A Scalable Overlay Multicast Architecture for Large-Scale Applications

Li Lao[1], Jun-Hong Cui[2], Mario Gerla[1]

llao@cs.ucla.edu, jcui@cse.uconn.edu, gerla@cs.ucla.edu

[1] Computer Science Department, University of California, Los Angeles, CA 90095

[2] Computer Science & Engineering Department, University of Connecticut, Storrs, CT 06029

Technical Report TR040008

Last Update: July 2004

Computer Science Department

UCLA

Abstract

We propose a two-tier overlay multicast architecture (TOMA) to provide scalable and efficient multicast support for a variety of group communication applications. In TOMA, multicast service overlay network (MSON) is advocated as the backbone service domain, while end users in the access domains form a number of small clusters, in which an application-layer multicast protocol is used for the communication between the clustered end users. Our two-tier architecture is able to provide efficient resource utilization with less control overhead, especially for large-scale applications. It also alleviates the forwarding state scalability problem and simplifies multicast tree construction and maintenance when there are large numbers of groups ongoing in the networks. To help the MSON provider efficiently plan backbone service overlay, we provide several dimensioning algorithms to locate proxies, select overlay links, and allocate link bandwidth. Based on our architecture, we also suggest a cost-based pricing model for the overlay ISP to charge multicast groups. This pricing model would provide key incentives for both service providers and clients to adopt our proposed TOMA service. Extensive simulation studies are conducted and the results demonstrate that TOMA performs well in several common scenarios, it provides efficient multicast transmission comparable to IP multicast, and is scalable to group size as well as to the number of co-existing groups. We also run experiments and show that our dimensioning algorithms could efficiently plan the network resources with little penalty. We believe that the invention of our practical, comprehensive, and profitable multicast service model would significantly facilitate the multicast wide deployment, making multicast service overlay from myth to reality.

1 Introduction

Over the years, there have been tremendous efforts to provide multicast (or group communication) support, ranging from IP multicast to recently proposed application-layer multicast. IP multicast utilizes a tree delivery structure which makes it fast, resource efficient and scale well to support very large multicast groups. However, even after approximately two decades since the inception of IP multicast, it is still far from being widely deployed on the Internet. This is due to many technical reasons as well as marketing reasons [13]. The most critical ones include: the lack of a scalable inter-domain routing protocol, the state scalability issue with a large number of groups, the lack of support in access control, the requirement of global deployment of multicast-capable IP routers and the lack of appropriate pricing models, as make Internet Service Providers (ISPs) reluctant to deploy and provide multicast service.

In recent years, researchers resort to application-layer multicast approach: multicast-related features are implemented at end hosts. Data packets are transmitted between end hosts via unicast, and are replicated at end hosts. Examples are End System Multicast [10],Yoid [17], ALMI [26], and NICE [3], to name a few. These systems do not require infrastructure support from intermediate nodes (such as routers), and thus can be easily deployed. However, application-layer multicast is generally not scalable to support large multicast groups due to its relatively low bandwidth efficiency and heavy control overhead caused by tree maintenance at end hosts. In addition, from the point of view of an ISP, this approach is hard for it to have an effective service model to make profit: group membership and multicast trees are totally managed at end hosts, thus it is difficult for the ISP to have the member access control and the knowledge of the bandwidth usage of the group, as makes an appropriate pricing model impractical, if not impossible.

As can be observed, there are more and more emerging group communication applications in the Internet, such as video conferencing, video on-demand, network games, distributed interactive simulation (DIS), etc. And it is also well known that multicast is more resource efficient than multiple unicast. In a multicast service, multiple parties are involved: network service providers (or higher-tier ISPs), Internet Service Providers (or lower-tier ISPs, which we refer to as ISPs for short in this paper), and end users. However, who really care about multicast? End users do not care as long as they can get their service at reasonable price. This is why many network games are implemented in unicast. Network service providers do not care as long as they can sell their connectivity service with good price. This actually contributes as one reason for the delay of IP multicast deployment. Obviously, ISPs in the middle are the ones who really care about multicast: their goal is to use limited bandwidth purchased from network service providers to support as many users as possible, i.e., make the biggest profit. Therefore, to develop a practical, comprehensive, and profitable multicast service model for ISPs is the critical path to multicast wide deployment.

Based on this strong motivation, in this paper, we propose a **T**wo-tier **O**verlay **M**ulticast **A**rchitecture (called **TOMA**) to provide scalable, efficient, and practical multicast support for a variety of group communication applications. In this architecture, we advocate the notion of **M**ulticast **S**ervice **O**verlay **N**etwork (referred to as **MSON**) as the backbone service domain. MSON consists of service nodes or proxies which are strategically deployed by the MSON provider (ISP). The design of MSON relies on well defined business relationships between the MSON provider, the network service providers (i.e., the underlying network domains), and the group coordinators (or initiators): the MSON provider dimensions its overlay network according to end user requests (provided by long-term measurement), purchases bandwidth from the network service providers based on service level agreement (SLA), and sells its multicast services to group coordinators via a service contract [1]. Outside MSON, end hosts (group members) subscribe to MSON by transparently connecting to some special proxies (called *member proxies*) advertised by the MSON provider. Instead of communicating with its member proxy using unicast, an end host could form a cluster with other end hosts close by. In the cluster, application-layer multicast is used for efficient data delivery between the limited number of end users. And the end users participating in the groups only need to pay regular network connection service for the communication outside MSON.

Our TOMA architecture not only provides scalable and efficient multicast support as well as a practical pricing model for ISPs, it also brings many advantages. First, an MSON provider can support a variety of group communication applications simultaneously, unlike some other existing multicast overlays (such as [10], [17], [7], and [19] etc.), with each overlay only supporting one group. This provides an additional incentive for ISPs to adopt TOMA. Second, it is relatively easy for ISPs to manage resources since MSON is based on well-defined business relationships via SLA with network service providers and service contracts with group coordinators. They can put major efforts on planning and managing their overlay networks. Third, the notion of MSON will significantly simplify the management of underlying networks: network service providers only need to provide services to limited numbers of MSON providers instead of millions or billions of individual

[1] How a group coordinator shares the cost with the end users (group members) is up to their inside negotiation. We suggest some approaches in Section 4

end users. This level of traffic aggregation, in the long run, will make IntServ practical[2]. A good analogy to this scenario is the relationship between manufacturers, dealers, and consumers in the real life. Lastly, MSON can be further extended to support virtual group services, such as web content distribution. Web clients who share similar interests can form a virtual group managed by MSON. And the data transmission inside MSON can be reorganized in order to provide better service.

To make multicast service overlay networks (MSON) a reality, we face many challenges:

- **Efficient management of MSON**: It is anticipated that an MSON will accommodate a large number of multicast groups. How does an MSON provider efficiently establish and manage numerous multicast trees?
- **Cluster formation outside MSON**: End users of multicast groups might disperse around the country (or even the world). They first need to subscribe to MSON and connect to some member proxies. How should appropriate member proxies be selected? And how are efficient clusters formed among end users?
- **MSON dimensioning**: Given that the MSON provider has an overlay architecture, how should it dimension the overlay network? In other words, where should the overlay proxies be placed, which links are used to transmit data, and how much bandwidth on each link should be reserved?
- **Pricing Model**: The lack of mechanisms to measure resource usage and bill multicast users is one of the main forces that delays the deployment of IP multicast. Therefore, how to charge the multicast group users of MSON is an important factor to decide whether MSON is a practical solution.

