting online or off-line optimization algorithms, MPLS-TE can maximize operational network performance and balance traffic load. Moreover, working together with RSVP or DiffServ, MPLS-TE also provides a scalable QoS scheme. Typically, the procedures of MPLS-TE can be described as following:

- LSPs are pre-established between each ingress and egress node pair.
- Packets are classified into different Forwarding Equivalent Classes (FECs) when arriving at an ingress node.
- FECs are then grouped into traffic trunks, which are defined as routable objects placed inside of an LSP [42].
- Finally, traffic trunks are mapped to LSPs which can satisfy their QoS requirements with optimized network performance.

Two primary problems of MPLS-TE are layout design and flow assignment. It would be efficient to run off-line algorithms if we had a priori knowledge about traffic

demands and patterns. But such assumption is not valid in practice. Some online algorithms have been proposed to address LSPs layouts and flow assignments for unicast traffic [41, 43, 44]. However, Up till now, no algorithms have been published for traffic engineering multicast flows in MPLS domain.

1.3 Research Goals

Provisioning QoS and multicast services are both challenging yet promising problems. In reality, the primary concerns of each technique are scalability and ease of implementation. When combining both of them and designing integrated schemes to provide QoS support for multicasting, our goal is to propose scalable, flexible and practical solutions to avoid further complicating the problems. The strategy we take is to support multicast service on top of the existing scalable QoS frameworks, such as DiffServ and MPLS Traffic Engineering. Specifically, the research goals of this dissertation are listed below:

- Provisioning IP multicast in DiffServ domains.
- Providing QoS solutions for supporting IP multicast by MPLS traffic engineering.
- Supporting overlay multicast in DiffServ domains.

The rest of dissertation is organized as follows. In Chapter 2, problems and related works in this area will be presented. Proposed solutions are illustrated and analyzed from Chapter 3 through Chapter 5, with each chapter focusing on one

sub-problem as itemized above. Finally, the concluding remarks are presented in Chapter 6.

2.1 QoS-aware-route-based Multicast

Conventional IP multicast routing protocols like PIM-SM try to build the delivery tree solely based on network topology and group member distribution. While QoS requirements are not considered and traditional IP networks can only provide 'best-effort' service, such multicast routing protocols can only end up with schemes that are QoS incapable.

A number of protocols had been proposed [7, 8] to establish a multicast tree which satisfies the QoS constraints, and is optimized with minimum cost. However, such approaches are not practical in the internet environment. First, they incur high computational overhead. Second, they require maintenance of the global network state. Finally, they cannot handle dynamic multicast group membership. The Spanning Join Protocol [9] proposed later overcomes the previously listed drawbacks, but it relies on flooding to find a feasible tree branch. Thus Spanning Join Protocol will pose excessive communication overhead on the network. QoSMIC [16] and OMRP [17] are two major modifications to the Spanning Join Protocol. By using parallel searching, either local/global or multiple paths, they limited the message overhead while maximizing the chances to locate a feasible multicast branch.

Although these schemes can find efficient QoS-aware routes for multicasting, they are not sufficient for provisioning QoS. The path search, if succeeds, can only guarantee that the resulting path meets the QoS requirements when there is no

CHAPTER 2 RELATED WORK

2.1 QoS-aware-route-based Multicast

Conventional IP multicast routing protocols like PIM-SM try to build the delivery tree solely based on network topology and group member distribution. While QoS requirements are not considered and traditional IP networks can only provide 'best-effort' service, such multicast routing protocols can only end up with schemes that are QoS incapable.

A number of protocols had been proposed [7, 8] to establish a multicast tree which satisfies the QoS constraints, and is optimized with minimum cost. However, such approaches are not practical in the internet environment. First, they incur high computational overhead. Second, they require maintenance of the global network state. Finally, they cannot handle dynamic multicast group membership. The Spanning Join Protocol [9] proposed later overcomes the previously listed drawbacks, but it relies on flooding to find a feasible tree branch. Thus Spanning Join Protocol will pose excessive communication overhead on the network. QoSMIC [16] and QMRP [17] are two major modifications to the Spanning Join Protocol. By using parallel searching, either local/global or multiple paths, they limited the message overhead while maximizing the chances to locate a feasible multicast branch.

Although these schemes can find efficient QoS-aware routes for multicasting, they are not sufficient for provisioning QoS. The path search, if succeeds, can only guarantee that the resulting path meets the QoS requirements when there is no

congestion. QoS services such as RSVP must be supported to maintain this QoS satisfied path. Further, lack of global QoS support makes QoS-aware-route-based schemes inefficient.

Consider an example where a host makes a request with a QoS requirement $q1$. Assuming that a path is found and the host joins the multicast tree at node n . If the upstream links of node n can only provide service lower than $q1$, then an end-to-end QoS requirement at level $q1$ cannot be guaranteed and the path searching will turn out to be wasted.

2.2 Provisioning Multicasting in DiffServ Domains

The strength of this scheme is that it separates QoS issues from multicast routing. That is, multicast delivery trees are built and maintained by normal multicast protocol without going through a complicated QoS path searching process. The QoS is provided by the underlying DiffServ architecture. Since DiffServ is a scalable approach, it is desirable to incorporate it with multicast to provide QoS enabled multicast.

2.2.1 NRS Problems

In the context of DiffServ, network resources are consumed based on the pre-negotiated SLA. However, in DiffServ-aware multicasting environment, it is possible that the actual resources consumed exceed the pre-negotiated SLA. This problem is denoted as Neglected Reserved Sub-tree (NRS) problem [21]. It vio-

lates the SLAs and adversely affects any existing traffic flows. Basically, there are two types of NRS problems:

Case 1: As shown in Figure 2.1(a), a new node R2 joins the multicast group at CR1. The branching point is the interior core node CR1 in DiffServ domain A. Since traffic meters are normally not available in core routers of a DiffServ domain, the extra

traffic in DiffServ domain A will consume the resources that was subscribed. In other words, the extra multicast traffic is "stealing" bandwidth from lower service levels on the output link.

NRS problem can be solved by assigning a Lower than Best Effort (LBE) PHB to the newly branched multicast traffic [22]. In this approach, the resources and processing of existing traffic are not affected by the multiplicity of the

DiffServ model. In order to get higher level of services, the joining node has to explicitly negotiate with the BBs. Upon succeeding, the BBs will reconfigure the

Case 2: Figure 2.1(b) shows a multicast tree originating from source S and destined to R1 and R2. The branching point is at the egress node ER3 of DiffServ domain A. Assume that the existing multicast flow has subscribed the service level L. The border routers are equipped with meters and they normally do traffic conditioning to ensure that the traffic going to the downstream DiffServ domain B conform to the SLA between two domains. In the case where R2 joins the multicast group at ER1, the extra traffic generated between ER3 and ER1 by this new multicast traffic for R2 may exceed the service L that domain B has subscribed from

Figure 2.1: Illustration of the NRS-Problem.

Case 1: Figure 2.1(a) shows a multicast tree originating from source S and destined to R1 and R2. The branching point is at the egress node ER3 of DiffServ domain A. Assume that the existing multicast flow has subscribed the service level L. The border routers are equipped with meters and they normally do traffic conditioning to ensure that the traffic going to the downstream DiffServ domain B conform to the SLA between two domains. In the case where R2 joins the multicast group at ER1, the extra traffic generated between ER3 and ER1 by this new multicast traffic for R2 may exceed the service L that domain B has subscribed from

Case 2: Figure 2.1(b) shows a multicast tree originating from source S and destined to R1 and R2. The branching point is at the egress node ER3 of DiffServ domain A. Assume that the existing multicast flow has subscribed the service level L. The border routers are equipped with meters and they normally do traffic conditioning to ensure that the traffic going to the downstream DiffServ domain B conform to the SLA between two domains. In the case where R2 joins the multicast group at ER1, the extra traffic generated between ER3 and ER1 by this new multicast traffic for R2 may exceed the service L that domain B has subscribed from

Case 3: Figure 2.1(c) shows a multicast tree originating from source S and destined to R1 and R2. The branching point is at the egress node ER3 of DiffServ domain A. Assume that the existing multicast flow has subscribed the service level L. The border routers are equipped with meters and they normally do traffic conditioning to ensure that the traffic going to the downstream DiffServ domain B conform to the SLA between two domains. In the case where R2 joins the multicast group at ER1, the extra traffic generated between ER3 and ER1 by this new multicast traffic for R2 may exceed the service L that domain B has subscribed from

Case 4: Figure 2.1(d) shows a multicast tree originating from source S and destined to R1 and R2. The branching point is at the egress node ER3 of DiffServ domain A. Assume that the existing multicast flow has subscribed the service level L. The border routers are equipped with meters and they normally do traffic conditioning to ensure that the traffic going to the downstream DiffServ domain B conform to the SLA between two domains. In the case where R2 joins the multicast group at ER1, the extra traffic generated between ER3 and ER1 by this new multicast traffic for R2 may exceed the service L that domain B has subscribed from

Case 5: Figure 2.1(e) shows a multicast tree originating from source S and destined to R1 and R2. The branching point is at the egress node ER3 of DiffServ domain A. Assume that the existing multicast flow has subscribed the service level L. The border routers are equipped with meters and they normally do traffic conditioning to ensure that the traffic going to the downstream DiffServ domain B conform to the SLA between two domains. In the case where R2 joins the multicast group at ER1, the extra traffic generated between ER3 and ER1 by this new multicast traffic for R2 may exceed the service L that domain B has subscribed from

domain A. Thus if the SLA is not renegotiated, the over-subscribed packets with service L will be dropped randomly without discriminating between the flows. The consequence is that both unicast and multicast traffic with service L are adversely affected.

Case 2: As shown in Figure 2.1(b), R2 joins the multicast group at CR1. The branching point is the interior core node CR1 in DiffServ domain A. Since traffic meters are normally not available in core routers of a DiffServ domain, the extra traffic in DiffServ domain A will consume more than that was subscribed. In other words, the extra multicast traffic may 'steal' the traffic quota from lower service levels on the output link.

NRS problem can be solved by assigning a Lower than Best Effort (LBE) PHB to the newly branched multicast traffic [22]. In this approach, the resources and processing of existing traffic are protected while maintaining the simplicity of the DiffServ model. In order to get higher level of services, the joining node has to explicitly negotiate with the BBs. Upon succeeding, the BBs will reconfigure the routers accordingly.

In the LBE approach, the network management entities do not have correct information about the amount of traffic an individual DiffServ domain has served because the traffic counters are located only at the ingress routers. That is, a multicast flow with service better than best-effort will be counted only once even if it has been replicated and branched multiple times within a domain. This scheme, therefore, is not attractive to ISPs. Moreover, every receiver with QoS requirements can only get the lowest level of services even though the available network resources

are sufficient. This of course is not desirable for the users.

2.2.2 Marking Problem

In the DiffServ network, packets are marked as 'In' or 'Out' based on the profile negotiated through the SLA. In unicast communication, the marking of the packets is usually done on the aggregate basis of bandwidth requirements. As DiffServ is uni-directional, the current marking scheme is normally sender-based, which do not consider the QoS requirements of the receivers. However, such a marking would not be adequate in multicast communications. Packet marking in DiffServ multicasting environment differs from that of the unicast case in the following three aspects:

1. IP multicast works in a group communication mode and receivers in a multicast group may have different QoS requirements. When multicast packets are duplicated in a router, every outgoing branch may be marked differently. It also implies that the marking should be based on the requirements and the capabilities of the receivers.
2. Group membership in multicast operation is dynamic. When a new host joins a multicast group, a new branch may be generated. From the discussions in Section 2.2.1, simply coping the DSCP code from existing branch may lead to SLA violations.
3. When heterogeneous marking are allowed in a DiffServ domain, statically marking the lower level of service subtree after admission control will bring

in the issue of unfairness between multicast flows and unicast flows. Consider the scenario in Figure 2.2, a multicast flow enters a DiffServ Domain at ingress router E1, and is destined to E3 and E2 requesting *EF* and *AF1* respectively. Assuming that at the incoming interface of E1, 80 percent of the *AF1* packets are in profile and marked with lower dropping probability *DSCP AF11*, while the remaining *AF1* packets are marked with higher dropping probability *DSCP AF12*. Further assume that network congestion only happens on the link from core router CR to edge router E2. Then only keeping a static *AF11* DSCP in the multicast branching node CR is not sufficient, because the new subtree originates from the core router is escaped from traffic conditioning. In this example, unicast *AF1* class packets traveling from E1 to E2 will be dropped more severely than that of multicast packets, since 20 percent of the unicast *AF1* packets are marked with higher dropping probability *AF12* but all the *AF1* multicast packets are granted *AF11* DSCP after admission control. It depicted that unfairness exists between multicast flows and unicast flows if the marking mechanism is not handled properly.

In Chapter 3, we have proposed a fair marking scheme which accommodates heterogeneous QoS requirements of the receivers without violating the SLAs.

2.3 Supporting IP Multicast in MPLS Domains

MPLS is standardized by IETF and is expected to be implemented in the near future. It is thus important to address the issues of supporting IP multicast in MPLS

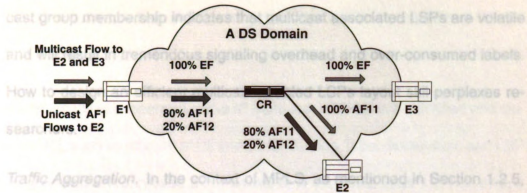


Figure 2.2: Illustration of the Unfairness Marking Problem.

domains. Furthermore, MPLS traffic engineering has the potential to provide QoS for IP multicast.

While MPLS offers a great flexibility in packet forwarding, it does not enrich the functionality of native IP multicast routing. On the contrary, problems arise when layer 3 multicast trees are mapped onto layer 2 LSPs. Thus a number of issues need to be addressed, such as flood and prune, source/shared trees, uni/bi-directional trees, and encapsulated multicast [45]. Specifically, when we leverage the power of MPLS traffic engineering to support QoS-aware multicasting, we may encounter a series of difficulties generalized as follows.

- LSP design.** The multicast tree structure requires establishing point-to-multipoint LSPs or even multipoint-to-multipoint LSPs. In current MPLS architecture, only point-to-point LSP has been addressed. MPLS does not exclude other type of LSPs, but no mechanism has been standardized for this purpose. In fact, to the author's knowledge, so far only multipoint-to-point LSP [47] has been studied and proposed to save label space. Moreover, dynamic multi-

cast group membership indicates that multicast associated LSPs are volatile and will result in tremendous signaling overhead and over-consumed labels.

How to design an efficient multicast-enabled LSPs layout still perplexes researchers.

General issues of supporting native IP multicast in MPLS are identified and discussed in [45], such as mixture of L2 and L3 forwarding, label distribution, and LSP

- **Traffic Aggregation.** In the context of MPLS, as mentioned in Section 1.2.5, traffic is aggregated and mapped to LSPs at the entrance of the network to achieve scalability. This feature will hardly be applied to multicast traffic. To handle this situation, one needs to invent algorithms, which can aggregate unicast flows with multicast flows as well as aggregate multiple multicast flows. Unfortunately, current studies on the aggregatability of multicast [48] are limited to the forwarding state of each router rather than an LSP consisting of an order of routers/switches.

- **Coexistence of Layer 2 and Layer 3 forwarding in core LSRs.** There are two situations where layer 2 incoming labels alone cannot determine the outgoing labels. One is due to switch-over from a shared tree to a source based tree. In this situation, it might happen that certain on-tree routers are on both trees and have both forwarding state (*,G) and (S,G) for the same destination address G. The other situation is when labels are not assigned appropriately. Supposing a multicast flow is mapped to the same label as some unicast flows, then at the branching node of the multicast tree, the label will be split. In both of the cases, it mandates that such LSRs to examine the layer 3 header as well as the layer 2 label. This requirement is at odds with current

of MPLS standard, where it only demands edge LSRs be capable of layer 3 and forwarding.

General issues of supporting native IP multicast in MPLS are identified and discussed in [45], such as mixture of L2 and L3 forwarding, label distribution, and LSP setup trigger mode. It proposed a framework of IP multicast in MPLS. However, it did not address issues with regard to traffic engineering multicast in MPLS domain, among active member hosts. While end-host multicasting offers an easy and general implementation of multipoint communication, it has limitations in terms of scalability. The proposed Edge Router Multicasting (ERM) scheme, which is described in Section 4.1, eliminates most of the problems mentioned in [45]. Logically, supporting ERM in MPLS can be conceived as label switching multiple simultaneous unicast flows. Problems of traffic aggregation and label assignment can thus be reduced to the problems of unicast cases.

An MPLS Multicast Tree (MMT) scheme was introduced in [46] to remove multicast forwarding state in non-branching nodes by dynamically setting up LSP tunnels between upstream branching nodes and downstream branching nodes. Like ERM, MMT can dramatically reduce forwarding states. However, MMT still needs to set up and update LSPs between edge LSRs and core LSRs (if some core LSRs are branching nodes of multicast trees). The consequence is that, core LSRs have to support coexisting L2/L3 forwarding schemes. Note that, normally LSPs are built between edge LSRs. LSPs produced by MMT may not necessarily be able to aggregate with other unicast LSPs. Whereas in ERM, it is not necessary to set up any LSPs between edge LSRs and core LSRs. Thus ERM can make multicast traf-

fic completely aggregatable with unicast traffic. Another difference between MMT and ERM is that multicast tree is centrally calculated in MMT, while basic ERM is fully distributed, and the extension to ERM is partially distributed. Some end-host based multicasting approaches, such as [49, 33], can also avoid the problems described in Section 1.2.5. Instead of building a multicast tree on the network layer, [49, 33] set up a shared tree/mesh on the application layer only among active member hosts. While end-host multicasting offers an easy and general implementation of multipoint communication, it has limitations in terms of scalability and QoS support due to a complicated group management and the absence of network layer support. In contrast, the proposed ERM model is an alternative network layer multicasting which is designed to provide QoS with MPLS traffic engineering.

