

Algorithms and Trade-Offs in Multicast Service Overlay Design

Li Lao¹, Jun-Hong Cui², Mario Gerla³

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

¹ Google Santa Monica, 604 Arizona Avenue, Santa Monica, CA 90401

² Computer Science & Engineering Department, University of Connecticut, Storrs, CT 06269

³ Computer Science Department, University of California, Los Angeles, CA 90095

Corresponding Author: Li Lao

Contact Address: Google Santa Monica

604 Arizona Avenue

Santa Monica, CA 90401

Fax: (310) 309-6840 **Tel:** (310) 309-6937 **Email:** llao@google.com

Abstract

Multicast Service Overlay Network (MSON) has been recently introduced to address some of the deployment and maintenance problems of IP multicast and application-level multicast. An MSON is essentially a backbone service overlay provisioned by an MSON provider and designed to “multiplex” multiple user overlays. It consists of service nodes or proxies deployed by the provider. The proxies are strategically deployed to form a backbone overlay and provide multicasting functionalities for supporting many user groups simultaneously. The MSON provider designs the backbone overlay for a large user population and can therefore enjoy large scale economies and benefits. Since the deployment of an MSON is a capital-intensive investment, it is very imperative to carefully design the MSON so that the provider can make the best revenue out of this investment. In this paper, we formulate the MSON design problem by taking into account operational costs of the MSON provider. We then divide into three sub-problems: overlay proxy placement, overlay link selection, and bandwidth dimensioning. For each of these sub-problems, we present several algorithms and discuss their design trade-offs. By simulations, we investigate the effectiveness different overlay design algorithms, analyze their impact on multicast performance, and suggest some practical solutions for MSON design.

Index Terms

Overlay Design, Multicast Service Overlay Networks, Algorithms, Integer Linear Programming, Greedy

I. INTRODUCTION

Multicast is a useful data delivery method for group communication (from one or many sender(s) to many receivers). Over the years, a lot of research, development, and testing efforts have been devoted to multicast support, from early IP multicast to today’s application-layer multicast and overlay multicast.

IP multicast is resource efficient by utilizing a tree delivery structure. However, there are many issues which make Internet Service Providers (ISPs) reluctant to deploy and provide IP multicast service ([11], [2]), such as the lack of a scalable inter-domain routing protocol, the requirement of global deployment of multicast-capable IP routers and the lack of appropriate pricing models.

To overcome the scaling limitations of IP multicast, there has been increasing interest in application level multicast, in which multicast functionalities are pushed to end systems. End systems cooperatively

construct and maintain an overlay (which is referred to as a “user overlay”) among themselves to support a group of users, and multicast trees are constructed on top of this overlay. Several successful experiments have been reported in video and real time multicasting, and the state of the art of designing stable, efficient and scalable end system overlays has been steadily improving ([14], [9], [21], [4]). A major challenge in application-layer multicast and also a cause of substantial overhead has been the “provisioning” of the overlay topology, in part due to available bandwidth estimation on various candidate links. Compared with IP multicast, application-layer multicast protocols are much more flexible, with no requirement of global deployment of multicast-capable routers. However, they usually suffer from the inefficiency of bandwidth usage and the difficulty of managing group membership and maintaining multicast trees especially when multicast groups are large. Moreover, this approach excludes ISPs out of the picture, which, however, is the most motivated party to deploy multicast. Note that end users usually do not care too much about how group communication is implemented (using unicast or multicast), as long as they could get desired services at good prices.

To address these deployment and maintenance issues of multicast overlays, recently, the notion of Multicast Service Overlay Network (MSON) has been introduced. An MSON is essentially a backbone service overlay provisioned by an MSON provider and designed to “multiplex” multiple user overlays. It consists of service nodes or proxies deployed by the provider. The proxies are strategically deployed to form a backbone overlay and provide multicasting functionalities for supporting many user groups simultaneously. The MSON provider designs the backbone overlay for a large user population and can therefore enjoy large scale economies and benefits. The design of an MSON relies on a well-defined business relationship between the MSON provider, network service providers (i.e., underlying network domains), and end users: the MSON provider provisions its overlay network according to end user requests, purchases bandwidth from network service providers based on their service level agreements (SLAs), and sells its services to end users via contracts. This characteristic allows the MSON provider to better design networks in order to improve application performance, and it is complementary to existing end system

based approaches. In access domains, end users can communicate with each other by subscribing to MSON proxies. Indeed, communication among end users outside the MSON can employ application layer multicast schemes. For example, Two-tier Overlay Multicast Architecture (TOMA) has been proposed to provide large-scale multicast service [19]. It advocates MSON as the backbone service domain, and end users outside the MSON (i.e., in access domains) form a number of clusters, where an application-layer multicast protocol is used between clustered end users. With carefully designed MSONs, TOMA is claimed to be a scalable, efficient, and practical solution to multicast support.