In this paper, we address all these issues. We propose the TOMA architecture. To solve the state scalability issue when there are a large number of groups, in MSON we adopt *aggregated multicast* approach [16], with multiple groups sharing one delivery tree. Outside MSON, we develop efficient member proxy selection mechanisms, and choose a core-based application-layer multicast routing approach for data transmission in clusters. Besides, we provide effective algorithms to dimension overlay networks: locating overlay proxies, identifying overlay links, and dimensioning bandwidth. We also suggest a cost-based pricing model for the overlay ISPs who adopt the TOMA architecture to charge multicast users. We believe this pricing model provides incentives for both service providers and clients to adopt our multicast service scheme. Furthermore, we conduct extensive simulation studies and show the promising performance of TOMA as well as the effectiveness of our dimensioning algorithms.

The rest of this paper is organized as follows. In Section 2, we present our TOMA architecture and discuss some related issues. In Section 3, we describe several algorithms for overlay network dimensioning. In Section 4, we propose a pricing model to solve billing issues of multicast service. Then, in Section 5, by simulations we evaluate the performance of TOMA and the overlay dimensioning algorithms. Finally, we review some related work in Section 6 followed by a short summary in Section 7.

2 TOMA: A Two-Tier Overlay Multicast Architecture

We design a two-tier overlay multicast architecture (TOMA) to provide scalable, efficient and practical multicast services to end users. In this section, we present our proposed architecture.

2.1 Overview

In the TOMA architecture, MSON is advocated as the service backbone domain. In MSON, overlay proxies are strategically placed to form an overlay network, on top of which multicast distribution trees are built for data delivery. To reduce the

[2]From this aspect, we share a similar vision with SON [14], a service overlay proposed for end-to-end QoS support.

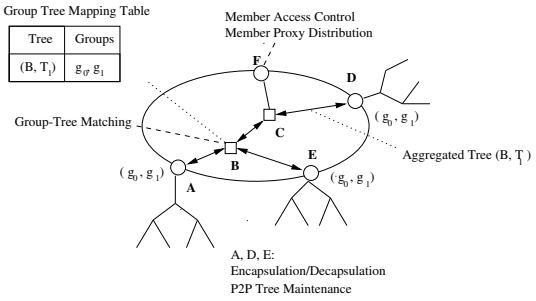

Figure 1: A big picture of TOMA, where F is the group registry server/DNS server, B is the host proxy for groups g_0 and g_1, A, D and E are member proxies, groups g_0 and g_1 share the aggregated tree, denoted by (B, T_1).

management overhead of a large number of trees and improve the multicast state scalability, we employ aggregated multicast approach [16], in which multiple groups are forced to share one delivery tree (called *aggregated trees*)[3]. Data packets are encapsulated at incoming proxies, transmitted on aggregated trees, and decapsulated at outgoing proxies. Outside MSON, end users subscribe to MSON by transparently connecting to some special proxies (called **member proxies**) advertised by the MSON provider. Each member proxy organizes some end users into a "cluster", where an application-layer multicast tree (also denoted as peer-to-peer or P2P multicast tree in this paper) is formed for data delivery among the cluster members.

Each TOMA group is identified by a URL-like unique name. Group names have the form of *TOMA://groupname.xxxmson.com/* [4]. End users (sources and receivers) explicitly subscribe to a group g, by sending out a join request containing the URL-like group name. Through DNS, this request will reach the DNS server of the MSON, in which there resides a **group registry server**. It is the responsibility of the group coordinator (or initiator) to register the group in the group registry server. In other words, the group registry server maintains the member information of each group, thus it is easy for the MSON to implement its member access control policy. After receiving a join request, the DNS server will send back to the subscriber a list of IP addresses of the advertised member proxies, from which an appropriate member proxy will be selected (the details are explained in Section 2.2).

After finding a member proxy, the end user sends its join request to the member proxy, which will then subscribe to the multicast group inside the MSON on behalf of this member. In addition, the member proxy will set up a peer-to-peer multicast tree in the local cluster and relay the join request to a predetermined proxy (referred to as **host proxy** for group g) to establish the overlay multicast tree in the backbone domain. Therefore, the member proxies participate in multicast data forwarding of both the backbone domain and user access domains (outside MSON).

In the backbone domain, each group is managed by a host proxy. After receiving a join request for group g relayed by a member proxy, this host proxy conducts multicast routing and group-tree matching mechanisms to map group g to an aggregated tree. The host proxy for group g can be randomly selected or by applying a hash function (called *group-to-host* hashing function).

In a nutshell, in TOMA, member proxies control member access policy and manage peer-to-peer multicast trees in its cluster, and host proxies conduct group-tree matching and manage aggregated multicast trees in the backbone domain. There are some **forwarding proxies** that are neither member proxies nor host proxies, and their main responsibility is to forward multicast data inside the backbone domain. A big picture of TOMA is illustrated in Fig. 1. Note that, each aggregated tree is addressed by a pair of a host proxy ID and a tree ID.

[3]In aggregated multicast, we need a procedure to assign proper aggregated trees to groups in order to control bandwidth waste due to data delivery to non-member nodes when trees are bigger than groups. We refer to this procedure as *group-tree matching.*

[4]The URL-like naming approach has been adopted by many systems, such as CDN (content delivery networks), Yoid [17], Scattercast [7], Overcast [19], etc.

Having discussed the main components (forwarding, member and host proxies) of TOMA and their functionalities, we describe its main design issues in more details as follows.

2.2 Member Proxy Selection

On receiving a list of candidate member proxies from the MSON DNS server, an end user selects one proxy based on the criteria of low latency and low workload (in terms of processing power and bandwidth availability), since low latency is critical to real-time applications and lightly-loaded proxies tend to have more resources.

The end user measures the RTT (round-trip time) to eligible proxies by sending *ping* requests. In the reply message to a *ping* request, the proxy piggybacks its workload information, e.g., the total number of end users it currently handles or the total amount of access bandwidth in use. The end user then discretizes the measured RTT values into pre-determined levels. This discretization approach improves the stability of RTT values and allows other metrics to be considered when the proxies have similar RTTs. If there are multiple proxies close by, the proxy with the lowest workload is preferred in that the member proxies can be load-balanced.

This selection scheme allows an end user to choose a good proxy; however, it induces a join latency of at least one RTT between the the end user and the proxies. Alternatively, the new group member can initially subscribe to a randomly selected proxy. If the performance with this temporary proxy node is not satisfactory, the user can probe other member proxies while continues to receive data via the current one. Once it finds a better proxy, it unsubscribes from the old one and subscribes to the new one. To avoid an end user oscillating between proxies with similar distances, the end user is allowed to switch proxy only if the RTT to the new proxy can be upgraded to a higher level.

2.3 Member Join and Leave

Once finding an appropriate member proxy mp, an end host sends to mp a join request, which will be forwarded to the host proxy hp for group g. After conducting the group-tree matching, the host proxy hp will find an appropriate aggregated tree, say, T for the group and send this group-tree mapping information in an ACK message to mp. If mp has not joined the delivery tree T, it will graft to the tree by sending graft messages towards the host proxy. At the same time, mp needs to construct or update the peer-to-peer multicast tree for this group in its local cluster and instruct the new member to connect to its parent (we will discuss P2P multicast tree construction protocol in Section 2.5). From now on, this member will start to receive packets for g relayed by mp.