2.4 Supporting Overlay Multicast in DiffServ Domains

Provisioning QoS in overlay multicasting is helpful for several reasons. First, typical multicast applications, such as video or audio conferencing, distant learning, require multimedia data stream transportation, and have certain bandwidth, delay or jitter requirements. Second, we argue that QoS provisioning in overlay multicasting is not merely an added-on service which only improves the quality of multicasting traffic. Rather, it helps to maintain a more stable overlay multicast delivery tree. For example, due to the dynamic changes of network conditions and group membership, previously proposed self-organized overlay multicast protocols [61, 64] use

adaptive mechanisms to re-construct delivery trees by periodically evaluating the cost, delay, and available bandwidth of the links among active end hosts. If the underlying networks have certain QoS support to guarantee or assure traffic on every on-tree multicast link, then such an overlay tree structure will be much less sensitive to the changes in network conditions. Ideally, with the help of network layer QoS support, overlay multicast trees only need to change when there are changes in group membership or node/link failures. In this context, many bandwidth or delay probing packets could be eliminated and the maintenance of overlay trees could be simplified.

Two fundamental solutions for supporting network layer QoS are Integrated Services (IntServ) [58] and Differentiated Services (DiffServ) [59]. In this dissertation, we focus on DiffServ architecture because of its scalability and ease of deployment. However, the proposed tree building algorithm could also be applied to IntServ.

Narada [61]. Narada is an end system multicast scheme proposed in CMU. It is an elegant practical self-organized mesh first approach. Narada can be considered as a general solution for providing application layer multicasting. Since we assume guaranteed services at the network layer, Narada does not perform very well in such situation, as indicated in the simulation results in Section 5.3.

Switch Tree Protocol[65]. It is a family of tree first protocols, whose switch-any approach bears some similarities with our on-line mode Incremental Join Protocol. The major difference is that it is a self-organizing approach which takes a longer time to produce a stable tree structure and needs loop avoidance mechanism.

Overcast[64]. The Overcast approach is optimized for bandwidth. The Up/Down

self-organizing protocol they proposed is scalable and has been commercialized. The limitation of their approach is that it does not apply to the multicast applications that have time constraints such as video/audio conferencing. It should be noted that when network resource (bandwidth) are sufficient, overcast would yield a concatenated unicast tree.

3.1 DiffServ and Multicast Model

Figure 3.1 shows two DiffServ domains A and B, serving a multicast message originating from node S and destined to nodes R1, R2 and R3. The assumptions for the network are as follows:

- Each core router is directly connected with edge routers or other core routers.
- The core routers can be multicast capable or incapable. In this dissertation, we only focus on the case where core routers are capable of multicasting. However, the framework proposed in this dissertation can be easily extended

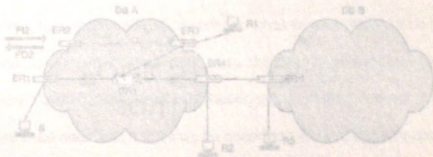


Figure 3.1: DiffServ Multicasting Model.

CHAPTER 3 DIFFSERV AWARE MULTICASTING (DAM)

Each domain is both multicast and DiffServ capable. That is, we do not consider the case in which intermediate domains are incapable of either multicast or DiffServ. Later we indicate how our scheme can be adapted for domains that are not DiffServ capable.

For core-based multicast protocol, the root of the tree (core in CBT, RP in

3.1 DiffServ and Multicast Model

Figure 3.1 shows two DiffServ domains A and B, serving a multicast message originating from node S and destined to nodes R1, R2 and R3. The assumptions for the network are as follows:

- Each core router is directly connected with edge routers or other core routers.

The core routers can be multicast capable or incapable. In this dissertation, we only focus on the case where core routers are capable of multicasting.

However, the framework proposed in this dissertation can be easily extended

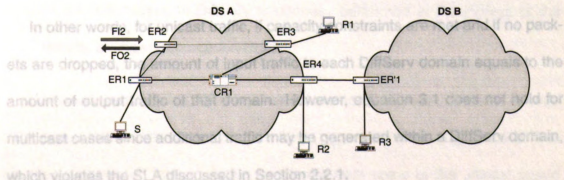


Figure 3.1: DiffServ Multicasting Model.

We propose a DiffServ-Aware Multicasting (DAM) technique, which is composed of three novel components: Weighted Traffic Conditioning (WTC) model,

to the situations where not all core routers are multicast capable.

- Each domain is both multicast and DiffServ capable. That is, we do not consider the case in which intermediate domains are incapable of either multicast or DiffServ. Later we indicate how our scheme can be adapted for domains that are not DiffServ capable.
- For core-based multicast protocol, the root of the tree (core in CBT, RP in PIM) is either located at the edge of the DiffServ Domain or functioned as edge routers.

3.2 Weighted Traffic Conditioning (WTC) Model

Given the network model as described above, for each edge router, the total amount of traffic that flows into domain i is denoted as FI_i , and total amount of traffic leaving domain i is denoted as FO_i . The capacity of upstream links and downstream links of router i are denoted as CI_i and CO_i respectively. Using the flow conservation property, the unicast flow equation of this model is:

$$\sum FI_i = \sum FO_i, \text{ where } FI_i \leq CO_i, FI_i \leq CI_i \quad (3.1)$$

In other words, for unicast traffic, if capacity constraints are met and if no packets are dropped, the amount of input traffic to each DiffServ domain equals to the amount of output traffic of that domain. However, equation 3.1 does not hold for multicast cases since additional traffic may be generated within a DiffServ domain, which violates the SLA discussed in Section 2.2.1.

We propose a DiffServ-Aware Multicasting (DAM) technique, which is composed of three novel components: Weighted Traffic Conditioning (WTC) model,

Receiver-Initiated Marking (RIM) scheme, and Heterogeneous DSCP Encapsulation (HDE). In the next four sections, we outline the components and the algorithm for DAM in detail. The WTC model aims to maintain the negotiated SLAs in DiffServ multicasting environment, and the RIM scheme is proposed primarily to accommodate heterogeneous QoS requirements of the receivers in multicast groups, while HDE provides a means to ensure the fairness among multicast flows and non-multicast flows.

3.2 Weighted Traffic Conditioning (WTC) Model

To motivate the proposed model, we first itemize the causes of the NRS problem.

1. Equation 3.1 is not satisfied in DiffServ multicasting environment.

2. BBs are unaware of the multicast traffic replications.

3. By default, the DSCP as well as data will be copied at the branching point of the multicast delivery tree.

4. Traffic conditioning in DiffServ is normally performed at the ingress of the domain on a per-aggregation basis.

The fundamental idea of WTC is to **count** the admitted multicast traffic as multiple unicast traffic while conditioning the traffic aggregate at the edge routers. This approach (discussed later in detail) is different from some of the unicast based multicasting schemes, such as [23], where one multicast flow is replaced by multiple unicast flows and each intermediate multicast receiver acts as a proxy server

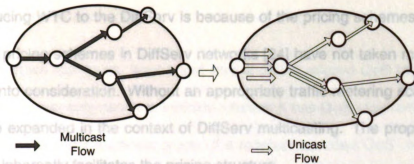


Figure 3.2: Counting A Multicast Flow As Multiple Unicast Flows.

by sending multiple unicast flows to its downstream receivers. In our approach, however, it is not necessary to convert a multicast flow into multiple unicast flows (meanwhile it can also be applied to the unicast based multicast techniques). The counting is done only on a logical basis. Thus, the WTC model retains the bandwidth saving feature of multicasting.

The WTC scheme can be illustrated by the example shown in Figure 3.2. If one multicast flow with Premium Service enters domain A and is replicated twice within this domain, the amount of that flow should be counted three times as much as the original amount at the boundary of the domain. With this approach, there will not be any SLA violations since the amount of traffic counted at the ingress point of a DiffServ domain equals to the actual amount of traffic flowing out of that domain. It must also be noted here that the proposed counting approach overestimates the bandwidth requirements within each domain for each of the multicast flows. However, such overestimation helps maintaining the SLAs.

The major goal of WTC is to keep the integrity of Equation 3.1 while it conforms to the fundamental idea of DiffServ and IP multicasting. Another reason

for introducing WTC to the DiffServ is because of the pricing schemes. Currently proposed pricing schemes in DiffServ networks [24] have not taken multicast duplication into consideration. Without an appropriate traffic metering scheme, they cannot be expanded in the context of DiffServ multicasting. The proposed WTC approach inherently facilitates the pricing structure.

3.3 Receiver-Initiated Marking (RIM) Scheme

Inherently QoS-aware multicasting is a receiver-based approach. Receivers join and leave at their own will and many have heterogeneous QoS needs. In the context of DiffServ-aware multicasting, packets should be marked according to receivers' QoS requirements. This concept is quite different from the popular DiffServ model which is unidirectional (usually sender-based) in nature. The proposed RIM scheme, on the other hand, allows receivers to initiate marking process. As we do not want to lose the scalability of DiffServ framework, we first classify QoS requirements of each new receiver into four levels, as enumerated below.

1. It has no QoS requirements.
2. It requests for whatever is the highest level of the available QoS at the node where the new member joins.
3. It explicitly specifies a QoS requirement that is lower than or equals to the highest available QoS at the node where it joins.
4. It explicitly specifies a QoS requirement that is higher than the available QoS

at the node where it joins. path toward the root of the multicast tree until an on-tree node having a DSCP equal to or higher than the requested QoS. Among these four types, level 1 and level 2 define relative QoS requirements, i.e., a new receiver only needs to indicate whether it has QoS requirements when it seeks to join a multicast group. If a receiver specifies QoS requirements explicitly, it indicates that the receiver wants absolute QoS requirements, which can be further classified as level 3 and level 4. Different levels of QoS requirements demand the packet marking scheme to appropriately handle them. To meet end-to-end QoS requirements, packet marking should be done in a consistent manner. In other words, a multicast sub-tree should be grafted at a node where its upper stream is marked at a level equal to or higher than the markings of the sub-tree.

The basic rules of this RIM scheme are described as follows, where DSCP 'DEFAULT' can be either BE or LBE.

3.4 Heterogeneous DSCP Encapsulation (HDE)

- **Level 1:** Mark the new branch as DEFAULT.

To solve the unfairness problem described in Section 2.2.2, we propose to insert

- **Level 2:** If the highest available QoS is DEFAULT, do the same as in the edge marking information in the packet header. In HDE, when a multicast flow enters a DiffServ domain and is supposed to be branched with heterogeneous QoS requirements at a core router, the markings for each of these branches are encapsulated in the packet header at the ingress router of the domain. Thus the traffic

- **Level 3:** Signal network management entities for admission control. If successful, update the WTC look-up table, and mark the new branch with a DSCP that corresponds to the best available QoS, otherwise, mark it as DEFAULT.

• **Level 4:** Traverse retracing path toward the root of the multicast tree until an on-tree node having a DSCP equal to or higher than the requested QoS requirement is found. Signal network management entities for admission control. If successful, mark the new branch with a DSCP that corresponds to the best available QoS and remark intermediate path with this new DSCP. If unsuccessful, either try to select a new path or simply mark it as DEFAULT.

The admission control mentioned above includes authentication, authorization and allocation. Since there are a number schemes proposed for SLAs renegotiation and the BB implementation [25, 30, 32], the actual resource allocation schemes could be different. The RIM and WTC we proposed are independent of service, and the actual DSCP code should be either copied or calculated from the information stored in the header. In our example, the branch from CR to E3

3.4 Heterogeneous DSCP Encapsulation (HDE)

To solve the unfairness problem described in Section 2.2.2, we propose to insert edge marking information in the packet header. In HDE, when a multicast flow enters a DiffServ domain and is supposed to be branched with heterogeneous QoS requirements at a core router, the markings for each of these branches are encapsulated in the packet header at the ingress router of the domain. Thus the traffic conditioning done at the edge routers will be equally applicable to all the multicast

The proposed DAM technique is not a simply combination of every component. Instead, it is an optimization of three components. For example, as we can see from Section 3.2 that the WTC scheme demands edge routers to maintain that of any existing unicast message. As the number of branchings within any Diff-

Serv domain is not expected to be too big (or a limit could be imposed), the HDE scheme will not pose significant overheads in terms of the header length. Furthermore, we need to capture the markings of only the heterogeneous AF traffic. The out-of-profile EF traffic will get dropped at the ingress router.

Consider Figure 3.3 as an example. A multicast flow enters a DiffServ domain at edge router E1. One branch leaves through E4 is marked as *EF*, another branch that flows out from E3 is marked as *AF1*, and the last branch exiting from E2 is marked as *AF2*. Suppose at E1, this flow is marked *AF12* for *AF1* class, and *AF21* for *AF2* class. This information will be inserted in the packet. When this packet is duplicated at the branching node CR, the DSCP stored in CR indicates the class of service, and the actual DSCP code should be either copied or calculated from the information stored in the header. In our example, the branch from CR to E3 belongs to class *AF1*, and ingress *AF1* marking for this packet is *AF12*, thus this branch will be marked as *AF12*. For the same reason, the branch from CR to E2 will be marked as *AF21*.

3.5 DiffServ Aware Multicasting (DAM) technique

In this section, we introduce our solution which embraces the three components presented in Section 3.2, Section 3.3, and Section 3.4.

The proposed DAM technique is not a simply combination of every component. Instead, it is an optimization of three components. For example, as we can see from Section 3.2 that the WTC scheme demands edge routers to maintain

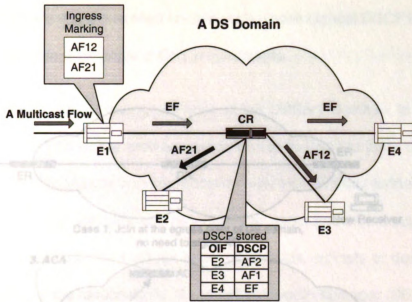


Figure 3.3: Illustration of Heterogeneous DSCP Encapsulation

and update flow-specific information. The load of updating WTC look-up table is reduced in DAM by taking receivers' QoS requirements into account. In DAM, receivers' QoS requirements will be piggybacked on the multicast JOIN packet. If a receiver requests no QoS requirements at all or the network fails to allocate the requested resources, the new branch will be marked as DEFAULT. Under such circumstances, it is not necessary to update the WTC look-up table. The weighted multicast flow traffic conditioning is required only when the new branch needs to be marked higher than the DEFAULT. So for the rest of this section, we only consider QoS requirements at levels 2, 3 and 4 (as defined earlier in Section 3.3).

When a receiver wants to join a multicast group, the existing multicast delivery tree may or may not exist in its DiffServ domain. There may be three possibilities that need to be taken care of, as shown in Figure 3.4. The 'join router' in the

discussion refers to the nearest on-tree node whose highest DSCP is either higher than or equal to the receivers' CoS requirements.

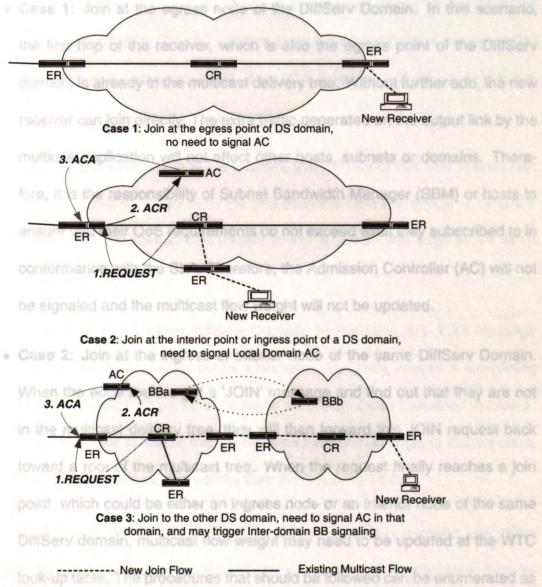


Figure 3.4: Three Cases of Joining Multicast Tree in A DiffServ Domain.

1. Mark the new branch as DEFAULT. If the branching router is a core router, it sends a REQUEST message to the upstream ER in its domain.

discussion refers to the nearest on-tree node whose highest DSCP is either higher than or equal to the receivers' QoS requirements.

- **Case 1:** Join at the egress node of the DiffServ Domain. In this scenario, the first hop of the receiver, which is also the egress point of the DiffServ domain, is already in the multicast delivery tree. Without further ado, the new receiver can join directly. The extra traffic generated on the output link by the multicast replication will not affect other hosts, subnets or domains. Therefore, it is the responsibility of Subnet Bandwidth Manager (SBM) or hosts to ensure that their QoS requirements do not exceed what they subscribed to in conformance with the SLA. Therefore, the Admission Controller (AC) will not be signaled and the multicast flow weight will not be updated.

- **Case 2:** Join at the ingress or interior node of the same DiffServ Domain. When the edge routers get a 'JOIN' message and find out that they are not in the multicast delivery tree, they will then forward this JOIN request back toward a root of the multicast tree. When the request finally reaches a join point, which could be either an ingress node or an interior node of the same DiffServ domain, multicast flow weight may need to be updated at the WTC look-up table. The procedures that should be followed can be enumerated as follows:

1. Mark the new branch as DEFAULT. If the branching router is a core router, it sends a REQUEST message to the upstream ER in its domain.

2. ER sends an Admission Control Request (ACR) [25] to the BB similar to the case when a unicast flow wants to send packets to this DiffServ domain.
3. Upon receiving ACR, the AC validates the request based on the SLA and resource availability. The AC will then send Admission Control Answer (ACA) message to the requesting ER. The ACR message will be positive if it is successful, otherwise it will be negative.
4. If the response from the AC is positive, the ER marks this new branch with the DSCP that corresponds to the best available QoS and sends an UPDATE message to the downstream routers in the path from this edge router down to the new receiver.