When an end host leaves group g, the member proxy mp may need to adjust the local peer-to-peer multicast tree and distribute the tree change to the rest of the cluster. If no more receivers are attached to mp, mp propagates the leave message to the host proxy hp in order to prune from the tree. The host proxy conducts another group-tree matching process, which may trigger removal of the tree T when there are no other groups mapped onto T. To handle the situation when an end host leaves the group ungracefully, the member proxies should exchange "heartbeat" messages periodically with their members and follow the same leave procedure described above once an ungraceful leave is detected.

2.4 Multicasting in MSON

When a host proxy receives a join request for a multicast group from a member proxy, it first looks up a multicast tree for the group. Here we present a simple group-tree matching algorithm with a small amount of additional control overhead.

As mentioned earlier, when an aggregated tree is bigger than a group, data will be delivered to non-member nodes, as leads to bandwidth waste. Obviously, there is a trade-off between bandwidth waste and aggregation: the more bandwidth we are willing to sacrifice, the more groups can share one tree, and thus the better aggregation we can achieve. Hence, it is

necessary to control the amount of bandwidth waste in group-tree matching. Assume that an aggregated tree T is shared by groups $g_i, 1 \leq i \leq n$, each of which has a native tree $T_0(g_i)$ (a native tree of a group is a "perfect" match for that group and it can be computed using multicast routing algorithms). Then the **average percentage bandwidth overhead** for T can be defined as

$$\begin{aligned} \delta(T) &= \frac{\sum_{i=1}^{n} B(g_i) \times (C(T) - C(T_0(g_i)))}{\sum_{i=1}^{n} B(g_i) \times C(T_0(g_i))} \\ &= \frac{C(T) \times \sum_{i=1}^{n} B(g_i)}{\sum_{i=1}^{n} B(g_i) \times C(T_0(g_i))} - 1, \end{aligned} \tag{1}$$

where $C(T)$ is the cost of tree T (i.e., the total cost of the links on tree T), and $B(g)$ is the bandwidth requirement of group g.

A group-tree matching algorithm similar to the one proposed in Aggregated QoS Multicast (AQoSM) [12] can be used to map a group to an aggregated tree, and set up or remove trees accordingly. The basic idea is as follows: the host proxy attempts to map the group to an existing aggregated tree if the tree can cover all group members and the estimated bandwidth waste is less than a pre-defined **bandwidth waste threshold** b_{th}. If this fails, it uses multicast routing protocol to compute the native multicast tree for this group as the aggregated tree and set up this new tree. The updated group-tree mapping is propagated to all the involved member proxies.

2.5 P2P Multicast outside MSON

Outside MSON, the end users associated with the same member proxy form a cluster. In each cluster, nodes communicate in a P2P fashion via application-layer multicast trees: when an end user sends packets to the group, the packets are transmitted to all other end users in the same cluster and to the member proxy as well. The member proxy will relay these packets to other member proxies (at the edge of the MSON), which will in turn deliver the data to the group members in their clusters.

In the literature, there are many application-layer multicast protocols. In TOMA, due to the existence of a member proxy node in every cluster, we adopt a core-based approach, which is similar to ALMI [26], to construct P2P multicast trees. In each cluster, the member proxy acts as a core, storing the group membership information in its cluster scope. Periodically, every end user monitors other users in the same cluster regarding the path quality (such as delay and available bandwidth) to each of them, and reports this information to its member proxy. Once the member proxy puts together the global picture of its clusters, it calculates appropriate P2P multicast delivery trees and distributes the $(parent, child)$ relationship to the end users. Finally, end users connect with their children and transmit data packets via unicast connections.

Since the member proxy nodes periodically compute new multicast trees, the clusters are able to maintain high-quality multicast trees in the presence of group dynamics and transient failure of links or nodes. If a member leaves ungracefully or if a path between two neighbors is broken, these events can be detected from periodic probing and the multicast tree will be repaired by the member proxy. Due to the limited size of each cluster, the control message overhead is likely to be very small.

2.6 Discussions

In this section, we discuss some additional features that can be employed by the MSON providers to improve the overlay management and services.

Member and Source Access Control One critical problem in traditional IP multicast is the lack of effective access control mechanisms. In TOMA, access control can be managed by the group registry server. Since every end user has to contact the MSON DNS server and the group registry server to obtain the list of candidate proxies, the group registry server is responsible for authentication and authorization check. Only after the user passes the check, the server sends a digital signature including a timestamp and the user's address, which is validated by member proxy to authorize user subscription.

For source access control, the delivery of data packets along the application-layer multicast trees and authentication between (parent, child) pairs are sufficient to ensure that only group members are allowed to send data packets.

Fault Recovery of Multicast Trees Since the MSON provider purchases bandwidth from higher tier ISPs and the overlay proxy nodes are specially designed servers, it is quite unlikely that proxy and link failures occur very often. However, to ensure fast recovery, a pre-planned restoration approach [] can be adopted. When a new multicast tree is established, the host proxy computes a backup tree by using some redundant tree fault-tolerance schemes, such as the algorithm described in [24, 15]. When a proxy or link failure occurs, the neighbor proxies will detect the failure from the lack of "heartbeat" message exchange and broadcast this discovery to all other proxy nodes. Then the host proxies determine the affected aggregated trees, retrieve their backup trees, and switch the related multicast groups to the backup trees. The employment of aggregated multicast greatly facilitates the fault tolerance of TOMA, because by reducing the number of aggregated trees, the backup tree computation and maintenance cost, and control overhead involved in failure recovery is significantly reduced.

Resilience of Member and Host Proxies The special functionalities of member proxies and host proxies require their failure to be separately handled. If a member proxy dies, the end hosts are able to detect the failure and will contact the MSON DNS server and the group registry server to re-join the group. In case of a host proxy failure, a new host proxy is chosen from the backup proxy list, and then member proxies will switch to this new host proxy, which will conduct group-tree matching and assign new multicast trees.

Load Balancing Frequently, some overlay links may become congested when there are a large number of bandwidth-demanding multicast groups in the overlay network. Therefore, the overlay proxies need to monitor the current congestion conditions on its adjacent links and report overloaded link back to host proxies. The host proxies will try to bypass those congested links when executing the group-tree matching procedure.

The member proxies can also be overloaded by excessive join requests from clients. Fortunately, the member proxy selection mechanism described earlier helps to distribute load evenly on member proxies, since users use proxy workload as one criteria to choose a good proxy.

Heterogeneity Handling For high-bandwidth applications such as video-streaming, the inherent heterogeneity of current Internet has made multicast a challenging problem, since there is no single rate that can fit the demand of receivers with different bandwidth and processing capabilities. A promising solution which divides the group members into a number of homogeneous sub-groups has been adopted in existing overlay multicast architectures [7, 8]. Even though this approach solves user heterogeneity problem, it exacerbates the state scalability problem, because multiple multicast trees are now needed for each multicast group. TOMA, on the other hand, can be seamlessly integrated with this approach without significantly increasing the number of aggregated trees to manage.

3 Overlay Network Dimensioning

In previous sections, we describe TOMA assuming that MSON is already constructed by the ISP: proxies are strategically deployed and overlay link bandwidth is purchased. But we still miss one important component: overlay network dimensioning. Obviously, the deployment of MSON is a capital-intensive investment. Thus, it is very imperative to carefully design MSON so that the ISP can make best revenue under given investment. In this section, we design algorithms for overlay network design by considering traffic pattern (group and member distribution), bandwidth requirement, and routing algorithms in MSON.

3.1 Problem Formulation

We model the physical network topology as an undirected graph $G = (V, E)$, where V and E denotes the the sets of network nodes (or routers) and physical links respectively. The overlay dimensioning problem can be formulated as follows: given a set of groups $\{g_i\}$ with group member distribution and bandwidth requirement, and a physical network topology $G = (V, E)$, find a virtual topology G' on top of G so that G' can accommodate all the groups, while keep the cost of G' minimum under the bandwidth waste threshold b_{th}. Here, we assume the group set $\{g_i\}$ are obtained from long-term measurement in the steady state of the network and group dynamics.

Clearly, to obtain G', the overlay ISP needs to make three decisions: 1) determine the locations of the proxy nodes; 2) identify the overlay connections between these proxies; 3) compute the bandwidth to be reserved on each overlay link. Thus, we divide the overlay network dimensioning problem into three sub-problems and present algorithms to solve them accordingly.

3.2 Overlay Proxy Placement

We mainly consider the member proxy placement since host proxies and forwarding proxies are usually decided strategically by the ISP due to their special functionalities. In fact, host proxies can be chosen from the set of member proxies, and it is not mandatary to have "pure" forwarding proxies: a member proxy for one group might act as a forwarding proxy for another group.

First of all, the locations of member proxies directly affect the latency of data transmission, since end users receive data packets from the closest member proxy. Intuitively, if a router is connected to a lot of users, a member proxy should be placed near this router to reduce the average delay. To quantify the group member density around a router, we can compute, for each router i, the number of directly attached end users participating in multicast groups, and we denote it as w_i. Due to the deployment cost of proxies, the total number of member proxies is likely to be bounded.

Thus, the member proxy placement becomes an optimization problem, which can be formulated as follows: given group member density w_i ($1 \leq i \leq M$) for all M routers and the shortest distance d_{ij} between every two routers i and j ($1 \leq i, j \leq M$), find no more than K ($1 \leq K \leq M$) routers as member proxies, such that the weighted sum of distances from each router to its nearest proxy is minimized.

We propose to use ILP (Integer Linear Programming) to solve this problem. Before presenting the ILP formulation, we define the following variables:

$$x_i = \begin{cases} 1, & \text{if router } i \text{ is selected as proxy} \\ 0, & \text{otherwise} \end{cases} \qquad \forall i \in [1, M]$$

and

$$y_{ij} = \begin{cases} 1, & \text{if router } j \text{ subscribes to proxy } i \\ 0, & \text{otherwise} \end{cases} \quad \forall i, j \in [1, M]$$

Here we use indicator variables y_{ij} in addition to x_i to overcome the difficulty of representing the closest proxy to a router, and our objective function guarantees that a router will subscribe to its closest proxy.

The objective is to minimize the total distance from a router to the closest proxy weighted by the router's member density:

$$\min \sum_{j=1}^{M} (\sum_{i=1}^{M} d_{ij} \times y_{ij}) \times w_j,$$

subject to the following constraints:

1) Every router subscribes to exactly one proxy:

$$\sum_{i=1}^{M} y_{ij} = 1 \qquad \forall j \in [1, M];$$

2) A router cannot subscribe to a non-proxy node:

$$y_{ij} \leq x_i \qquad \forall i, j \in [1, M];$$

3) Every proxy should have at least one router subscribed (this constraint is not enforced in the objective function since the goal is to minimize the total distance):

$$x_i \leq \sum_{j=1}^{M} y_{ij} \qquad \forall i \in [1, M];$$

4) No more than K nodes can be selected as proxy:

$$\sum_{i=1}^{M} x_i \leq K.$$

The complexity of this ILP formulation is $O(M^2)$ in terms of the number of variables and the number of constraints. After the ILP formulation is obtained, existing ILP solvers, such as lp_solve (an open source software) and CPLEX (a commercial mathematical optimization tool), can be used to solve for x_i and y_{ij}.

In the above formulation, we assumed that the maximum number of member proxies is pre-determined. Alternatively, the overlay ISP may want to minimize the sum of the cost to deliver a unit of data over each link and the proxy deployment cost. In this case, this problem is equivalent to the classical Warehouse Location Problem, which tries to determine the appropriate locations of warehouses to minimize the shipping cost and warehouse maintenance cost. Efficient branch-and-bound algorithms [20] and heuristic algorithms [21] can be adopted to solve this version of the member proxy location problem.

3.3 Overlay Link Identification

Once the location of the overlay proxies has been determined, the next step is to connect these proxies into a mesh, on top of which the overlay multicast trees will be constructed. We choose to use a so-called "Adjacent Connection" overlay topology proposed in [23] and [25]. In this approach, an overlay link is established between two overlay nodes if the shortest IP-layer path between them does not go through any other overlay nodes. As a result, the constructed overlay networks tend to resemble the underlying network topologies and thus have high routing efficiency, while at the same time provide redundant paths for potential load balancing and fault tolerance.

It is also possible to use other methods to identify the overlay links. For example, multicast trees are first computed according to given multicast routing algorithms and traffic patterns, and then highly used physical links are chosen (and combined) as overlay links (when there are no overlay nodes between two selected physical links). This approach could provide better usage of overlay links, but it is more tailored to routing algorithms and traffic patterns.

3.4 Bandwidth Dimensioning

Having determined the overlay topology, we need to identify the bandwidth required on overlay links to accommodate the groups obtained from measurements. As we mentioned earlier, MSON adopts aggregated multicast approach, thus we need to take the routing algorithm into account during bandwidth dimensioning. We use the group-tree matching algorithm presented

in Section 2.4 to compute the set of aggregated trees for the groups, given a bandwidth waste threshold. Then by summing up the group bandwidth requirements for each aggregated tree, the amount of bandwidth to be leased on every overlay link can be determined.

Based on long-term measurement results, the computed bandwidth is sufficient to satisfy bandwidth requirement of the groups on average. In reality, the traffic peak rate is likely to exceed the average rate frequently. Hence, the MSON providers should consider over-dimensioning the network by reserving a certain percentage of extra bandwidth. If the bandwidth is still insufficient during peak hours, MSONs can either reject some groups or lease bandwidth from higher-tier ISPs for short-term usage (possibly at a higher price).

4 Pricing Model

Pricing is one crucial issue that plagues IP multicast, in which group member information is distributed in routers that may reside in different domains, so there is no viable solution to estimate the network resource consumption of multicast communication. By contrast, in TOMA, group information is readily available from proxies, and thus can be used directly by ISPs to estimate resource usage inside MSON[5]. In this section, we propose a simple and efficient pricing model for the TOMA architecture.

The major cost of deploying TOMA services can be categorized into bandwidth cost, the cost to lease bandwidth from higher-tier ISPs, and equipment cost, the deployment and maintenance cost of the physical devices including storage, memory, CPU, etc. Therefore, in our model, the price for a multicast group is derived from these two costs.

Bandwidth Price Chuang-Sirbu Law states that the cost of a multicast tree varies at the 0.8 power of the multicast group size [11]:

$$\frac{L_m}{\overline{L_u}} = N_m^k \tag{2}$$

where L_m is the total number of links in the multicast distribution tree, $\overline{L_u}$ is the average number of links in a unicast path, N_m is the multicast group size, and k is a scaling factor of approximately 0.8.