• **Case 3: Join at another DiffServ Domain.** In this case, the JOIN message will be forwarded to other DiffServ domains. Basically, routers in the joining DiffServ domain will perform the same actions as those described in case 2. Only difference is that all the downstream inter-domain ingress routers along the new path should also update their WTC look-up table.

For the cases of 'LEAVE' or 'PRUNE', similar strategy is adopted with minor differences, such as decreasing the counter or removing the entry rather than increasing or generating the corresponding elements.

The DAM algorithms of multicast 'JOIN' with QoS requirements are formalized in Algorithms 1 and 2. And DSCP field generation for multiple outgoing interfaces with heterogeneous QoS requirements is described in Algorithm 3.

Algorithm 1 DAM at Core Router i

```
if received_DAMmessage = JOIN( $r_{ip}, qos, G$ ) then  
  if  $i$  is on-tree node of  $G$  then  
     $dscp \leftarrow DEFAULT$   
    create a routing entry  $R(in, out, G, dscp)$   
    forward REQUEST( $r_{ip}, qos, G$ ) to upstream  
  else  
    send JOIN( $r_{ip}, qos, G$ ) to upstream  
  end if  
  
  else if received_DAMmessage = REQUEST( $r_{ip}, qos, G$ ) then  
    reverse forward REQUEST( $r_{ip}, qos, G$ )  
  
  else if received_DAMmessage = UPDATE( $r_{ip}, newdscp, G$ ) then  
     $R.dscp \leftarrow newdscp$   
    unicast UPDATE( $r_{ip}, newdscp, G$ ) to downstream node  
  
  else  
    discard received_DAMmessage  
  end if
```

As shown in Algorithm 1, core routers only perform simple tasks like setting up new routing entry and passing messages. This design conforms to the basic concept of DiffServ architecture since it keeps core routers simple and fast. The complexity of DAM is pushed to the ERs as illustrated in Algorithm 2. The major tasks of ERs are updating their WTC look-up tables and signaling BBs in their domain. Algorithm 3 outlines a DSCP generation procedure when multicast packets are duplicated at the interior branching nodes. All the algorithms are independent of the multicast routing protocol, which makes DAM a flexible approach for implementation.

Algorithm 2 DAM at Edge Router j

```

if received_DAMmessage = JOIN(r_ip, qos, G) then
  if j is on-tree node of G then
    dscp ← DEFAULT
    create a routing entry R(in, out, G, dscp)
    if qos > (the highest dscp of G at j) then
      forward REQUEST(r_ip, qos, G) to upstream
    else
      if r_ip and R.out in the same subnet then
        R.dscp ← mapping(qos)
      else
        send ACR(r_ip, j, qos, G) to local BB
      end if
    end if
  else
    send JOIN(r_ip, qos, G)
  end if

else if received_DAMmessage = REQUEST(r_ip, qos, G) then
  if qos > (the highest dscp of G at j) then
    forward REQUEST(r_ip, qos, G) to upstream
  else
    send RAR(r_ip, j, qos, G) to the local BB
  end if

else if received_DAMmessage = UPDATE(r_ip, newdscp, G) then
  if R.in from other DiffServ Domain then
    update WTC table
  end if
  newdscp ← mapping(R.in, newdscp)
  R.dscp ← newdscp
  if r_ip and j not in same subnet then
    unicast UPDATE(r_ip, newdscp, G) to downstream node
  end if

else if received_DAMmessage = ACA(r_ip, newdscp, G) then
  if ACA = positive then
    if R.in from other DiffServ Domain then
      update WTC table
    end if
    R.dscp ← newdscp
    unicast UPDATE(r_ip, newdscp, G) to downstream node
  else
    discard received_DAMmessage
  end if

else
  discard received_DAMmessage
end if

```

Algorithm 3 DSCP Generation in DAM Forwarding

```
for every multicast oif do
  dscp ← oif.dscp
  for every dscp in the packet's hdhe do
    if packet.dscp.class = dscp.class then
      dscp ← packet.dscp
    end if
  end for
end for
```

3.6 Implementation Issues

In this section, we describe the implementation details of the proposed DAM technique.

3.6.1 WTC

In order to perform weighted traffic conditioning, ERs should maintain a look-up table and they should be informed of the number of replications of each multicast flow. The look-up table should contain the following fields: multicast group ID, DSCP and number of replications.

The architecture of traffic conditioners at the edge routers should be thus changed to facilitate weighted multicast metering, as illustrated in Figure 3.5.

When packets enter the edge router, they will enter either unicast classifier component or multicast classifier component based on their destination address. In the unicast case, the traffic conditioning structure remain unchanged. If a multicast flow *f* enters a domain, given that its destination IP address is a class D address, it goes to the multicast classifier unit and then checks the look-up table for the weight. As shown in the example of Table 3.1, the look-up results indicate that this flow has

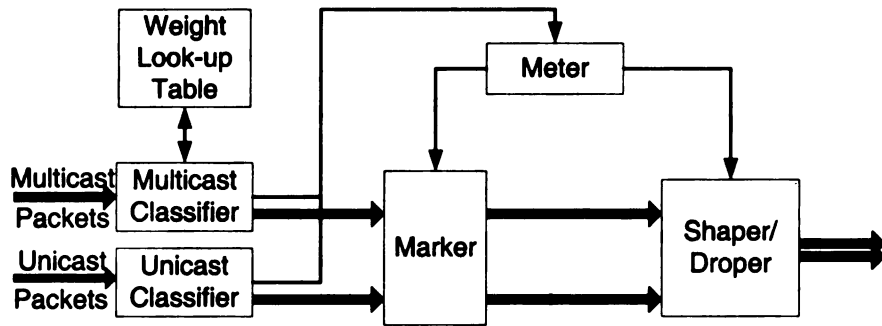


Figure 3.5: Logical View of Traffic Conditioner with WTC Component.

Destination Address	DSCP	Weight
226.35.7.28	EF	1
226.35.7.28	AF1	3
226.35.7.28	AF2	2
IP of flow f	$D1$	$w1$
IP of flow f	$D2$	$w2$
...

Table 3.1: An Example of Multicast Flow Weight Look-up Table.

two DSCP codes: $D1$ and $D2$, with their weights $w1$ and $w2$, respectively. It means that the flow has $w1+1$ branches marked as $D1$ and $w2+1$ branches marked as $D2$ leaving this domain. Thus packets of flow f should be shaped and conditioned based on weight look-up results. The other entries in the table correspond to AF and EF packets as indicated.

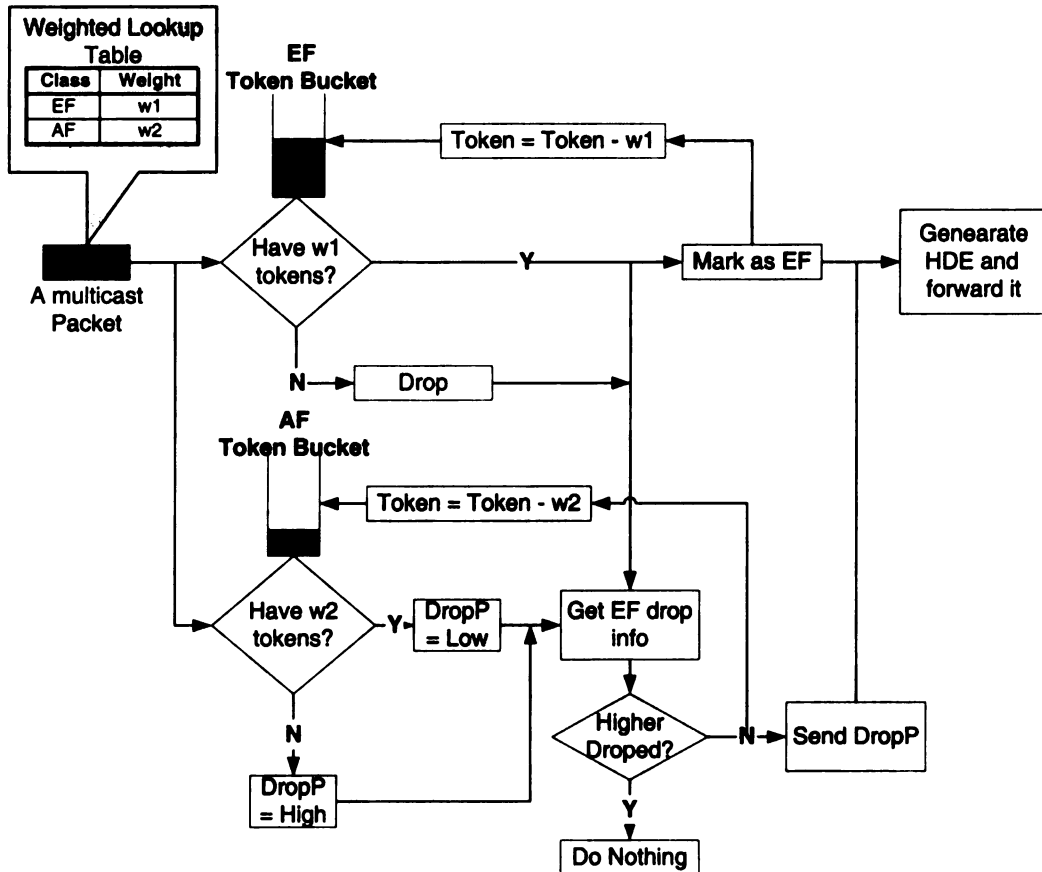


Figure 3.6: Token Bucket Marking Implementation at the Edge Routers

3.6.2 Marking

Figure 3.6 illustrates a token bucket implementation scheme of DAM marking at the edge routers. For simplicity, we assume that the DiffServ domain supports two classes of forwarding schemes, EF and AF, respectively. Further, we assume that each packet consumes one token.

To facilitate the receiver initiated marking scheme, every multicast-capable router needs to have one more DSCP field setup for the multicast flow. This extra field was also suggested in [21].

3.7 Performance Analysis

We provide both qualitative and quantitative performance evaluations of the proposed multicasting approach through analyses and simulations.

3.7.1 Qualitative Analysis

In this section, the proposed DAM technique will be evaluated qualitatively using the following performance matrices: scalability, flexibility, feasibility, and complexity.

- **Scalability.** DAM is scalable because it is built on top of the DiffServ model by pushing the complexity to the edges of the networks. Compared to the traditional DiffServ approach, DAM adds only one extra overhead, which is the multicast flow weight look-up tables maintained at the edge routers. However, this overhead is not significant and it will not adversely affect its scalability since the WTC look-up table is used only during traffic classification at the ingress point of the DiffServ domain. This WTC look-up table keeps per-flow per-DSCP records, which can be viewed as a variant of unicast Multi-Field (MF) classification [26]. After multicast weighted traffic conditioning is done, traffic can be aggregated without discriminating whether they are multicast traffic or not. In terms of traffic treatment, DAM is a *per-aggregation* based scheme rather than a *per-flow* based scheme.
- **Flexibility.** By its design, DAM is independent of the underlying routing protocol. Therefore, it can work in heterogeneous networking environments. This

routing independent feature also allows the coexistence of DAM with QoS-enabled multicast routing techniques like layered multicasting, multi-channel multicasting, multi-path multicasting and multiple-unicast multicasting. Further, DAM is isolated from the implementation details of BB architecture, so any changes made in the evolving BB design will have little impact on it.

- **Feasibility.** DAM is implemented in such a way that it requires routers to make only slight modifications. Every DiffServ capable router should add a 'DSCP' field in the forwarding table. For DiffServ edge routers, it needs to modify DiffServ traffic classifier and maintain WTC look-up tables. However, these changes are unavoidable to support correct traffic metering and pricing scheme in DiffServ multicasting. Also, DAM can be easily adapted to the situations where core routers are not multicast capable by implementing only Algorithm 2 at edge routers.
- **Complexity.** DAM adopts simple algorithms at both edge routers and core routers. The major overhead of these algorithms is performing weighted traffic conditioning at edge routers. In DAM, QoS is provided by the underlying DiffServ architecture. It avoids the complicated procedures of searching QoS-satisfied paths.

3.7.2 Quantitative Analysis

In DiffServ architecture, senders or receivers are not directly attached to the core routers, so multicasting can save bandwidth only when a core router is the bottle-

T	input EF traffic amount at an edge router in a DiffServ domain
T_c	maximum EF Input capacity of an edge router
a	ratio of EF multicast traffic over total EF traffic at an edge router
b	average duplication times within a DiffServ domain
N	number of DiffServ domains that a multicast tree spans
Pa	probability of having sufficient resources in a DiffServ domain
Pm	probability of having a branch marked higher than or equal to requested QoS in a DiffServ domain

Table 3.2: Notations.

neck, which is the case in actual networking environments. The analysis in this section assumes that after traffic conditioning at the edge routers, core routers still may get congested.

The quantitative analysis mainly focus on EF UDP traffic within a single DiffServ domain. Other results are shown in Section 3.7.3. Notations used in this section are listed in Table 3.2.

1. Amount of EF Output vs. Input at the DiffServ edge router:

The following equations show the relationship between the amount of EF input and actual EF output from a DiffServ domain for a multicast operation.

$$T_{wtc} = \begin{cases} (1 + a * b) * T, & \text{where } (1 + a * b) * T \leq T_c \\ T_c, & \text{where } (1 + a * b) * T > T_c \end{cases}$$

$$T_{Normal} = \begin{cases} (1 + a * b) * T, & \text{where } T \leq T_c \\ (1 + a * b) * T_c & \text{where } T > T_c \end{cases}$$

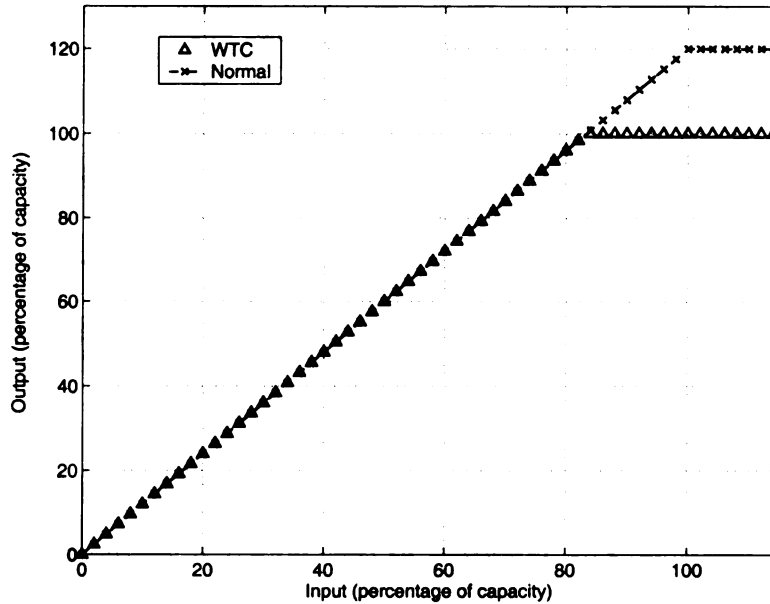


Figure 3.7: Performance of EF at A DiffServ Edge Router ($a=0.1, b=2$).

For the Normal DiffServ multicasting, multicast flows marked as EF are duplicated in the domain. Therefore, the actual amount of outgoing traffic will be $(1 + a * b) * T$, rather than T . However, the edge routers are unaware of the actual traffic amount. They only drop packets when T exceeds their capacity. For the proposed WTC scheme, packets will be dropped based on actual accounting of outgoing EF traffic. That is, packets will be dropped when $(1 + a * b) * T$ is greater than T_c .

The numerical results are plotted in Figure 3.7. It shows that WTC approach fully conforms to the SLAs since the actual output does not exceed its EF input capacity. But the normal multicast DiffServ method violates the SLA.

2. Message overhead

DAM is associated with three types of signaling messages: non-AC/BB signal-

ing, AC signaling, and BB signaling. The first type of messages includes 'JOIN', 'REQUEST' and 'UPDATE'. It can be observed from Algorithms 1 and 2 that the number of these non-AC/BB signaling is a linear function of the number of DiffServ domains that a multicast tree spans. AC signaling is required in DAM for avoiding NRS problems. Logically, each new branch of a multicast flow requires one round of signaling.

The last type of messages is related to BB. In DiffServ architecture, BBs will be signaled for every SLA negotiation or resource allocation request. For any approach based on DiffServ, it is essential to reduce the BB signaling overhead. So we focus on quantifying BB signaling overhead in this part.

Current Q-bone BB design adopts a 'nailed up' model [25]. BBs will stop signaling the next domain if the negotiation between two intermediate domains produce a negative result. Thus the signaling will be stopped at the i th (except the first and last) domain when previous $i - 1$ domains have sufficient resources and the i th domain does not. Assuming P_a is uniformly distributed, the Average BB Signaling Number (ABSN) for a unicast EF flow crossing N DiffServ domains is given by

$$ABS(N, P_a) = 2 + 2(N - 1)P_a^{N-1} + 2 \sum_{i=2}^{N-1} (P_a^{i-1}(1 - P_a)(i - 1)).$$

For DAM, BB signaling starts from the join domain instead of the domain in which the multicast tree root locates. The join domain is the one which has a branch marked higher than or equal to the requested QoS and is the nearest to the new receiver. So the average of BB signaling can be described by

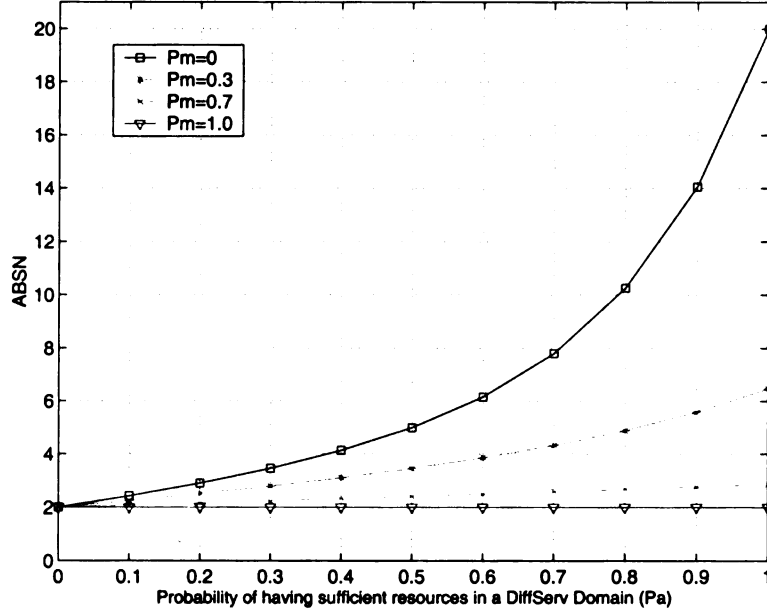


Figure 3.8: Average BB Signaling of DAM.

$$ABS_{N_{DAM}} = 2P_m + (1 - P_m)^{N-1} ABSN(N, P_a) + \sum_{i=2}^{N-1} (P_m ABSN(i, P_a) (1 - P_m)^{i-1}).$$

We further define the Relative BB Signaling Cost (RBSC) as:

$$RBSC = \frac{ABS_{N(N, P_a)}}{ABS_{N_{DAM}}}$$

Figure 3.8 shows $ABS_{N_{DAM}}$ with varying P_a and P_m . We observe that $ABS_{N_{DAM}}$ decreases when P_a decreases, which means that when network resources become scarce, the BBs are less likely to be signaled. This feature reduces the message overhead of the network when congestion happens. Figure 3.8 also shows that $ABS_{N_{DAM}}$ increases when P_m decreases. In the worst case, when P_m equals to 0, the $ABS_{N_{DAM}}$ is the same as the unicast case. Figure 3.9 depicts the variation of $RBSC$ with respect to P_m and P_a , which indicates that the relative BB signaling cost drops when P_m increases. The BB signaling overhead analysis reveals two

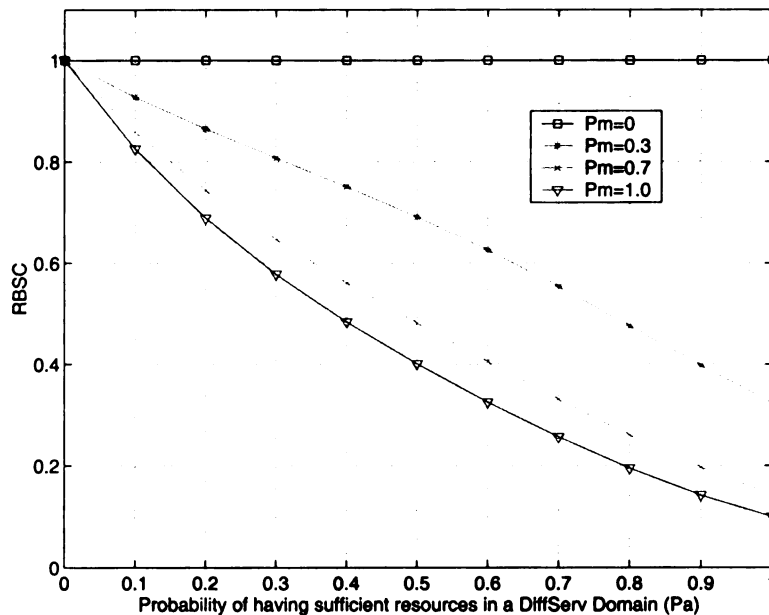


Figure 3.9: Relative BB Signaling Cost of DAM.

features of DAM. First, BBs will not be over-burdened when congestion occurs. Second, the relative BB signaling cost is less than or equal to the unicast case.

3.7.3 Simulations Results

We have implemented the normal DiffServ multicasting, and DAM on the NS simulator [27]. The goal of the simulations is to evaluate the above approaches by comparing their packets transmission ratios, as defined the ratio of the number of packets transmitted over the total number of packets. This study focuses on a single DiffServ domain. In the simulation, we assume that multicast traffic is based on UDP. For each scenario, unicast UDP and TCP traffic are studied. The network topology of the simulation is illustrated in Figure 3.10. The bandwidth of each link is 10Mb. Consider that S1 is delivering multicast packets at a rate of 1Mb per sec-

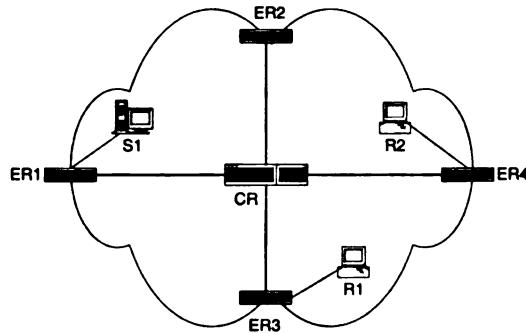


Figure 3.10: Network Topology of the Simulation.

ond through ER1, CR and ER3 to a multicast receiver R1. Host R2 wants to join the multicast group. Existing unicast traffic flow aggregations are: ER1 to ER4 – 4Mb BE; ER3 to ER4 – 2Mb EF; and ER2 to ER4 – 6Mb AF.

If the multicast flow is EF traffic and the maximum rate of EF traffic which are allowed to enter the domain at ER1 is 2Mb, then DAM produces the same results as the normal DiffServ approach. EF traffic has the highest priority level, and no packets are dropped unless the amount of EF traffic exceeds the link capacity. Thus for the EF multicast flow, we mainly study its impact on other traffic classes, such as AF traffic and BE traffic.

Figure 3.11 illustrated the EF traffic simulation results. For AF class traffic in this domain, we have studied both UDP traffic and TCP traffic. Both results indicate that normal DiffServ multicasting noticeably reduces the packet transmission ratio of BE traffic, while DAM approach has little impact on the existing traffic. The results agree with the quantitative analysis we performed in Section 3.7.2.

If the multicast flow belongs to AF traffic and the maximum rate of AF traffic that is allowed to enter the domain at the edge router ER1 is 5Mb, then a part of the AF

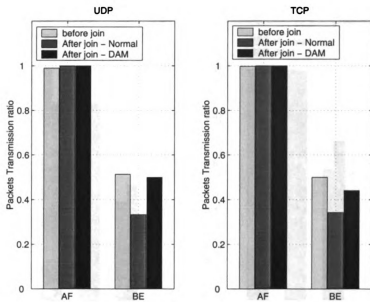


Figure 3.11: EF Multicast Results.

traffic will be marked down to BE if DAM technique is adopted.

The simulation results of AF multicast traffic are shown in Figure 3.12. When the existing AF traffic is UDP based, the normal DiffServ multicasting approach produces better results for new AF multicast flow at the cost of severely dropping BE traffic, while DAM forms a compromise between the new multicast traffic and the existing BE flows. The packet transmission ratio of the multicast flow is about 0.9 in DAM without any significant impact on the BE traffic. When other AF traffic belongs to TCP, the results demonstrate the same trends except that unicast TCP AF traffic remains unchanged in terms of packet transmission ratio. This behavior is due to the TCP congestion control mechanism.

Both EF and AF simulations indicate that WTC is necessary when applying per-aggregation-based resource management schemes in DiffServ domains. SLA

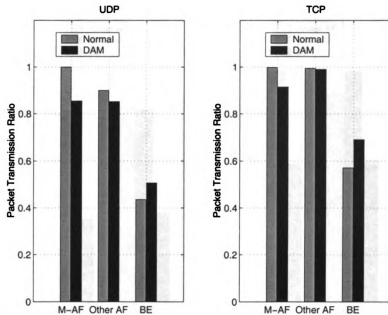


Figure 3.12: AF Multicast Results.

violation problems can be avoided in DAM without per-flow resource management approaches like RSVP.

As discussed in Section 3.4, HDE approach is adopted in DAM to improve fairness. For the same network topology, we assume one EF multicast flow which originates from ER1 is remarked to AF at CR for a sub-tree through ER4. Comparisons have been made with unicast AF flows traveling through ER1 to ER4 to study packets transmission ratio. Traffic distribution on link CR to ER4 is: 2Mb EF, 1Mb AF multicast, 6Mb unicast AF and 6Mb unicast BE. Figure 3.13 clearly illustrates that the multicast AF traffic presents nearly the same performance as that of unicast UDP AF traffic when HDE is used. For unicast TCP AF traffic case, around 10% of the multicast AF packets are dropped with HDE, while nearly no multicast AF packets get dropped without HDE. Packet transmission ratio is not affected for

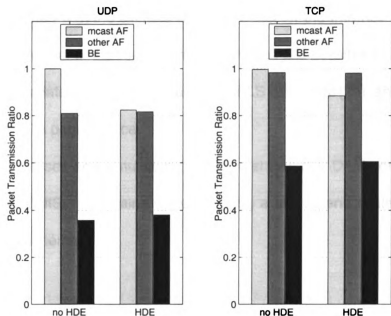


Figure 3.13: AF Fairness Results

TCP AF traffic. But the average transmission rate decreases about 4 percent with HDE. Both the results show HDE scheme ensures fairness between AF multicast flows and AF unicast flows.

In short, DAM technique avoids the SLA violation problem and unfairness issue introduced in DiffServ multicasting environments.

3.8 Summary

In this chapter, we proposed a DiffServ-Aware Multicasting (DAM) technique to provide QoS in multicasting. In DAM, the NRS problem is solved by Weighted Traffic Conditioning (WTC) at the edge routers, and the heterogeneity in QoS requirements of the receivers are handled by Receiver-Initiated Marking (RIM). Fairness is achieved with Heterogeneous DSCP Encapsulation (HDE). DAM protocol can

be easily integrated with the existing DiffServ model for the Internet. It is scalable and also independent of the routing protocol. Hence it is possible to incorporate it with other QoS-related approaches, such as MPLS [28], RLM [29], and QMRP [17] to further boost its performance.

Through analyses and simulations, we have shown that DAM conforms to the SLAs between DiffServ domains while requiring a simple and scalable resource managements scheme.

CHAPTER 4 EDGE ROUTER MULTICASTING WITH MPLS-TE

Multi-Protocol Label Switching (MPLS) is evolving as an efficient approach for packet forwarding in the internet. In this chapter, we describe an approach for QoS-aware multicasting in MPLS. To get around the difficulties mentioned in Section 2.3, we have proposed an *edge routers multicasting* scheme, as described in Section 4.1 and Section 4.2.

4.1 Basics of Edge Router Multicasting in MPLS (ERM)

4.1.1 Motivations

We assume that in MPLS domains, multicast group members are directly attached to edge LSRs, and core LSRs are only connected with other LSRs. The proposed edge router multicasting scheme tries to construct a multicast tree whose branching points are only located at edge LSRs. As shown in Figure 4.1, edge LSRs ER1, ER2, ER3 and ER4 are active members of a multicast group. Figure 4.1.a depicts the multicast tree produced by conventional IP multicast routing protocols. The branching nodes are core LSRs CR1, CR2 and CR3. In ERM, a multicast tree branches at edge LSRs ER1 and ER4, and is connected by pre-established LSPs, namely LSP1, LSP2, and LSP3, respectively, as shown in Figure 4.1.b.

By limiting the branching points only at the edges, conceptually, ERM converts a multicast flow into multiple quasi unicast flows on the network layer in each MPLS domain. Compared to native IP multicasting, ERM scheme has distinct advantages

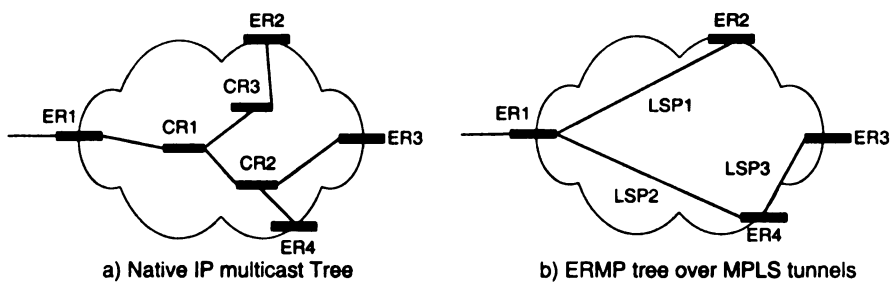


Figure 4.1: ERMP Illustration

that are itemized below:

1. *Simplifies LSP setup.* Since the diverging nodes of the tree only locate at edge LSRs, there is no need to create and maintain point-to-multipoint or multipoint-to-multipoint LSPs. Instead, a tree can be decomposed and mapped to multiple point-to-point LSPs.
2. *Makes multicast flows aggregatable.* Each branch of a multicast flow can be aggregated with other unicast flows which share the same ingress and egress LSRs. Thus the scalability of MPLS traffic engineering will not be compromised.
3. *Relaxes the requirements on core routers.* One of the reasons that IP multicast is not widely implemented is because many core routers in the backbone are not multicast ready [39]. As the core routers are usually carrying out critical missions, they are unlikely to be upgraded off-line in the near future. Edge router multicasting approach can be designed in such a way that it poses little or no multicasting restrictions on core routers.
4. *Requires no encapsulation to setup multicast tunnels.* When a multicast

router communicates with its multicast peers through non-multicast routers, a typical solution is manually building tunnels by IP-in-IP encapsulation. That is, a whole IP header is inserted in the packet leaving from the upstream peer and then it is removed at the downstream peer. While in MPLS environment, LSPs can be directly used as multicast tunnels if multicast peers are edge routers.

4.1.2 ERM Basics

ERM consists of three fundamental components: edge router multicast routing, multicast LSPs mapping, and edge router multicast forwarding.

Edge Router Multicast Routing

We focus on intra-domain routing scheme since inter-domain routing protocols like MSDP/MBGP [50, 51] and BGMP [52] allow each autonomous system having its own multicast implementation. For ERM, different multicast routing algorithms need to be developed to construct ERM trees. We first present a simple solution by slightly modifying existing IP multicast routing protocols. In Section 4.2, a Steiner tree-based heuristic routing algorithms will be discussed in detail.

For sparse mode IP multicasting like PIM-SM, and CBT [57, 55], multicast trees are constructed by explicit join. To extend these routing schemes for edge router multicasting, the following two steps need to be adopted.

- Select edge routers as the core or Rendezvous Point (RP) of the tree.

Source IP	Group IP	Outgoing Interfaces	Downstream Edge Peer IP
4.10.25.10	234.62.37.6	1	63.42.7.91
		2	63.1.3.85
18.2.36.88	242.11.7.8	2	63.15.9.28

Figure 4.2: Multicast Routing Table at An Edge LSRs

- Allow a sub-tree to join only at the edge routers.

For dense mode protocols, such as DVMRP, MOSPF and PIM-DM [53, 54, 56], multicast delivery trees are built by flood and prune. To support ERM, the process should be changed to 'flood and acknowledgment'. Each edge router should inform its upstream peer explicitly whether it has any active members on its outgoing interfaces. Each edge router should also keep the multicast state, and record its next downstream edge routers, if any. Reverse path forwarding algorithms can still be employed to limit the impact of flood.

In both modes, core LSRs are involved in building multicast trees, but they do not need to keep the multicast state. ERM routing table at edge LSRs should record its downstream peers in addition to downstream outgoing interfaces. For example, in Figure 4.2, for multicast state (4.10.25.10, 234.62.37.6), the outgoing interface is 1 and 2, with downstream edge peer 63.42.7.91 and 63.1.3.85 respectively.

Multicast LSPs Mapping

After the multicast routing process, each edge LSR has the knowledge about its downstream peers. A multicast flow can thus be mapped onto multiple LSPs based on downstream destination addresses of an edge LSR and QoS requirements of the flow as if there are multiple unicast flows destined to downstream peers. In Figure 4.2, a multicast flow from 4.10.25.10 to 234.62.37.6 will be mapped onto two unicast LSPs destined to 63.42.7.91 and 63.1.3.85 respectively.

Edge Router Multicast Forwarding

When multicast packets need to be forwarded in the ERM protocol, edge LSRs need to duplicate packets based on their routing table, and assign corresponding MPLS labels. Core LSRs do not have to duplicate any packets. The forwarding decisions can be made by simply examining the incoming labels. In fact, core LSRs do not have to distinguish whether a label is associated with multicasting or not, because in ERM, they only have one outgoing interface for each incoming packet.

4.2 Extension to ERM Routing

The multicast routing approach described in Section 4.1 is easy to implement, and it requires only minor modifications in the current multicasting protocols. However, it still demands core routers participate in the multicast routing process. In MPLS-TE, we assume that the network resource usage and availability is either available

from centralized management nodes or from each edge LSRs. Thus an ERM based Steiner heuristic tree can be constructed without the involvement of core LSRs, which leads to an extended version of the ERM protocol, termed as ERM2 in this dissertation.

4.2.1 Basic Characteristics

- *Source-based Tree.* EMR2 constructs a multicast tree per source for the consideration of implementation issues. Source-based tree has an advantages over core-based tree in address allocation, since each source can freely pick any address and create a unique (S,G) state. Moreover, core-based tree are typically shared among each group member, which also requires the support of bi-directional tree. Bi-directional LSPs are still under investigation in the current MPLS architecture.
- *Explicit Join.* We avoid using flood-and-prune approach for the following reasons. First, the density of a multicast group is likely to be sparse compared to the size of internet. Explicit join would be more efficient in such scenario. Second, flood and prune are traffic driven, not control driven. When a multicast flow arrives at the edge of a network, it needs to set up a tree first, only after which the flow assignment algorithms can be executed to map the flow onto an LSP. This will increase the latency for the delivery of the first very packet.