Based on this result, we can derive a similar relationship between overlay multicast and overlay unicast. Recall that in TOMA, we use a threshold b_{th} to control the trade-off between bandwidth waste and tree aggregation. Assuming that each overlay link has the same cost, we can bound the relative bandwidth cost L_m of aggregated trees vs. average bandwidth cost $\overline{L_u}$ of overlay unicast paths by:

$$\frac{L_m}{\overline{L_u}} = (1 + b_{th}) \times N_p^k \tag{3}$$

where N_p is the number of member proxies participating in the multicast group. From this equation, it seems that the relative multicast efficiency, defined as $\frac{L_m}{N_P \overline{L_u}}$, may exceed 1 for high b_{th} and low N_p. This means multicast may not be as efficient as unicast, especially when a lot of bandwidth is wasted to achieve higher tree aggregation. To further investigate this issue, we plot the relative multicast efficiency with various N_p and b_{th} in Fig. 2. Fortunately, even when the number of member proxies is as small as 10, multicast efficiency is indeed smaller than 1 for reasonable values of b_{th} (i.e., $b_{th} < 0.5$). As the number of member proxies increases, multicast becomes more and more efficient than unicast.

If we know the price P_u charged for a group using overlay unicast, the price P_m charged by the MSON provider can be calculated as:

$$P_m = (1 + b_{th}) \times N_p^k \times \frac{P_u}{N_p} = (1 + b_{th}) \times N_p^{k-1} \times P_u \tag{4}$$

[5]We focus on the bandwidth usage inside MSON only, since we assume it is in end users' best interest to select appropriate Internet providers within their budget. In a further step, MSON providers can lease bandwidth outside MSON and provide end-to-end services, which is, however, out of the scope of this paper.

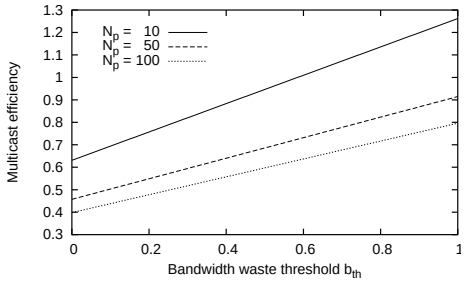

Figure 2: Relative multicast efficiency vs. bandwidth waste threshold b_{th} (assuming the scaling factor k takes a value of 0.8).

This approach is appealing in the sense that it does not require complicated network monitoring except the number of member proxies participating in the group. In practice, we can set a price in the range of [P_m, P_u]:

$$P_1 = P_m + \alpha \times (P_u - P_m) \qquad \alpha \in [0, 1] \tag{5}$$

Let us consider two extreme cases: if $\alpha = 1$, the multicast group is paying the same to the MSON provider as when there is only unicast support; on the other hand, if $\alpha = 0$, the price may only be able to compensate for the cost of the bandwidth purchased by the MSON ISP. Setting α in between creates a "win-win" situation in which both parties save money and thus have the incentives to support this service model.

Equipment Price Because the equipment price is not affected significantly by group size, a flat-rate price should suffice. For each multicast group, its equipment price can be determined as:

$$P_2 = \beta \times \frac{C_{eq}}{N_g} \qquad (\beta \geq 1) \tag{6}$$

where C_{eq} is the total cost of physical equipments, and N_g is the total number of multicast groups. Similar to α in P_1, β is used to control the net profit for P_2.

Finally, the total price of a multicast group is the sum of bandwidth price P_1 and equipment price P_2:

$$P_{total} = P_1 + P_2 \tag{7}$$

After the price is charged by the overlay ISP, the group coordinator can leverage existing approaches, such as Equal Tree Split, Equal Link Split among Downstream members, or Equal Next-Hop Split, to decide each receiver' share of the charge [18].

5 Performance Evaluation

In this section, we first conduct simulation experiments in NS-2 to evaluate the performance of TOMA. We compare TOMA with a scalable application-layer multicast protocol NICE [3], an IP multicast protocol (Core-Based Tree [2]), and unicast protocol in various network topologies. We find that TOMA can achieve very competitive performance for large groups with hundreds or even thousands of members. Then we present some simulation results on the effectiveness of our overlay dimensioning algorithms.

5.1 Simulation Settings

To comprehensively evaluate our architecture, we use two types of network topologies. The first set of network topologies are generated using the Transit-Stub Model developed by Institute of Georgia Technology [5]. These topologies have 50 transit

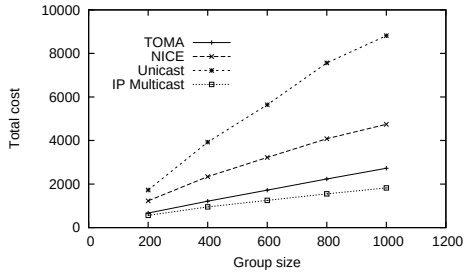

Figure 3: Tree cost vs. group size in Transit-Stub topology.

domain routers and 500-2000 stub domain routers. Each end host is attached to a stub router uniformly at random. To test the scalability of different schemes, we focus on large group sizes and vary the number of members in each group from 200 to 1000.

The second type of network topology is abstracted from a real network topology, AT&T IP backbone [1], which has a total of 123 nodes: 9 gateway routers, 9 backbone routers, 9 remote GSR (Gigabit Switch Routers) access routers, and 96 remote access routers. The abstract topology is constructed as follows: the attached remote access routers of a gateway or backbone router is "contracted" into one **contracted node**. In addition, we create a neighbor node called **exchange node** for each gateway router in the backbone, since gateway routers represents connectivity to other peering networks and/or Internet public exchange points. This abstraction procedure results in a simplified network of 54 nodes. Each end host are randomly assigned to a contracted node or exchange node according to *Random Node Weight Model*. In this model, each router is assigned a weight, which represents the probability that this router has attached end host(s) participating in a multicast group. Thus, for a group with fixed group size, the number of group members attached to a router is proportional to this router's weight. For the different routers in this abstracted network, gateway nodes and backbone nodes are assumed to be core routers only and are assigned weight 0. Each access router is assigned a weight of 0.01, and a contracted node's weight is the summation of the weights of all access routers from which it is contracted. Exchange nodes are assigned a weight of 0.9 since they usually connects to peering networks and tend to have large number of group members.

Due to the lack of real data on group member distribution, we choose to construct overlay network in a heuristic way. For the Transit-Stub topologies, we randomly select 80% of the transit nodes (i.e., 40 nodes) as proxy nodes. We repeat simulations with different sets of proxy nodes and take the average values. For the AT&T backbone topology, we select gateway routers (9 nodes) as proxies. The overlay links are constructed using the Adjacent Connection described in Section 3.

5.2 Multicast Tree Performance

We use the following metrics to compare the multicast tree performance. *Multicast tree cost* is measured by the number of links in a multicast distribution tree. It quantifies the efficiency of multicast routing schemes. Application level multicast trees and unicast paths may traverse an underlying link more than once, and thus they usually have a higher cost than IP multicast trees. To assess the quality of data paths, we measure link stress and path length when data are transmitted from a randomly selected source to all members. *Link Stress* is defined as the number of identical data packets delivered over each link. IP multicast trees has the least link stress since only a single copy of a data packet is sent over each link. *Path Length* is the number of links on the path from the source to a member. Unicast and shortest-path multicast schemes are usually optimized on this metric and thus have smallest path lengths.

In simulation experiments, end hosts join the multicast group during an interval of 400 seconds. Then we collect the metrics after the multicast tree has stabilized. We found that TOMA is able to converge within 10 seconds of simulation time after the join process is completed. In contrast, NICE tree needs hundreds of seconds of simulation time to stabilize.

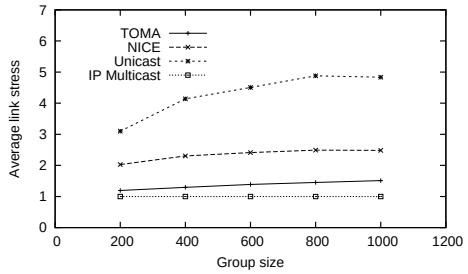

Figure 4: Average stress vs. group size in Transit-Stub topology.

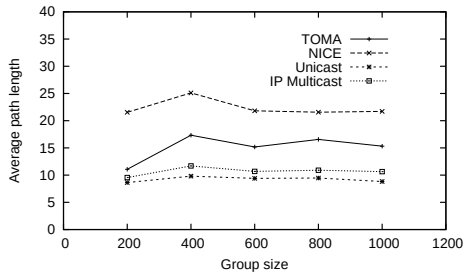

Figure 5: Average path length vs. group size in Transit-Stub topology.

Multicast tree cost in Transit-Stub topology We first use the Transit-Stub topologies to evaluate the performance of TOMA. In Fig. 3, we plot the average tree cost of TOMA, NICE, and CBT as group size increases from 200 to 1000. As a reference, we also include the total link cost for unicast. Compared with the cost of unicast paths, NICE trees reduce the cost by $30-46\%$, TOMA trees reduce the cost by approximately $61-70\%$, and CBT trees save the cost by $68-80\%$. Clearly, the performance of TOMA is comparable to IP multicast. In addition, TOMA outperforms NICE in all cases, and their difference magnifies as group size is increased.

Average link stress in Transit-Stub topology Fig. 4 shows the average link stress as the group size varies. IP multicast maintains a unit stress since no duplicate packets are transmitted on the same link. TOMA trees exhibit average link stress between 1.19 and 1.51, whereas the average link stress of NICE trees is always higher than 2.00. For both TOMA and NICE, the link stress does not vary greatly with group size. However, unicast is not as scalable as TOMA and NICE, since its link stress keeps increasing when group size grows.

Average path length in Transit-Stub topology The results for average path length are shown in Fig. 5. As expected, unicast and IP multicast have the shortest end-to-end paths. Additionally, the path lengths of TOMA trees are much shorter than those of NICE trees on average. For instance, at group size of 1000, the average path lengths of TOMA and NICE trees are 15.32 and 21.72, respectively.

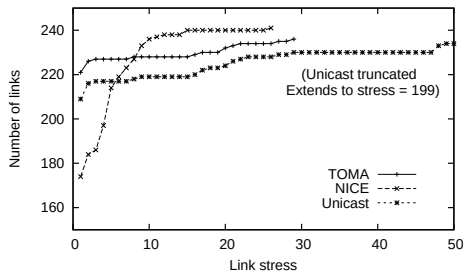

Figure 6: Stress distribution in AT&T topology.

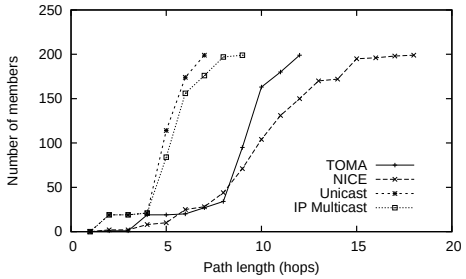

Figure 7: Path length distribution in AT&T topology.

Link stress distribution in AT&T topology We repeat the simulations for AT&T topology, and plot the cumulative distribution of stress and path length in Fig. 6 and 7, respectively, when group size is 200. As shown in Fig. 6, it is evident that TOMA and NICE are much more efficient than unicast, since their maximum link stresses (i.e., 29 for TOMA and 26 for NICE) are significantly smaller than that of unicast, which is 199. In addition, TOMA uses fewer number of links and has smaller link stress than NICE: in TOMA, 231 links have unit stress, and only 9 links have stress higher than 5; whereas in NICE, 174 links have unit stress, and 27 links have stress higher than 5.

Path length distribution in AT&T topology Fig. 7 confirms our earlier observation that the performance of TOMA is closer to unicast scheme than NICE in terms of path length. In TOMA, 81.9% members have paths to the source within 10 hops. On the other hand, in NICE, approximately half of the group members use paths between 10 to 18 hops.

5.3 Control Overhead

In this subsection, we conduct two sets of experiments to evaluate the control overhead incurred by TOMA. First, we compare the control overhead generated by TOMA and NICE for a single group when group size varies, since NICE is shown to have relatively low control overhead by organizing group members into a hierarchical overlay topology [3]. Second, we evaluate the effectiveness of aggregated multicast in reducing multicast tree setup and maintenance overhead when there are a large number of co-existing groups.

Control overhead for a single group in Transit-Stub topology In this set of experiments, end hosts join a multicast group during an interval of 400 seconds, and the session ends at 1000 seconds. We collect the total number of control messages transmitted during the group's life time. For the sake of a fair comparison, we set the parameters in NICE and TOMA to the same values whenever possible. We found out that in NICE, the total number of control messages increases very rapidly with group size, while in TOMA, the increase is much more steady (Fig. 8). At group size of 1000, TOMA generates only about one third as many control messages as NICE. The reason is simple: in TOMA, the local clusters remain rather static and a member stays in the same cluster in spite of the behaviors of other members. However, in NICE, as members join and leave, the clusters split and merge very frequently to enforce the bounds on cluster size, which induces numerous control messages.

Control overhead for multiple groups in AT&T topology To examine the effectiveness of aggregated multicast, we estimate the saving in control overhead when tree aggregation is adopted. In TOMA, the establishment and tear-down of multicast trees are accomplished through the relay of tree setup messages by intermediate proxies, and the multicast state maintained by each proxy needs to be explicitly refreshed periodically. Correspondingly, we quantify *multicast tree setup overhead* with the total number of tree setup messages and *tree maintenance overhead* with the total number of refresh messages.

In the simulation experiments, we assume multicast groups arrive as a Poisson process, and their lifetime has an exponential distribution. Recall that the group-tree matching algorithm presented in Section 2.4 controls the bandwidth waste with a

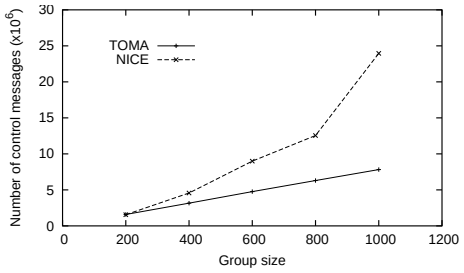

Figure 8: Control Overhead for a single group in Transit-Stub topology.

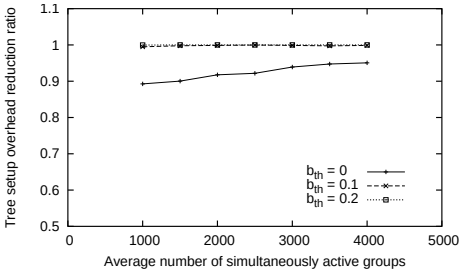

Figure 9: Multicast tree setup overhead reduction ratio for multiple groups in AT&T topology.

threshold b_{th}, which we vary from 0 to 0.2 in the simulations.