- *Centralized Control* We proposed a dedicated node called "Multicast Manager" (MM) in ERM2. The role of MM is different with that of "core" or "RP" in CBT or PIM. MM is not designated to be the root of a delivery tree. Rather, it functions like a DNS server, and is responsible for group membership management in an MPLS domain. It keeps a record of current active on-tree edge routers and returns a list of candidates to a new receiver. The merits of centralized control includes easy implementation and simplified routing algorithms.
- *Protocol Independence.* In view of heterogeneous nature of internet, ERM2 is designed to be independent of unicast routing protocols. Thus it can be implemented on top of distance vector protocol as well as link state protocol.

4.2.2 ERM2 Illustration

The "Join" process in ERM2 can be illustrated by the example depicted in Figure 4.3, where edge router E1, E2, E4, and E7 are on-tree routers of multicast group G. Suppose edge router E5 wants to join group G. The routing procedure is enumerated as follows.

1. Edge router E5 sends a QUERY message to MM.
2. MM returns an ANSWER message with a list of candidates to E5. In this example, the candidates are S, E1, E2, E4, and E7.
3. Based on its own routing table, or resource availability, E5 picks the best the candidate, say E4, as the join point.

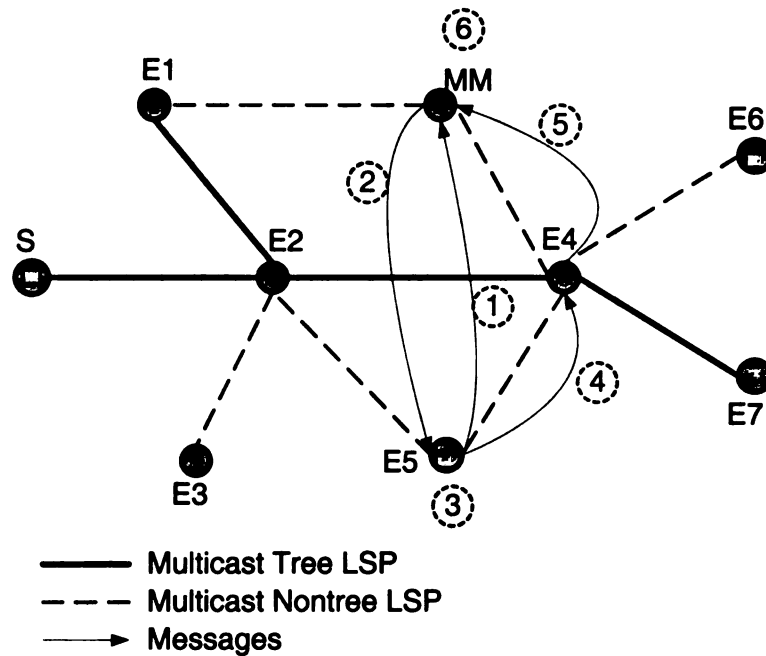


Figure 4.3: ERMP2 Join Example

4. E5 sends a JOIN message to E4 and E4 create an outgoing entry for state (S,G).
5. If successful, E4 informs MM that E5 is now an active on-tree edge router through an ADD message.
6. MM inserts E5 in the active member list.

An edge node will leave a multicast tree when two conditions are met. First, it detects that there is no active member directly attached to it by Internet Group Management Protocol (IGMP) report. Second, it does not have any downstream peer. The leaving node will send a SUBTRACT message to the MM to update the member list, and a PRUNE message to its upstream peer.

The state machine of edge routers is shown in Figure 4.4.

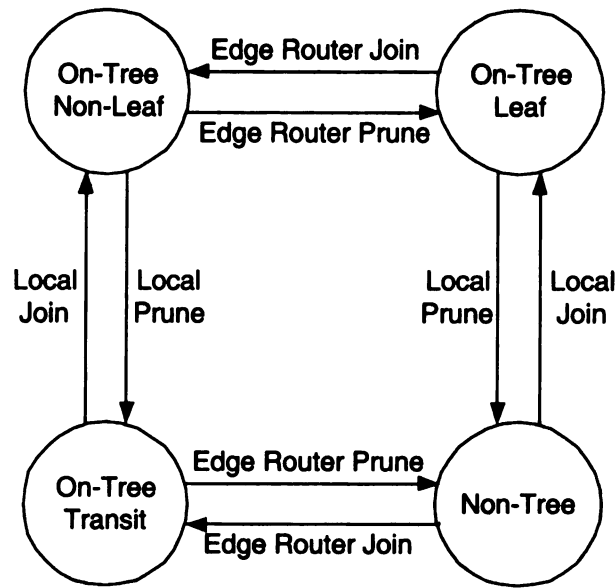


Figure 4.4: State Machine of ERMP2 Edge Routers

4.3 Performance Analysis

Edge router multicasting scheme will not create a fully optimized multicast delivery tree. Identical packets could be transmitted on the same out-going link. We have conducted a series of simulations to compare tree cost, link stress, and relative delay of following protocols(algorithms): Distance Vector Multicast Routing Protocol(DVMRP), ERM, ERM2 and Minimum Spanning Tree. The results demonstrated that edge router multicast scheme, especially when ERM2 routing is employed, will not noticeably compromise the benefits of native IP multicast.

Network topology and group density are two major factors which affect performance of multicast routing protocols. A variety of flat random graphs [60] have been proposed to model networks in aim to reflect realistic network topologies. All the variations randomly distribute vertices in a plane and add an edge between

Model	Edge Probability
Waxman1	$\alpha e^{-d/(\beta L)}$
Waxman2	$\alpha e^{-rand(0,L)/(\beta L)}$
Locality	$\begin{cases} \alpha & \text{if } d < L \times radius \\ \beta & \text{if } d \geq L \times radius \end{cases}$

Table 4.1: Edge Probability of Selected Flat Random Graph Models.

each pair of vertices with certain probabilistic parameters. We have chosen three commonly used random graph models in our study, namely Waxman1, Waxman2, and locality. The edge distribution functions are summarized in Table 4.1

In Table 1, $0 < \alpha, \beta \leq 1$, d is the Euclidean distance between two vertices, and L is the maximum distance between any two vertices. Intuitively, locality model has the richest short distance connectivity in three models, while Waxman1 generate less long distance edges than Waxman2.

For each model, we use GIT network topology generator produced 1024 nodes flat network. Among them 300 out of 1024 nodes are randomly selected as the edge routers. The simulation results are collected and tabulated by recording performance metrics in different topology by increasing group members from 5 to 300. For ERM2 routing, we assume each edge router picks the node with least cost path as the join point.

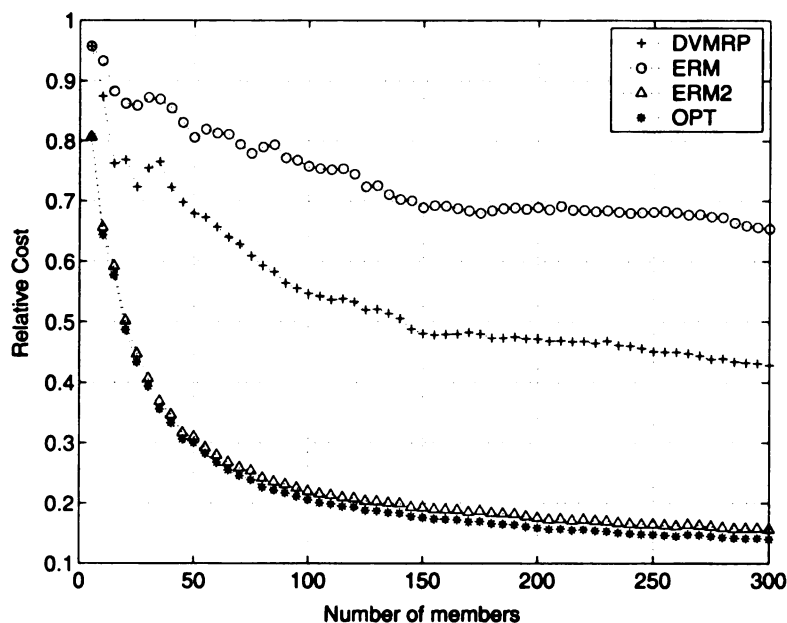


Figure 4.5: Relative Tree Cost, Locality Model

4.3.1 Relative Tree Cost

Relative tree cost is defined as the ratio of tree cost over the sum of unicast path cost. Figure 4.5, 4.6 and 4.7 show the comparison of relative tree costs. In the figures, OPT refers to optimal results produced by Minimum Spanning Tree algorithm. For all the topologies, ERM yields worst relative tree cost, while ERM2 incurs less cost than DVMRP and even demonstrates near-optimal performance. These results prove that edge router multicast scheme may not necessary leads to very high tree cost. As a matter of fact, with careful design, it could be more efficient than least-cost unicast path tree built by protocols like DVMRP. Another interesting observation inferred made from Figures 4.5, 4.6 and 4.7 is that the widely accepted multicast protocols like DVMRP only save half of the link cost when all the edge nodes join a multicast tree.

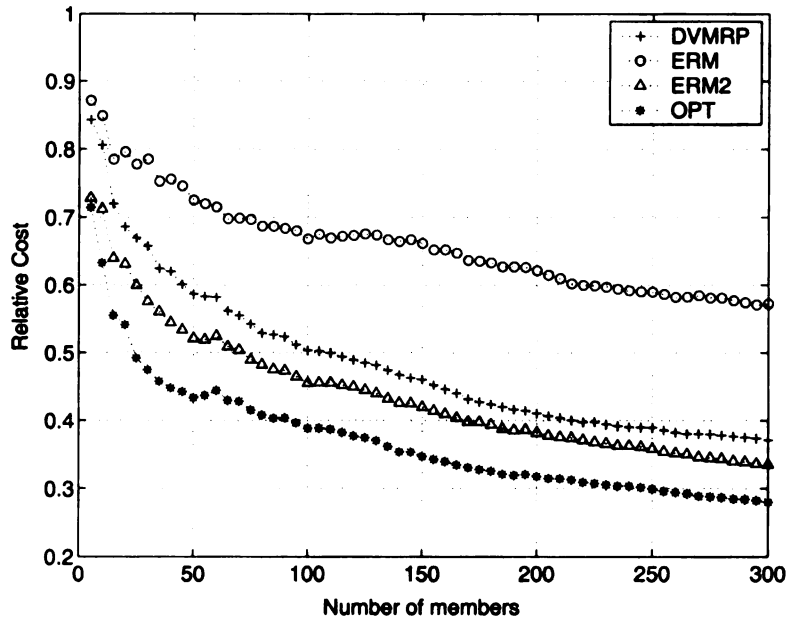


Figure 4.6: Relative Tree Cost, Waxman1 Model

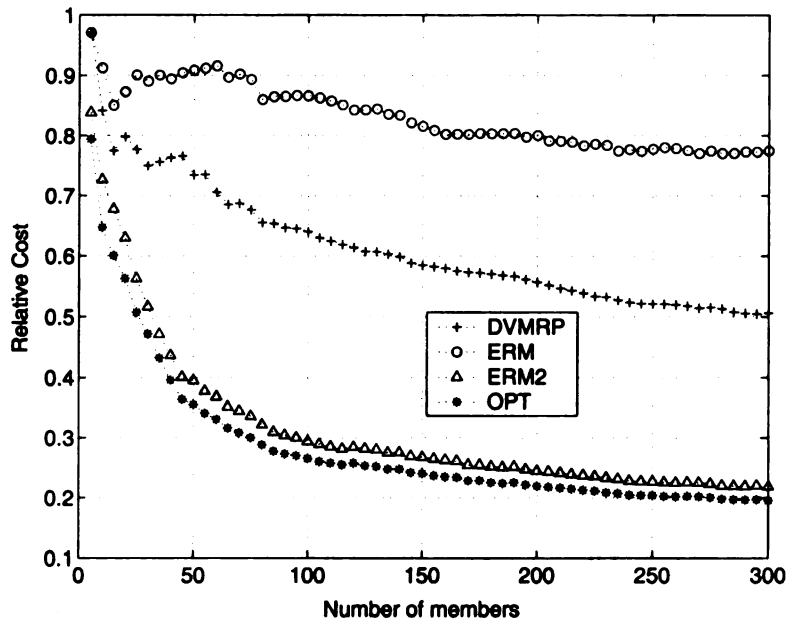


Figure 4.7: Relative Tree Cost, Waxman2 Model

4.3.2 Link Stress

Stressed link refers to those links which have multiple identical packets on the outgoing interface. The number of the identical packets is denoted as link stress. For native multicast protocols, link stress is always equals to one. For unicast, source node link stress equals to the total number of on tree node numbers in a domain. Combining tree cost results presented above, the most important feature of multicast may be releasing link stress, rather than saving bandwidth. The ERM protocol could introduce stressed link. Results of link stress are plotted in Figure 4.8, 4.9 and 4.10. ERM and ERM2 both have average link stress between 2 to 3, and the ratio of stressed link are both less than 20 percent. However, the maximum link stress of ERM is much higher than ERM2. In the worst cast, the maximum link stress is as high as nearly 40. Link stress performance can be easily improved by adding maximum link stress restrictions. The side effect of this restriction would produce worse results for other performance metrics like tree cost and relative delay.

4.3.3 Relative Delay

Relative delay is defined as the ratio of multicast end-to-end delay over unicast end-to-end delay. DVMRP and ERM protocol lead to shortest path source-based trees, thus their relative delay amounts to 1. So, this performance metric is only significant for Minimum Spanning Tree approach and ERM2. Figures 4.11 4.12 and 4.13 illustrate relative delay results. In all the topologies, and for both average and

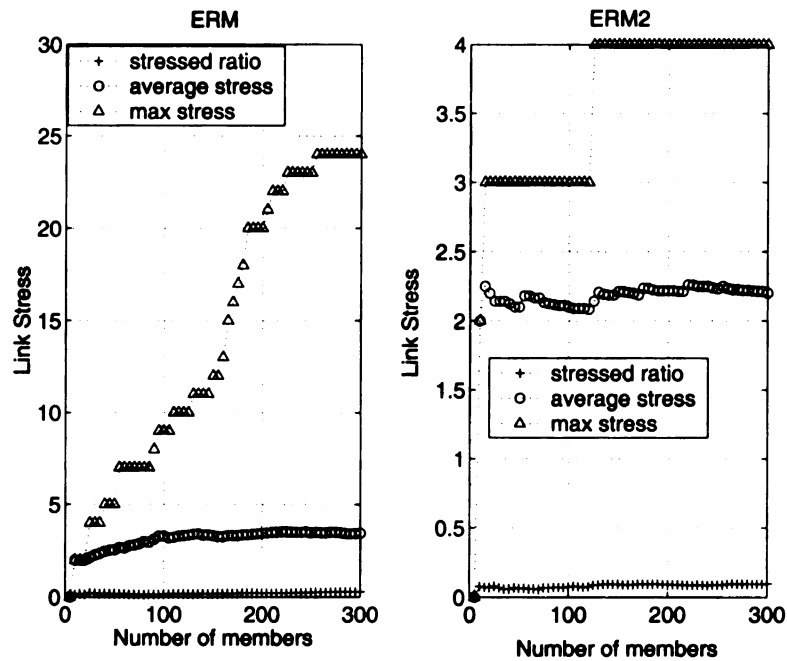


Figure 4.8: Link Stress, Locality Model

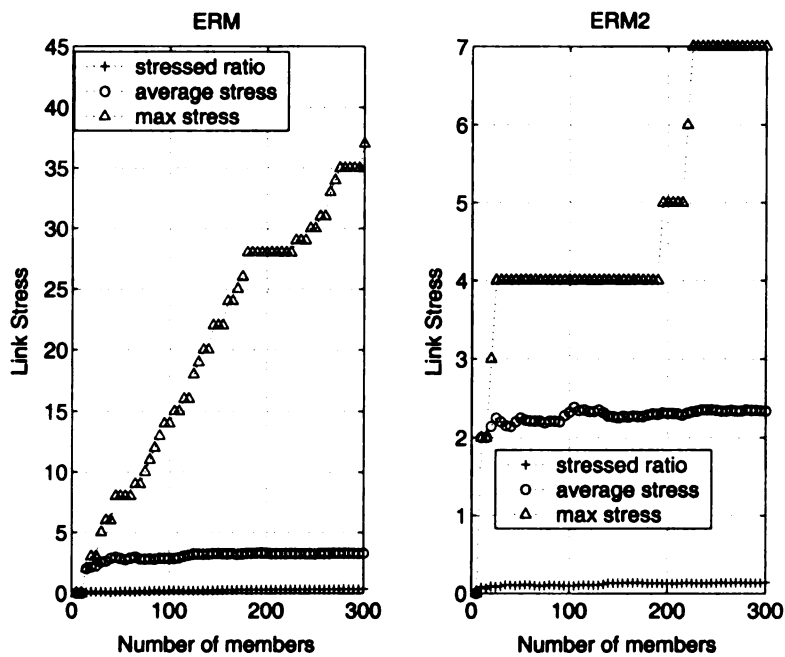


Figure 4.9: Link Stress, Waxman1 Model

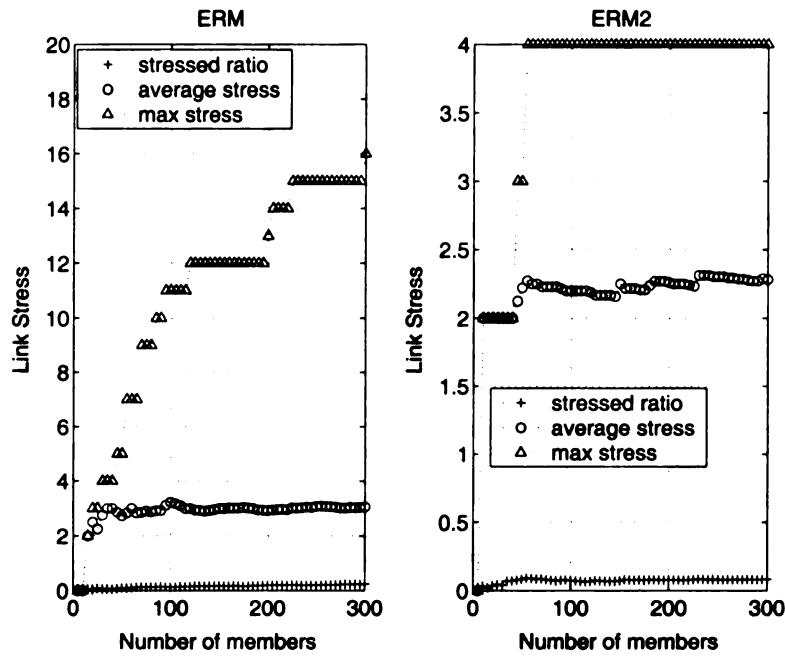


Figure 4.10: Link Stress, Waxman2 Model

maximum value, the performance of ERM2 stays closely to the Minimum Spanning Tree results. The relative delay ranges from 1 to 1.4. These results convince that ERM2 routing would not affect the relative delay.