We plot the reduction ratio (i.e., the percentage of control overhead reduced by using aggregated multicast) of multicast tree setup and maintenance overhead in Fig. 9 and 10, respectively. Clearly, in both cases, about 70 to 90% control overhead can be reduced even when b_{th} is 0. We also observe that as more multicast groups become active, the reduction ratio improves. When b_{th} is raised to 0.2, we are able to eliminate more than 98% of the control overhead, though at the expense of higher bandwidth waste. These results indicate that overlay ISPs can control the trade-off of bandwidth cost versus control overhead by setting b_{th} properly.

5.4 Multicast State Scalability

To investigate the multicast state scalability of TOMA, we perform the simulations for multiple groups in AT&T topology. Since member proxies maintain group state information even with tree aggregation, we only measure the *reducible multicast forwarding entries*, that is, the multicast state entries in non-member proxies. The results for reduction ratio of reducible multicast forwarding state with aggregated multicast are shown in Fig. 11. This figure exhibit a similar trend as Fig. 9 and 10: more reduction in multicast state is accomplished when there are more concurrent groups and more bandwidth is wasted to

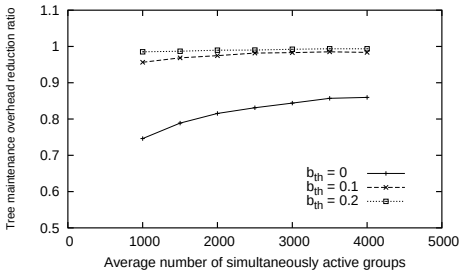

Figure 10: Multicast tree maintenance overhead reduction ratio for multiple groups in AT&T topology.

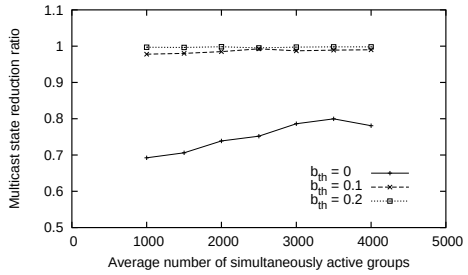

Figure 11: State reduction ratio for multiple groups in AT&T topology.

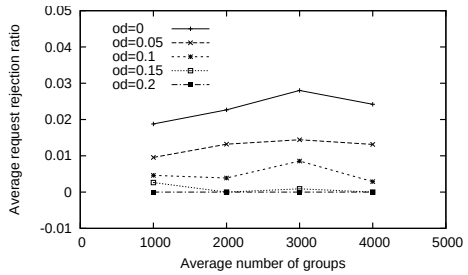

Figure 12: Request rejection ratio for overlay bandwidth dimensioning.

force higher aggregation. This figure demonstrates that TOMA is scalable to the number of co-existing groups.

5.5 Overlay Dimensioning

In this subsection, we evaluate the bandwidth dimensioning algorithm. The overlay network is constructed on the abstracted AT&T backbone topology. We assume the properties of multicast groups are obtained through long-term measurements, and we use the following model to capture these properties. For every group, a member proxy has attached end users participating in this group with a given probability as described in Section 5.1. The groups arrive according to a Poisson process, and the average life time is exponentially distributed. Each group requires the same bandwidth throughout its lifetime, and the bandwidth requirement for different groups follows some distribution. In the simulations, we define three types of multicast groups: 50% of the groups are low bandwidth (10K), 30% are medium bandwidth(100K), and 20% are high bandwidth (1M).

In our simulation, we generate group traces with different number of groups. For each group trace, we compute the aggregated trees using the group-tree matching algorithm, determine the bandwidth required on each link at each sampling point, and take the average of the samples. Then we over-dimension the bandwidth by a certain amount and validate the dimensioning results by measuring *group join request rejection ratio* for another group trace with the same properties. The join request of a multicast group is rejected if the residual bandwidth of the overlay links is not enough to accommodate the multicast tree computed for that group.

Fig. 12 shows the average request rejection ratio when the percentage of over-dimensioning (denoted as od) and the average number of active groups are varied. We observe two trends in the figure. First, the rejection ratio remains as low as $2-3\%$ for different numbers of co-existing groups, even without over-dimensioning. It is reasonable that the rejection ratio is not 0 because the bandwidth is reserved for average traffic rates instead of peak rates. Second, as od increases, the rejection ratio decreases. When 15% more bandwidth is reserved, the rejection ratio is close to 0. These results indicate that our bandwidth dimensioning scheme is indeed very effective.

In conclusion, our observations through simulation experiments can be summarized as follows: TOMA creates multicast distribution trees with tree cost and average link stress almost comparable to that of IP multicast; the data paths of TOMA

trees have lower latency than those of NICE trees; the control overhead of TOMA is significantly less than NICE for large groups; TOMA is scalable to large numbers of groups in terms of control overhead and multicast state. Finally, we also demonstrate the effectiveness of our bandwidth dimensioning scheme.

6 Related Work

Recently application layer multicast has emerged as a new architecture to apply multicast paradigm in the Internet. The proposed schemes can be classified into two categories: structured [27] [30] [6], and unstructured [10] [3] [26] [22] [17]. Structured application layer multicast schemes leverage Distributed Hash Table (DHT)-based overlay networks and build multicast forwarding trees on top of this structured overlay. Here we concentrate on unstructured application layer multicast schemes because they are more related to our work.

End System Multicast [10] [9] and Application Level Multicast Infrastructure (ALMI) [26] are targeted at applications with small and sparse groups, such as audio and video conferences. In End System Multicast, end hosts periodically exchange group membership information and routing information, build a mesh based on end-to-end measurements, and run a distributed distance vector protocol to construct a multicast delivery tree. The authors also demonstrate the importance of optimizing and adapting the overlay to application-specific requirements such as latency and bandwidth. ALMI uses a centralized entity to collect membership information, periodically calculate a minimum spanning tree based on the measurement updates received from all members, and distribute this information to group members.

NICE [3], on the other hand, is designed to support applications with very large receiver sets and relatively low bandwidth requirements. It recursively arrange group members into a hierarchical overlay topology, which implicitly defines a source-specific tree for data delivery. It has been shown that NICE scales to large groups in terms of control overhead and logical hops.

Other application layer multicast protocols include TAG [22] and Yoid [17]. TAG exploits the underlying network topology when constructing application-layer multicast trees. In Yoid, each member is responsible for discovering and selecting a parent, and thus multicast trees are constructed without underlying mesh.

However, pushing the multicast functionalities to end systems creates a dilemma as to who is motivated to deploy this architecture. Despite the flexibility and self-organization capability of application-layer multicast protocols, the distributed nature of multicast group and tree information prevents overlay service provider from managing and charging the multicast groups efficiently. Thus, the end users must be relied upon to cooperatively and spontaneously adopt this kind of protocols. However, the end users do not care much about bandwidth resource efficiency, which means they have no incentives to employ these protocols at all. In addition, due to the inherent inefficiency of bandwidth usage and the difficulty of managing group membership and maintaining multicast trees, the above-mentioned protocols provide very limited support (if any) for large multicast groups.

A number of schemes have also been proposed to construct multicast overlay using proxies. Overcast [19] provides reliable single-source multicast by using a distributed protocol to build data distribution trees rooted at a central source. This root is responsible for redirecting a client's HTTP requests to a Overcast node, from which the client receives data using TCP connection. RMX [8] provides reliable data delivery to heterogeneous end users by using a set of RMX proxies that are organized into a spanning tree. The end users are split into a number of locally-scoped multicast data groups of homogeneous members, each of which contains a RMX proxy. Each RMX proxy communicates with peer proxies using TCP and uses simple multicast congestion control within its data group.

[29] [28] and [4] focus on optimizing the end-to-end delay and access bandwidth usage at the Multicast Service Nodes. Shi et al proposes a set of heuristic algorithms to solve minimum-diameter degree-limited spanning tree problem and bounded-diameter residual-balanced spanning tree problem [29] [28]. The authors of [4] formulate the problem as minimum average-

latency degree-bounded spanning tree problem and proposed an iterative distributed algorithm.

To the best of our knowledge, these existing overlay multicast protocols mainly focus on supporting multicast for a single application or a single group, and most of them do not address the issues of who is responsible for deploying overlays and how to dimension them. In contrast, our work targets at a completely different service model, namely, multicast service overlays deployed and managed by overlay service providers to support a variety of applications and numerous groups. Accordingly, we are challenged by a number of different issues potentially faced by MSON providers.

7 Conclusions

In this paper, we propose and develop a two-tier overlay multicast architecture (TOMA) to support a variety of group communication applications in an efficient and scalable way. Our contributions could be summarized as follows:

- We provide a detailed design of the TOMA architecture, which adopts MSON as the backbone service domain and application-layer multicast in the access domains to achieve efficient resource utilization with reduced control overhead.
- By applying aggregated multicast inside MSON, the control overhead for establishing and maintaining multicast trees can be further reduced, and significantly less forwarding state needs to be maintained at proxy nodes.
- To efficiently plan backbone service overlay, we develop several dimensioning algorithms to locate proxies, select overlay links, and allocate link bandwidth.
- We suggest a simple but effective pricing model, which would stimulate the ISPs and customers to deploy and purchase TOMA service.
- By extensive simulation studies, we show that the performance of TOMA is very promising: it can provide efficient multicast transmission comparable to IP multicast, and it is scalable to both large group size and large numbers of co-existing groups.
- Simulation studies also show that our dimensioning algorithms could efficiently plan the network resources with little penalty.

We believe that the invention of our practical, comprehensive, and profitable multicast service model would significantly facilitate the multicast wide deployment, making multicast service overlay from myth to reality.

References

[1] *AT&T IP Backbone*, 2001. http://www.ipservices.att.com/backbone/.

[2] A. Ballardie. Core Based Trees (CBT version 2) multicast routing: protocol specification. *IETF RFC 2189*, Sept. 1997.

[3] S. Banerjee, C. Kommareddy, and B. Bhattacharjee. Scalable application layer multicast. In *Proceedings of ACM SIGCOMM*, Aug. 2002.

[4] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, and S. Khuller. Construction of an efficient overlay multicast infrastructure for real-time applications. In *Proceedings of IEEE INFOCOM*, Apr. 2003.

[5] K. Calvert, E. Zegura, and S. Bhattacharjee. How to model and internetwork. In *Proceedings of IEEE INFOCOM*, Mar. 1996.

[6] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron. Scribe: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communications*, 20(8):1489 – 1499, Oct. 2002.

[7] Y. Chawathe, S. McCanne, and E. A. Brewer. *An Architecture for Internet Content Distributions as an Infrastructure Service*, 2000. Unpublished, http://www.cs.berkeley.edu/yatin/papers/.

[8] Y. Chawathe, S. McCanne, and E. A. Brewer. RMX: Reliable multicast for heterogeneous networks. In *Proceedings of IEEE INFOCOM*, Mar. 2000.

[9] Y.-H. Chu, S. G. Rao, S. Seshan, and H. Zhang. Enabling conferencing applications on the internet using an overlay multicast architecture. In *Proceedings of ACM SIGCOMM*, Aug. 2001.

[10] Y.-H. Chu, S. G. Rao, and H. Zhang. A case for end system multicast. In *Proceedings of ACM Sigmetrics*, June 2000.

[11] J. Chuang and M. Sibru. Pricing multicast communications: A cost-based approach. *Proceedings of the Internet Society INET'98 Conference*, July 1998.

[12] J.-H. Cui, J. Kim, A. Fei, M. Faloutsos, and M. Gerla. Scalable QoS multicast provisioning in Diff-Serv-supported MPLS networks. In *Proceedings of IEEE GLOBECOM*, Nov. 2002.

[13] C. Diot, B. Levine, J. Lyles, H. Kassem, and D. Balensiefen. Deployment issues for the IP multicast service and architecture. *IEEE Network*, Jan. 2000.

[14] Z. Duan, Z.-L. Zhang, and Y. T. Hou. Service overlay networks: Slas, qos, and bandwidth provisioning. *IEEE/ACM Transactions on Networking*, 11(6):870–883, 2003.

[15] A. Fei, J.-H. Cui, M. Gerla, and D. Cavendish. A "dual-tree" scheme for fault-tolerant multicast. In *Proceedings of IEEE ICC*, June 2001.

[16] A. Fei, J.-H. Cui, M. Gerla, and M. Faloutsos. Aggregated Multicast: an approach to reduce multicast state. *Proceedings of Sixth Global Internet Symposium(GI2001)*, Nov. 2001.

[17] P. Francis. *Yoid: Extending the Multicast Internet Architecture*. White papar, http://www.aciri.org/yoid/.

[18] S. Herzog, S. Shenker, and D. Estrin. Sharing the "cost" of multicast trees: An axiomatic analysis. In *Proceedings of ACM SIGCOMM*, Aug. 1995.

[19] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O. Jr. Overcast: Reliable multicasting with an overlay network. In *Proceedings of USENIX Symposium on Operating Systems Design and Implementation*, Oct. 2000.

[20] B. M. Khumawala. An efficient branch and bound algorithm for the warehouse location problem. *Management Science*, 18(12):B718–B731, Aug. 1972.

[21] A. A. Kuehn and M. J. Hamburger. A heuristic program for locating warehouses. *Management Science*, 9(4):643–666, July 1963.

[22] M. Kwon and S. Fahmy. Topology aware overlay networks for group communication. In *Proceedings of NOSSDAV'02*, May 2002.

[23] Z. Li and P. Mohapatra. QRON: QoS-aware routing in overlay networks. *IEEE Journal on Selected Areas in Communications, Special Issue on Service Overlay networks*, 22(1):29–40, Jan. 2004.

[24] M. Medard, S. Finn, R. Barry, and R. Gallager. Redundant trees for preplanned recovery in arbitrary vertex-redundant or edge-redundant graphs. *IEEE/ACM Transactions on Networking*, 7(5):641–652, Oct. 1999.

[25] A. Nakao, L. Peterson, and A. Bavier. A routing underlay for overlay networks. In *Proceedings of ACM SIGCOMM*, Aug. 2003.

[26] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel. ALMI: An application level multicast infrastructure. In *Proceedings of the 3rd USNIX Symposium on Internet Technologies and Systems*, Mar. 2001.

[27] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Application-level multicast using content-addressable networks. In *Proceedings of NGC*, Nov. 2001.

[28] S. Shi and J. S. Turner. Routing in overlay multicast networks. In *Proceedings of IEEE INFOCOM*, June 2002.

[29] S. Shi, J. S. Turner, and M. Waldvogel. Dimensioing server access bandwidth and multicast routing in overlay networks. In *Proceedings of NOSSDAV'01*, June 2001.

[30] S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. H. Katz, and J. D. Kubiatowicz. Bayeux: An architecture for scalable and fault-tolerant wide-area data dissemination. In *Proceedings of NOSSDAV'01*, June 2001.