Furthermore, for ERM technique, the average relative delay is not sensitive to topology changes. Rather, it tends to be less when group size increases. And the maximum relative delay is not sensitive to both topology changes as well as group size changes.

4.4 Summary

In this chapter, we have proposed an edge router multicasting approach in MPLS traffic engineering environment. ERM converts design of point-to-multipoint LSP setup to a multiple point-to-point LSP problems, and make multicast traffic suitable

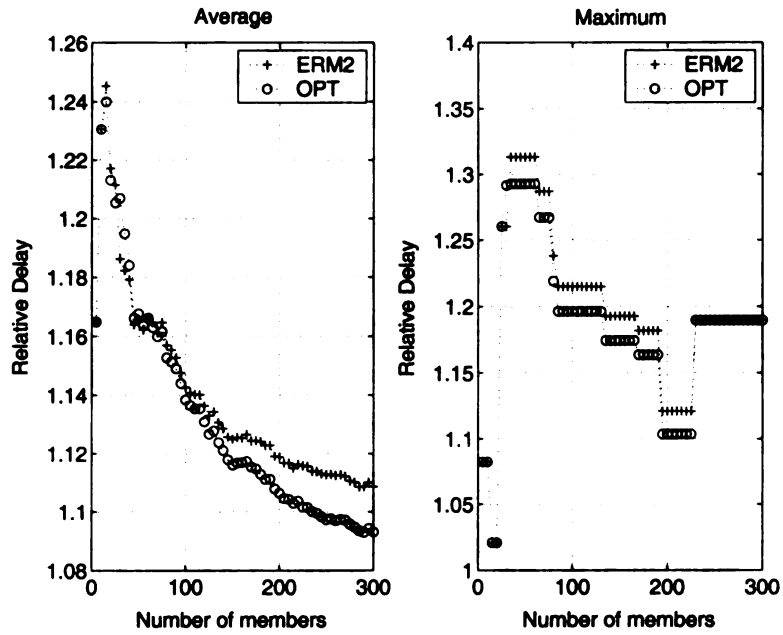


Figure 4.11: Relative Delay, Locality Model

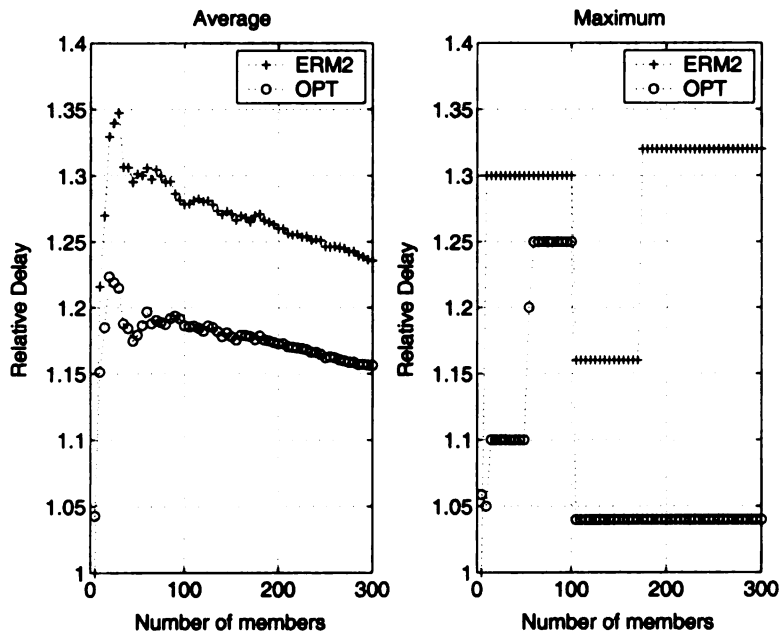


Figure 4.12: Relative Delay, Waxman1 Model

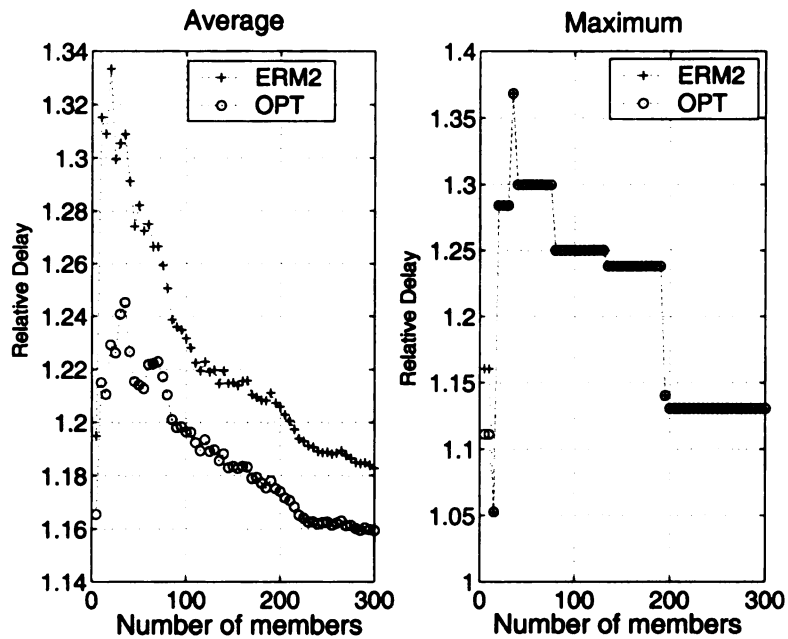


Figure 4.13: Relative Delay, Waxman2 Model

for aggregation. In the ERM protocols, the multicast trees branch only at the edge routers and use the MPLS tunnels set up by the core routers. In addition, the proposed approach does not lose the strength of native IP multicast. The implementation of the ERM protocol is incrementally deployable as it does not require any changes in the core routers. Simulation results show that the proposed ERM2 has near optimized tree cost, low link stress, and incurs low delay.

CHAPTER 5 SUPPORTING OVERLAY MULTICAST IN DIFFSERV DOMAINS

In this chapter, we focus on QoS provisioning for multicasting using application layer protocols. Specifically, we address the issues associated with overlay multicasting in DiffServ domains.

5.1 Network Models and Design Issues

DiffServ architecture offers a number of features that will affect the design of overlay multicasting.

5.1.1 DiffServ Architecture and Its Features

When designing a scheme to support overlay multicasting in DiffServ domains, we should be aware of the following characteristics of DiffServ:

- *SLAs*. As mentioned above, there is no service guarantee for out-of-profile traffic. The fanout of each end host should be bounded by its maximum amount of traffic specified in SLAs, which is less than the actual available network bandwidth. As and when necessary, SLAs can be renegotiated.
- *Signaling overheads*. In DiffServ domains, dedicated nodes, termed as Bandwidth Brokers (BBs), are responsible for any intra-domain and inter-domain resource management. According to current QBone BB design[68], end hosts need to request BBs for resource allocation to ensure that SLAs are

not violated and sufficient resources are available along the path. It implies that every link change in the overlay multicast tree may initiate a BB signaling.

- *Uni-direction.* DiffServ only provides QoS for one way traffic. In order to get a bi-directional DiffServ link, both participants should signal their BBs for resource allocation. Because the actual SLAs and network utilization could vary, such bi-directional links could fail. Thus, constructing a shared tree in DiffServ may not be feasible.

5.1.2 Design Issues

We assume the underlying network can provide either premium service of DiffServ or guaranteed service of IntServ. The overview of design issues follows.

- *Changes in tree topology.* Since every change of the tree topology would lead to BB signaling, the tree-building algorithms should construct a tree in a way such that it could avoid extensive topology changes when group members or network condition changes.
- *Source specific tree.* Since DiffServ is uni-directional, we prefer to construct per-sender-based multicast trees.
- *Direct tree construction.* There are two distinct approaches to construct overlay trees. One approach is to first set up a richer connected graph, termed as *mesh*. A tree is then constructed using DVMRP like routing protocol. Mesh first approach is good for constructing shared trees and is suitable when the

underlying networks performance or reliability is undesirable. But it would be harder to detect mesh partitions, and resulting tree topology may changes dramatically from time to time. For the above considerations, we have chosen the other scheme, i.e. to build the tree directly.

- *Sub-optimized for tree cost.* We favor minimal or sub-minimal tree-cost. Saving overall network resources is one of the major strengths of multicast. However, fully optimized schemes, such as Minimum Spanning Tree, are less likely to be implemented in practice, and would produce a deeper tree with longer latency.
- *Incremental join at proper place.* Most overlay multicast protocols are self-organizing. It is a desirable feature when there are no service guarantees and no signaling overheads associated with the changes in the tree topology. In this chapter, we present an approach that tries to insert a new member at the appropriate place of the tree, while considering the SLAs as well as the overheads associated with topology changes.

5.1.3 Performance Metrics

In summary, the solution we are seeking should yield fewer tree topology changes and have a good balance between overall tree cost and individual latency. In this chapter, we use the following performance metrics to evaluate different approaches:

- *Tree Cost.* Tree cost is defined as the sum of the link costs. If we ignore

the network layer and application layer overheads, this metric reflects the efficiency of system resource usage.

- *Link Changes*. The notation of link changes refers to the number of different links between the original tree and the new trees built after new members join. In DiffServ overlay multicasting environment, link changes incur overheads associated with BB signaling.
- *Relative Delay Penalty (RDP)*. RDP is denoted as the ratio of overlay multicast delay to the unicast shortest-path delay. This metric is defined from the perspective of each host.

5.2 Group Management and Incremental Insertion Protocol

To support overlay multicasting in DiffServ domain, there could be two approaches. One is to make QoS provisioning totally independent of tree construction algorithms. That is, after the tree is constructed, each member host then signals the edge router for appropriate packet marking. However, it may happen that by then some of the nodes have consumed all of the resources specified in their SLAs. The marker would then mark the multicast packets as out-of-profile. Unless the SLAs are renegotiated successfully, the QoS guarantee on certain branches would be compromised, and the unfairness among each multicast member would be aggravated. Thus we adopt another approach by taking each members available SLAs into account while building the tree.

5.2.1 Group Management Overview

In order to achieve better scalability, we propose a two-level hierarchical scheme which includes inter-domain and intra-domain group management. Logically, a tree topology for a cross-domain multicast group G with sender S , denoted as (S,G) , is shown in Figure 5.1. At inter-domain level, each domain except the source domain is assumed to have exactly *one* link connected with its parent domain through which the multicast tree is formed. There may or may not be other links connecting the domains, but are not used for multicasting. For example, link $RA3 \rightarrow RB1$ is the cross-domain link which hooks up domain B with its parent domain A. In the source domain, sender is the domain *multicast root*. In other domains, the node responsible for receiving inter-domain traffic is denoted as the domain *multicast root*. In Figure 5.1, the domain multicast root for (S,G) at domain A, B, C and D are S, RB1, RC1 and RD1, respectively. It should be noted that the term *domain* refers to the actual DiffServ domain as defined by IETF. We do not propose the dynamic clustering approach suggested in [66, 67], because in DiffServ, inter-domain resources could be far more scarce and its resource management is much more complicated. If the nodes from different DiffServ domains self-organize to form a *cluster* as the basic tree building block, then the resource management within each cluster would be complex.

In this chapter, we focus on studying intra-domain group management and tree construction techniques. We further assume that certain bootstrap mechanisms exist such that any node j wants to join a group (S,G) could obtain the address of

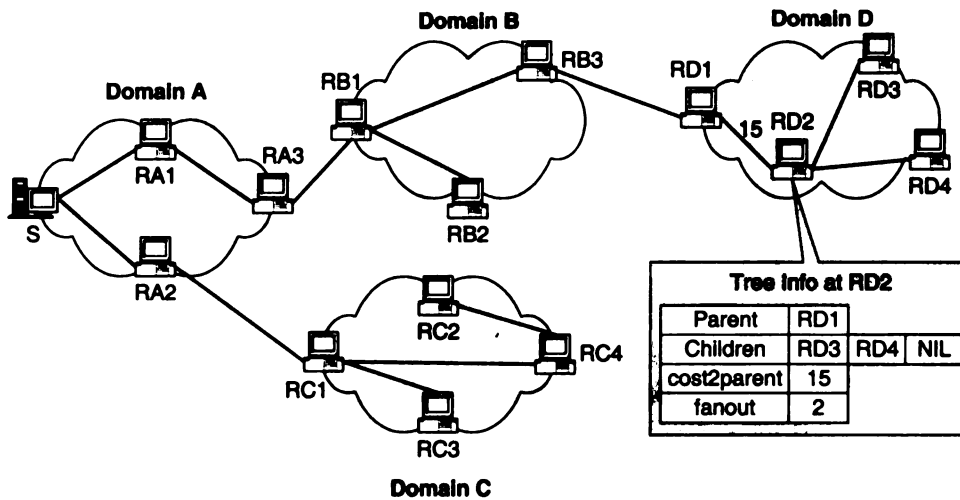


Figure 5.1: Logical View of Tree Topology for (S,G)

multicast root in its domain, and from which it could get a full list of active members in the domain.

5.2.2 Incremental Insertion Algorithm (IIA)

We have proposed an *Incremental Insertion Algorithm* for building overlay multicast trees in a DiffServ Domain by constructing the multicast tree incrementally. Thus simultaneous joins should be serialized and then be processed one by one. This serialization could be accomplished by maintaining a queue at a dedicated node such as domain multicast root.

As depicted in Figure 5.1, each active member in the multicast tree should maintain its parent node, cost to its parent, children nodes and the number of its children (denoted as *fanout* in this chapter). When node j wants to join a multicast group (S,G), it will first evaluate its cost to all the active members in the domain and then calculate the cost gain for two cases. One case is to attach itself to node

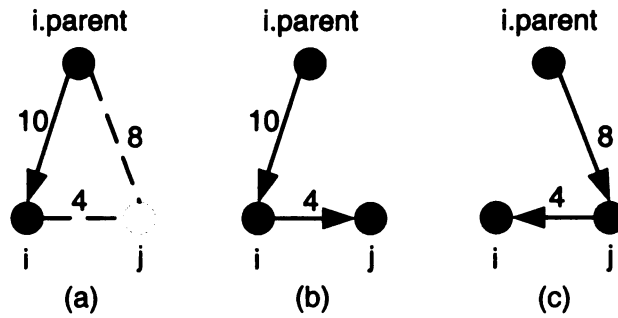


Figure 5.2: Illustration of Leaf-join and Insertion

i and become a new leaf in the tree, which is denoted as *leaf-join*. The other case is called *insertion*, that is inserting node j between node i and i 's parent node. For instance, in Figure 5.2.a, node j is evaluating its cost gain toward node i . Leaf-join would produce a new tree structure as shown in Figure 5.2.b, and Figure 5.2.c depicted the new tree structure if node j is inserted between i and $i.parent$. If numbers in the figure denote costs of the links, the cost gain for leaf-join will be 4, while insertion would be 2. Since insertion cost less, node j will evaluate cost gain toward node i as 2, with an operation *insertion*. After node j goes through all the active members, it will join to the node having minimum cost gain with the corresponding operation. The formal presentation of the algorithm is described in Algorithm 4.

One of the problems associated with IIA is that it may increase the join latency. We have defined a non-QoS working mode to allow a new member to receive multicast traffic as quick as possible by joining a random node in the tree. The new member will leave non-QoS mode after it finishes processing Algorithm 4.

Algorithm 4 Incremental Insertion Algorithm at Node j

Require: $j.cost2parent \leftarrow \infty, j.parent \leftarrow NIL, j.child \leftarrow NIL, j.fanout \leftarrow 0$ {For off-line mode only}

for every active member i do

if $i.fanout < FANOUT$ and $j.cost2parent > cost(i, j)$ then

$j.cost2parent \leftarrow cost(i, j)$

$j.parent \leftarrow i$

$mode \leftarrow leaf_join$

end if

if $i.fanout < FANOUT$ and $i.parent.fanout < FANOUT$ then

$insertCost \leftarrow (cost(j, i) + cost(i.parent, j) - i.cost2parent)$

if $insertCost < j.cost2parent$ then

$j.cost2parent \leftarrow insertCost$

$j.parent \leftarrow i.parent$

$j.child \leftarrow i$

$mode \leftarrow insertion$

end if

end if

end for

add j as a new child of $j.parent$

$j.parent.fanout ++$

if $mode = insertion$ then

 remove $j.child$ from $j.parent$

$j.parent.fanout --$

 add $j.child$ as a new child of j

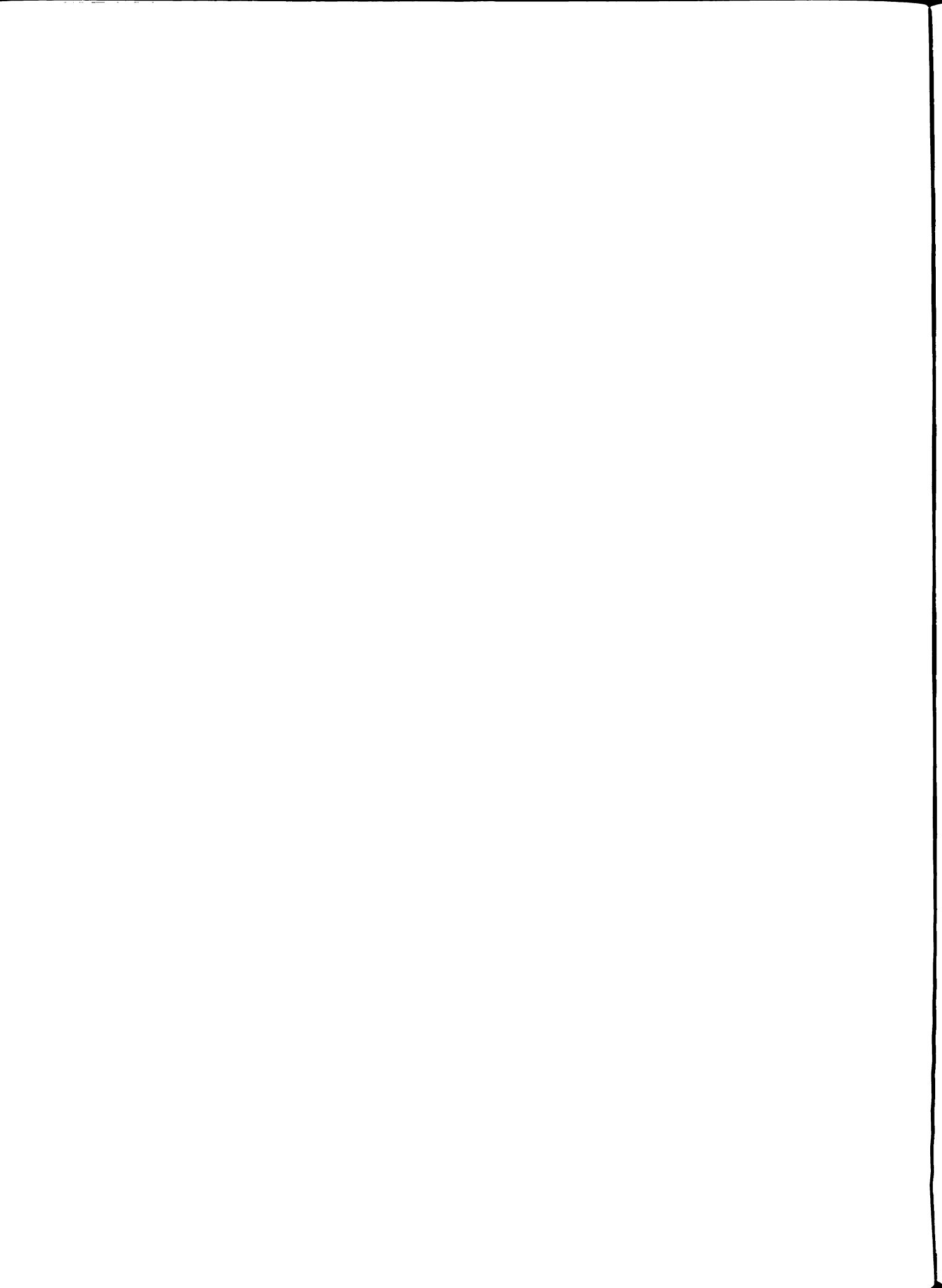
$j.fanout ++$

end if

5.2.3 Partition Repair

There are two situations that would create tree partition problem: member leaves and link/node failures. In both cases, the tree should be re-constructed. If we assuming single site failure, it could be detected if every node periodically 'ping' its parent. Member leaving can be also handled in the same way. But it would be quicker if we let the leaving node send a message to its children to trigger re-building process.

We have proposed both on-line mode and off-line mode of repairing processes. The former aims at fast re-pairing, but would generate a tree with worse performance. The later process could produce a tree with better performance, but is only suitable when there is no multicast traffic on the flight. Both repairing algorithms are straightforward. For example, in Figure 5.3a, node i leaves tree T . In on-line mode, i 's children $C1$ and $C2$ will execute a slightly modified IIA by considering $C1$ and $C2$ as two nodes wanting to join tree T . The only modification is that when a node join as on-line repairing mode, it should not go through the initialization part. A possible on-line repairing result is shown in Figure 5.3.b. We can see that on-line mode repairing would not change the trees rooted at i 's direct children $C1$ and $C2$, and only require 2 link changes in the example. On the other hand, in off-line mode, all the descendants of node i will be treated as new requesting nodes and will carry out Algorithm 4 one by one. Such repairing algorithms could produce a quite different tree structure shown in Figure 5.3.c, where the trees originally rooted at $C1$ and $C2$ are not retained.



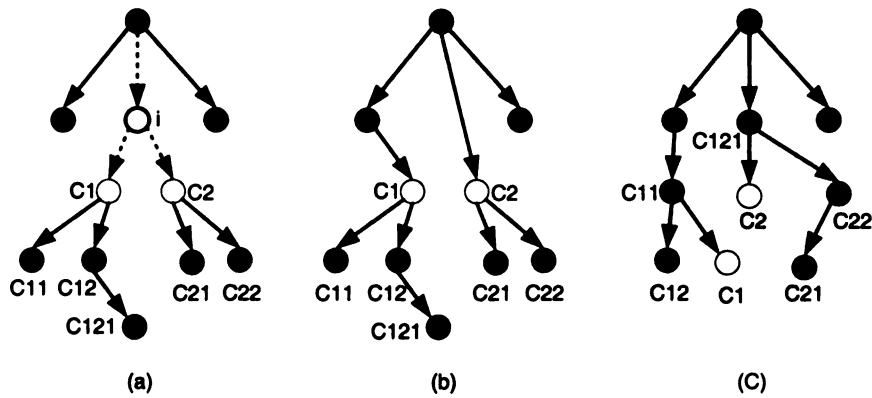


Figure 5.3: On-line Mode and Off-line Mode Repair

5.3 Simulations Results

In our simulation, we use delay as the metric of link cost. Three types of network topologies, i.e. Waxman1, Waxman2, and Locality model, are created by using GIT network topology generator [60]. Each type of network has 100 random graphs, and each graph consists of 1024 node, with cost distributed in a 100x100 plane. For every graph, we randomly pick a non-tree node and vary the group size from 2 to 100. The maximum fanout of any member is set to 6.

The proposed scheme is compared to Narada tree [61, 62], minimum-cost tree, and the minimum-leaf join tree. In the implementation of Narada, we use 0.75 as the threshold of utility gains for link addition, and choose 0.25 as the lower bound of consensus cost to drop a link. After each member joins, we allow Narada tree to stabilize before processing the next new member.

The performance metrics we used is described in Section 5.1.3 Because the results are similar for all three different type of networks, we only present the detailed results of Waxman2 topology model in Table 5.1, which summarizes the statistic of

Table 5.1: Statistic of 100 Runs (100 member, Waxman2 mode)

	Narada	Leaf-Join	IIA	OPT
Tree Cost(ave)	3605.76	1919.52	1716.34	1540.72
Tree Cost(max)	4251	2114	1898	1382
Tree Cost(min)	2915	1681	1534	1723
Tree Cost(std)	24.01	8.82	8.27	7.44
Link Changes (ave)	665.80	99	165.96	295.77
Link Changes(max)	1155	99	192	371
Link Changes(min)	331	99	138	236
Link Changes(std)	19.69	0	0.99	2.7
RDP(ave)	2.55	2.56	2.43	3.26
RDP(50th percentile)	1.92	2.09	2.11	2.77
RDP(95th percentile)	4.66	4.71	4.34	6.58
RDP(worst case)	33.0	23.20	13.55	18.75

the results. The average results of all the topologies are plotted from Figure 5.4 to Figure 5.12.

5.3.1 Tree Cost

Narada performs poorly for tree cost because it produces a tree in a DVMRP fashion, which is not optimized for overall cost. Another reason is that the mesh that Narada constructs is on a shared-tree basis. So the quality of the mesh may not be good for a source-specific tree. In Figures 5.4,5.5 and 5.6, Narada's tree cost is more than two times of that of the optimal case. As expected, minimum-cost leaf-join is worse than IIA approach. Intuitively, IIA can be conceived as another

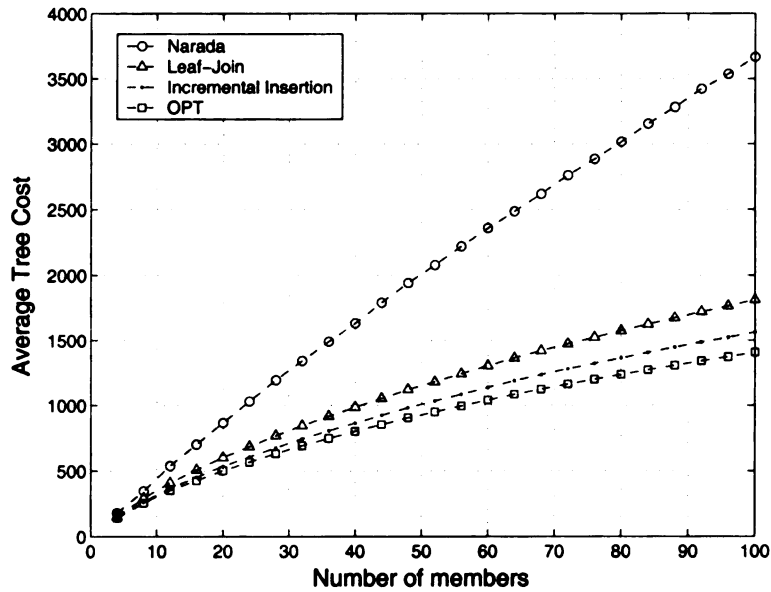


Figure 5.4: Tree Cost Comparison, Locality Model

optimization on top of *leaf-join* because it considers both *leaf-join* and *insertion*. The results show that the average tree cost of IIA is roughly 10% more than the optimal case, whereas that of average tree-cost of minimum-cost leaf-join is around 20% to 30% more the optimal case.

5.3.2 Link Changes

It is clear from Figures 5.7, 5.8 and 5.9 that both Narada and the optimal tree cause extensive link changes. It is not a surprise since both of them do not consider link change as an optimization objective. Narada is even worse than the optimal tree in our results. One reason for this behavior is that the mesh is always changing, which makes the corresponding tree topology change more aggressively. Another reason is probably because of the threshold value we chose in the simulation, which may not be good enough. Minimum-cost leaf-join demonstrate the best performance in

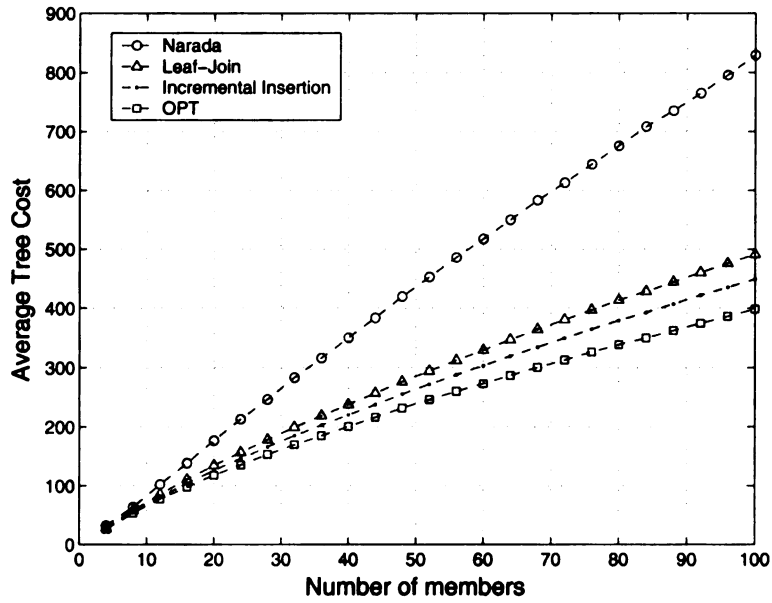


Figure 5.5: Tree Cost Comparison, Waxman1 Model

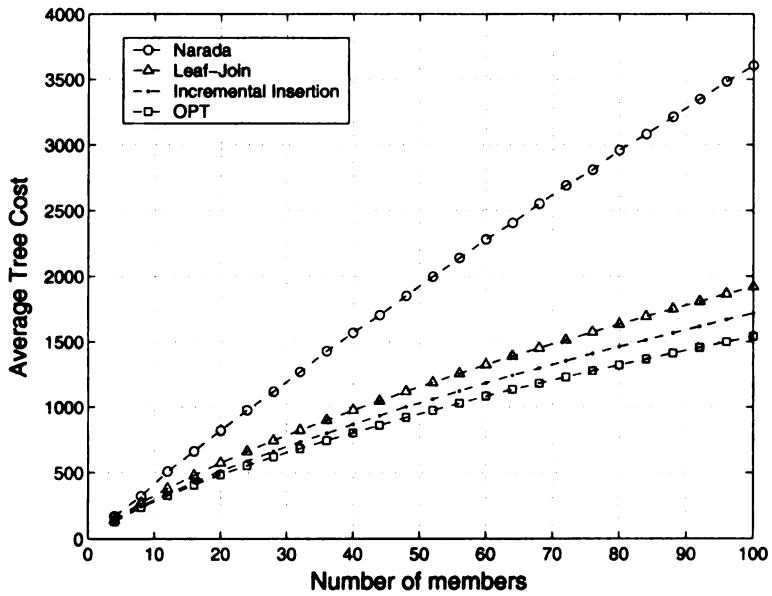


Figure 5.6: Tree Cost Comparison, Waxman2 Model

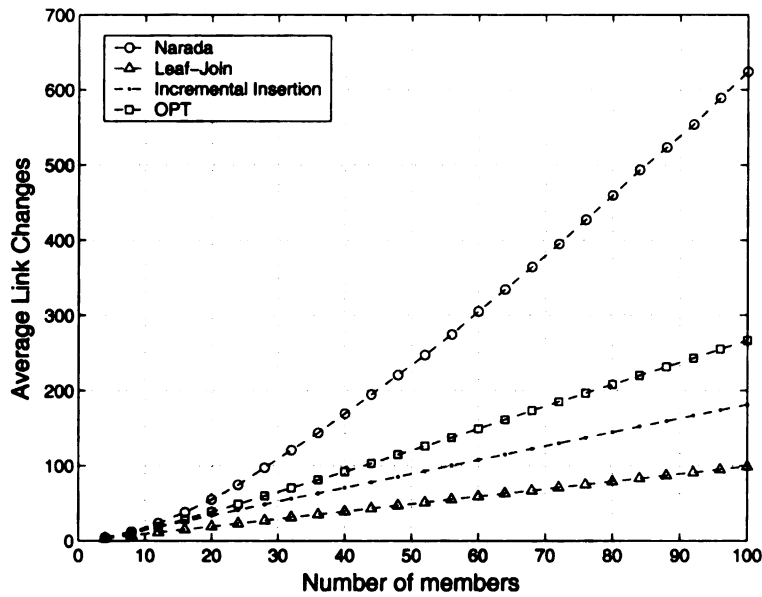


Figure 5.7: Link Changes, Locality Model

terms of this metric. The average link changes is in fact equal to $n - 1$ for Minimum-cost leaf-join, where n is the total number of members. Although IIA causes nearly two times as many as that of minimum-cost leaf-join, it is still a linear function of group members (n) with the upper bound as $3n - 3$.

5.3.3 Accumulative Relative Delay Penalties (RDP)

Optimal-cost tree incurs the lowest RDP as shown in Figures 5.10, 5.11, 5.12 and Table 5.1. Narada has slightly less average RDPs than IIA up to the 80th percentile. But its average is a little higher than that of IIA. Moreover, its worst case RDP is much larger. The possible explanation lies in Narada's shared mesh, which implies that shared delay optimized tree may not be fully optimized from each sender's perspective. Results also indicate that the IIA has better performance than minimum-cost leaf-join for all workload conditions.

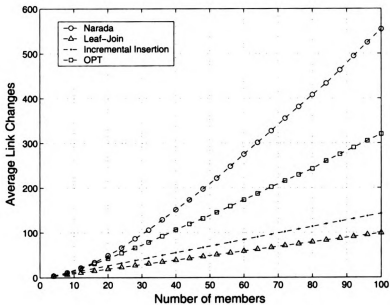


Figure 5.8: Link Changes, Waxman1 Model

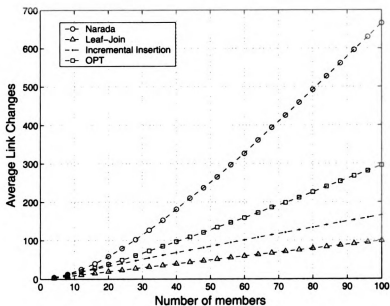


Figure 5.9: Link Changes, Waxman2 Model

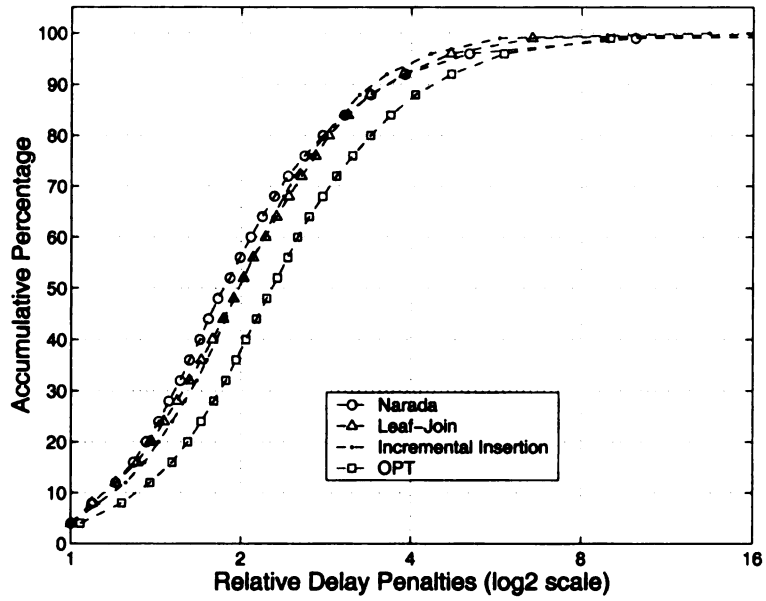


Figure 5.10: Relative Delay Penalties, Locality Model

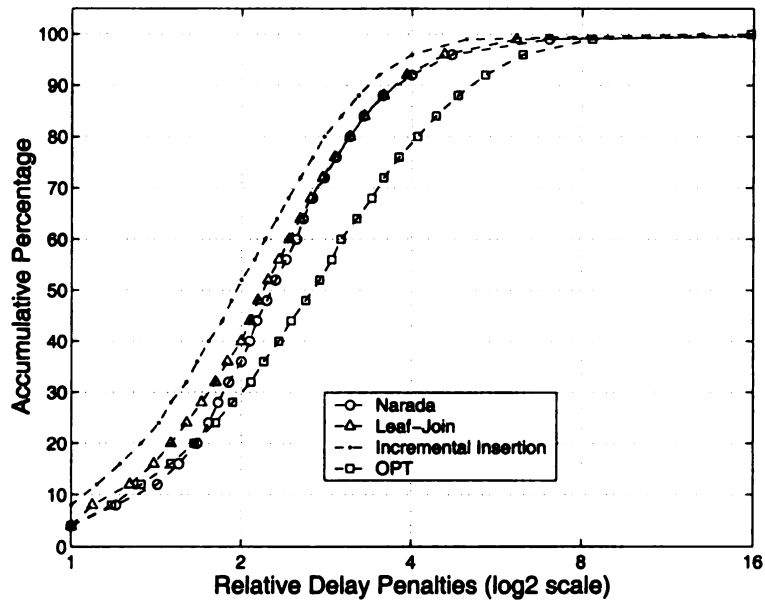


Figure 5.11: Relative Delay Penalties, Waxman1 Model

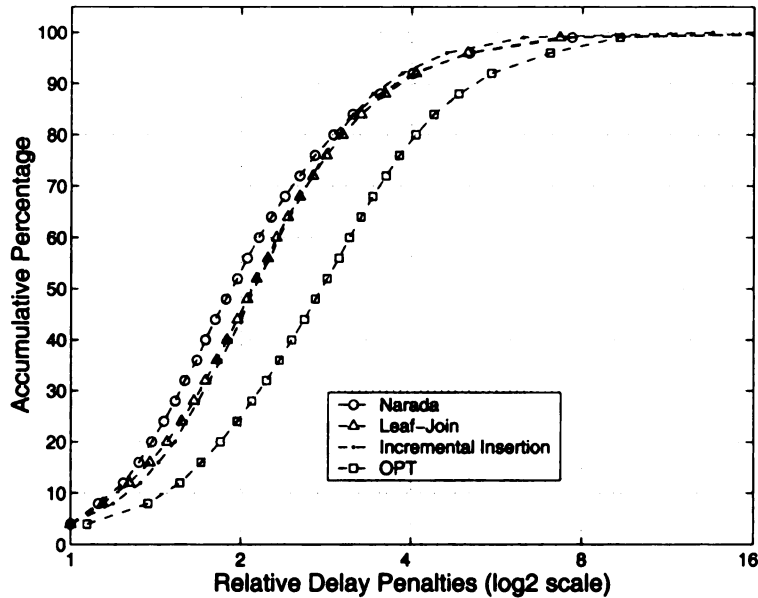


Figure 5.12: Relative Delay Penalties, Waxman2 Model

5.4 Summary

In this chapter, we have proposed an heuristic algorithm named Incremental Insertion to construct source-specific multicast overlay tree for DiffServ domains. The algorithm determines an appropriate location for inserting a new joining node while considering the QoS constraints. With the network QoS support, it is easier to build and maintain QoS-aware overlay trees. The simulation results depict that the proposed scheme has a balanced performance for various performance metrics analyzed in this chapter.

CHAPTER 6 CONCLUDING REMARKS

6.1 Conclusions

The core stateless architecture of internet is the gaining momentum of its fast growing. While it is not difficult to come up with a perfect per-flow QoS proposal or an efficient multicast protocol in a small area network, significant efforts are still in demand to make them scalable. The problem gets more complicated for developing a scalable framework to provision QoS for multicast service.

As scalability is the primary concern of the problem, the solution we are seeking should be simple, aggregatable and practical. To this end, we have suggested building multicast on top of the scalable QoS frameworks, specifically DiffServ and MPLS Traffic Engineering. The advantages of our approach including reduced computational complexity, layered structure and simplified implementation.

Throughout the dissertation, the idea of 'edge-based' is gradually evolved. In DAM, we count multicast flows as multiple unicast flows at the edge of a DiffServ domain. Next, in ERM, we only grant edge routers the ability to be multicasting capable. Finally, we propose a tree-building algorithm IIA and apply to the recently proposed overlay multicast schemes. This trend clearly reflects how our goal, scalable multicast QoS proposals, are achieved.

Trade-offs have to be made when designing each scheme. The major performance metrics that we concern are overall cost, link stress, relative delay and protocol overheads. While native IP multicast techniques can produce the best

results in overall cost, link stress as well as relative delay, their state-full design imposes tremendous overheads on the networks and makes them unable to scale well. On the other hand, although having great potential to be scalable, stateless multicast or core stateless multicast cannot be as efficient as IP multicast. Therefore, the key question is to provide stateless, aggregatable multicast techniques with performance close to IP multicast. We conduct a series of studies, which include DAM, ERM and IIA. Our results indicate that our proposals have achieved a balanced performance.

6.2 Future Research

The paradigms proposed in this dissertation is scalable from QoS point of view, because the multicast traffic in our designs are aggregatable. However, in terms of multicast group organization and management, the research is mainly focused on intra-domain cases. Further research should be conducted to build well-performed, scalable inter-domain multicast on top of DiffServ or MPLS-TE.

Another interesting work that could be done to is to compare three proposals presented in this dissertation. Due to time constraints, we have only measured each of the schemes with its corresponding related works. A comparison of Edge-Router-only multicast and overlay multicast may help us further understand the strength and 'sweet-spot' of multicast.

BIBLIOGRAPHY

- [1] P. Ferguson, and G. Huston, *Delivering QoS on the Internet and in Corporate Networks*, John Wiley & Sons Inc., 1998.
- [2] S. Deering, *Multicast Routing in Internetworks and Extended LANs*, Proceedings of ACM SIGCOMM, pp 55-64, Aug 1988.
- [3] S. Deering, *Host Extensions for IP Multicasting*, RFC1112, Aug 1989.
- [4] L. Berger, D. Gan, G. Swallow, P. Pan, F. Tommasi, S. Molendini, *RSVP Refresh Overhead Reduction Extensions*, RFC 2961, Apr 2000.
- [5] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, J. McManus, *Requirements for Traffic Engineering over MPLS*, RFC 2702, Sep 1999.
- [6] E. Rosen, A. Viswanathan, R. Callon, *Multiprotocol Label Switching Architecture*, RFC 3031, Jan 2001.
- [7] V.P.Kompella, J.C.Pasquale, G.C.Polyzos, *Multicast Routing for Multimedia Communication*, IEEE ACM Transaction on networking, Jun 1993.
- [8] Q.Zhu, M.Parsa, J.J. Garcia-Luna-Aceves, *A Source Based Algorithm for Delay-Constrained Minimum Cost Multicasting*, Proc. IEEE INFOCOM 95, Boston, MA, Apr 1995.
- [9] K.Carlberg, J. Crowcroft, *Building Shared Trees Using a One-to-Many Joining Mechanism*, computer Communication Review, pp.5-11, Jan 1997.
- [10] K. Kar, M. Kodialam, T. V.Lakshman, *Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications* IEEE Journal on Selected areas in Communications, pp2566-2579, Vol. 18, Dec 2000.
- [11] H. Saito, Y.Miyao, M.Yoshida, *Traffic Engineering using Multiple Multipoint-to-Point LSPs* IEEE INFOCOM, 2000.
- [12] D. Kosiur, *IP Multicasting: The complete guide to interactive corporate networks*, John Wiley&Sons Inc., 1998.
- [13] D. Thaler and M. Handley, *On the Aggregatability of Multicast forwarding state*, IEEE INFOCOM, 2000.
- [14] K. Nicholds, V. Jacobson and L. Zhang, *A Two-bit Differentiated Services Architecture for the Internet*, work in progress.
- [15] K. Carlberg and J. Crowcrof, *Building shared trees using a one-to-many join mechanism*, ACM Computer Communication Review, pp 5-11, Jan 1997.
- [16] M. Faloutsos, A. Banerjea, and R. Pankaj, *QoSMIC: Quality of Service sensitive Multicast Internet Protocol*, SIGCOMM, pp 144-153, Sep 1998.

- [17] S.Chen, K.Nahrstedt, and Y.Shavitt, *A QoS_Aware Multicast Routing Protocol*, IEEE Journal on Special Areas in Communication, pp 1594-1603, 2000.
- [18] B. Wang and J. Hou, *Multicast Routing and its QoS Extension: Problems, Algorithms and Protocols*, IEEE Network, pp 22-36, Jan/Feb, 2000.
- [19] J. Wroclawski, *The Use of RSVP with IETF Integrated Services*, RFC 2210.
- [20] K. Fujikawa and I. Sheng, *Simple Resource ReSerVation Protocol (SRSVP)*, work in progress.
- [21] R. Bless, and K. Wehrle, *IP Multicast in Differentiated Services networks*, work in progress.
- [22] R. Bless, K. Wehrle, *A Lower Than Best-Effort Per-Hop Behavior*, work in progress.
- [23] R. Cohen and G. Kaempfer, *A Unicast-based Approach for Streaming Multicast*, IEEE INFOCOM 2001.
- [24] P. Marbach, *Pricing Differentiated Services Networks: Bursty Traffic* IEEE INFOCOM 2001.
- [25] R. Neilson, J. Wheeler, F. Reichmeyer and S. Hares, *A Discussion of Bandwidth Broker Requirements for internet2 Qbone Deployment, version0.7*, Qbone draft, URL: <http://www.merit.edu/>.
- [26] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, *An Architecture for Differentiated Services*, RFC 2475, IETF, Dec 1998.
- [27] Network Simulator – NS (version 2), <http://www-mash.cs.berkeley.edu/ns/>.
- [28] P. Vaanane and R. Ravikanth, *Framework for Traffic Management in MPLS Networks*, work in progress.
- [29] S. McCanne, V. Jacobson, and M. Vetterli, *Receiver-driven layered multicast*, ACM SIGCOMM, pp 200-208, Aug 1996.
- [30] A.Terzis, L. Wang, J. Ogawa, L. Zhang, *A two-tier resource management model for the Internet* GLOBECOM '99 , p.p. 1779-1791, vol.3, 1999.
- [31] S. Floyd and V. Jacobson *Random Early Detection Gateways for Congestion Avoidance* IEEE/ACM Transactions on Networking, p.p. 397–413, 1993.
- [32] M. Bilmer, T. Braun, *Evaluation of bandwidth broker signaling* ICNP Proceedings, p.p. 145-152, 1999.
- [33] Y. Chu, S.G. Rao, H. Zhang, *A Case For End System Multicast* pp 1-12, Proceedings of ACM SIGMETRICS, CA, Jun 2000.
- [34] S. Ratnasamy, Y. Chawathe, S. McCanne, *A Deliverybased Model for Multicast Protocols using ScatterCast*, work in progress, Jul 1999.

- [35] S. Deering, *Multicast Routing in Internetworks and Extended LANs*, Proceedings of ACM SIGCOMM, pp 55-64, Aug 1988.
- [36] E. Rosen, A. Viswanathan, and R. Callon, *Multiprotocol Label Switching Architecture*, RFC 3031, Jan. 2001.
- [37] L. Andersson, P. Doolan, N. Feldman, A. Fredette, B. Thomas *LDP Specification* RFC3036, Jan 2001.
- [38] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow, *RSVP-TE: Extensions to RSVP for LSP Tunnels*, work in progress, <http://www.ietf.org/internet-drafts/draft-ietf-mpls-rsvp-slp-tunnel-09.txt>.
- [39] C. Diot, B. N. Levine, B. Lyles, H. Kassem and D. Balensiefen, *Deployment Issues for the IP Multicast Service and Architecture* IEEE Network, pp. 78-88, Jan/Feb 2000.
- [40] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, *Requirements for Traffic Engineering Over MPLS*, RFC 2702, Sep. 1999.
- [41] K. Kar, M. Kodialam, and T.V.Lakshman, *Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications*, IEEE Journal on selected areas in communications, pp. 2566-2579, Vol18, No.12, December 2000.
- [42] T. Li, and Y. Rekhter, *Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)* RFC 2430, Oct. 1998.
- [43] P. Aukia, M. Kodialam, P. Koppol, T. Lashman, H. Sarin, and B. Suter *RATES: A server for MPLS Traffic Engineering* IEEE Network Magazine, pp. 34-41, Mar./Apr. 2000.
- [44] A. Elwalid, C. Jin, S. Low, and I. Wkdjaja, *MATE: MPLS Adaptive Traffic Engineering*, IEEE Infocom, 2001.
- [45] D. Ooms, B. Sales, W. Livens, A. Acharya, F. Griffoul, and F. Ansari, *Framework for IP Multicast in MPLS*, work in progress, <http://www.ietf.org/internet-drafts/draft-ietf-mpls-multicast-07.txt>.
- [46] A. Boudani and B. Cousin, *An Effective Solution for Multicast Scalability: The MPLS Multicast Tree (MMT)*, work in progress, <http://search.ietf.org/internet-drafts/draft-boudani-mpls-multicast-tree-00.txt>.
- [47] H. Saito, Y. Miyao, and M. Yoshida, *Traffic Engineering using Multiple Multipoint-to-Point LSPs*, IEEE Infocom, 2000.
- [48] D. Thaler, and M. Handley, *On the Aggregatability of Multicast Forwarding State*, IEEE infocom, 2000.
- [49] P. Francis, *Yoid: Extending the Internet Architecture*, Work in progress, Apr 2000, <http://www.yallcast.com>.

- [50] D. Meyer, and B. Fenner, *Multicast Source Discovery Protocol (MSDP)*, work in progress, <http://www.ietf.org/internet-drafts/draft-ietf-msdp-spec-12.txt>. Sep. 2001.
- [51] T. Bates, R. Chandra, D. Katz, and Y. Rekhter, *Multiprotocol Extensions for BGP-4* RFC 2283, Feb. 1998.
- [52] D. Thaler, C. Alaettinoglu, D. Estrin, and M. Handley, *The MASC/BGMP architecture for Interdomain Multicast Routing*, pp. 93-104, ACM SIGCOMM, Sep. 1998.
- [53] S. Deering, C. Partridge, and D. Waitzman, *Distance Vector Multicast Routing Protocol* RFC 1075, Nov. 1988.
- [54] J. Moy, *Multicast Extensions to OSPF* RFC 1584, Mar. 1994.
- [55] A. Ballardie, *Core Based Trees (CBT) Multicast Routing Architecture* RFC 2201, Sep. 1997
- [56] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, A. Helmy, D. Meyer, and L. Wei, *Protocol Independent Multicast Version 2 Dense Mode Specification* work in progress, <http://www.ietf.org/proceedings/99nov/I-D/draft-ietf-pim-v2-dm-03.txt>, Jun. 1997.
- [57] 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, Jun. 1998.
- [58] J. Wroclawski *The Use of RSVP with IETF Integrated Services* RFC2210, September 1997.
- [59] S. Blake et al. *An Architecture for Differentiated Services* RFC2475, December 1998.
- [60] K. Calvert, M. Doar and E. W. Zegura, *Modeling Internet Topology*, IEEE Communications Magazine, pp. 160-163, June 1997.
- [61] Y. Chu, S. G. Rao and H. Zhang, *A Case For End System Multicast*, Proceedings of ACM SIGMETRICS, pp. 1-12, June 2000,
- [62] Y. Chu, S. G. Rao, S. Seshan and H. Zhang, *Enabling Conferencing Applications on the Internet using an Overlay Multicast Architecture* Proceedings of ACM SIGCOMM, August 2001.
- [63] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, *ALMI: An Application Level Multicast Infrastructure*, 3rd USENIX Symposium on Internet Technologies, pp. 49-60, March, 2001.
- [64] J. Jannotti, D. K. Gifford, and K. L. Johnson, et al., *Overcast: Reliable Multicasting with an Overlay Network*, Proceedings of 4th USENIX OSDI, pp. 197-212, October 2000.

- [65] D. A. Helder and S. Jamin, *End-host Multicast Communication Using Switch-tree Protocols*, Proceedings of GP2PC on Large Scale Distributed Systems, May 2002.
- [66] S. Jain, R. Mahajan, D. Wetherall and G. Borriello *Scalable Self-Organizing Overlays*, Submitted for review.
- [67] S. Banerjee, B. Bhattacharjee and S. Parthasarathy, *A Protocol for Scalable Application Layer Multicast*, CS-TR 4278, Department of Computer Science, University of Maryland, College Park, July 2001
- [68] QBone Signaling Design Team, *Final Report* <http://qbone.internet2.edu/bb/>

MICHIGAN STATE UNIVERSITY LIBRARIES



3 1293 02327 0865