Clearly, the deployment of an MSON is a capital-intensive investment. Thus, it is very imperative to carefully design the MSON so that the MSON provider can make the best revenue under given investment. In this paper, we focus our efforts on MSON design by taking into account the operational costs of MSON providers. We formulate the problem and divide it into three sub-problems: overlay proxy placement, overlay link selection, and bandwidth dimensioning. For each sub-problem, we present several algorithms and discuss their design trade-offs: ILP and greedy algorithms for overlay proxy placement; Complete Graph, K-MST (Minimum Spanning Tree), and Adjacent Connection for overlay link selection; and a simulation-based bandwidth dimensioning approach. Through simulation studies, we investigate the effectiveness of these algorithms, and analyze their impact on the performance of TOMA. It is worth pointing out that our algorithms are not applicable to TOMA only. They can be adopted or easily adapted for the design of general multicast service overlay networks.

The remainder of this paper is organized as follows. In Section II, we review some background information as well as related work. In Section III, we formulate the overlay provisioning problem, and divide it into three sub-problems. We then present some practical algorithms for each sub-problem in Section IV. Finally, we evaluate the performance of the proposed algorithms through simulation study in Section V, and draw conclusions in Section VI.

II. BACKGROUND AND RELATED WORK

In this section, we give a brief review on TOMA. We also go over some closely related work.

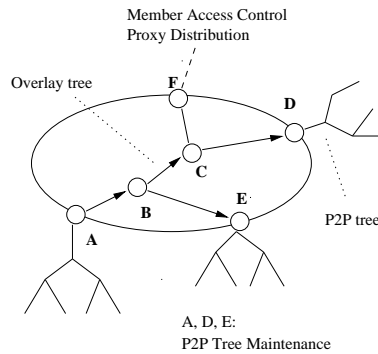


Fig. 1. A high-level picture of TOMA, where F is the group registry server/DNS server, and A, B, C, D and E are overlay proxies.

A. Review of TOMA

TOMA is a two-tier overlay multicast architecture [19], in which MSON is advocated as the service backbone domain. In the MSON, overlay proxies are strategically placed to form an overlay network, on top of which multicast distribution trees are built for data delivery. Outside the MSON, end users subscribe by transparently connecting to some proxies advertised by the MSON provider. Each 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.

TOMA uses URL-like unique names to identify groups (e.g., *TOMA://groupname.xyzmson.com/*). An end user explicitly subscribes 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. After receiving the request, the DNS server will send back to the subscriber a list of IP addresses of the advertised proxies, from which an appropriate proxy will be selected by the user.

After finding a proxy, the end user sends its join request to the proxy, which will then “subscribe” to the multicast group inside the MSON on behalf of this member. The proxy will add the new user to the peer-to-peer multicast tree already set up for this group (or create a new tree) in the local cluster. It will relay the join request towards the source proxy (the proxy to which the source subscribes) in order to extend (or establish) an overlay multicast tree in the backbone domain.

A high level picture of TOMA is illustrated in Fig. 1. There is one multicast group with a source proxy

A and destination proxies D and E in this figure. An overlay multicast tree is built in the MSON for this group. Three P2P trees (one for each proxy) are established outside the MSON. It is claimed that this architecture can provide scalable and efficient multicast support if the MSON is carefully designed [19]. In this paper, we will study different provisioning algorithms and examine their effects on the performance of TOMA.

B. Related Work

Besides TOMA, several other schemes have employed multicast overlay using proxies, such as Overcast [16], Scattercast [7], RMX [8], AMcast [23] and OMNI [5]. However, these architectures are based on different service models from TOMA. Namely, TOMA is based on a well-defined business relationship between MSON providers, network service providers, and end users (or group coordinators). In contrast, Overcast and RMX are deployed exclusively in the user domain; they do not involve the ISPs. Consequently, they mainly address user performance issues such as reliability. In [23], [22] and [5], although the authors mention the concept of overlay tree construction, they focus on optimizing end-to-end delay and access bandwidth usage at the proxies. The actual overlay proxy placement, overlay link selection, and bandwidth dimensioning within the service overlay network are usually not addressed. Instead, the authors in [23], [22] and [5] all assume that the proxies are pre-deployed, overlay links between any two proxies are possible (i.e., a full mesh among proxies is used), and the link capacity is simplified as the proxy out-degree bound. Moreover, these proposals only consider overlay tree construction for a single group. In contrast, our work addresses all the problems of proxy placement, link selection, and bandwidth dimensioning by taking into consideration large numbers of groups.

The overlay architecture closely related to our work is SON [12], which incorporates bandwidth dimensioning algorithms for a unicast service overlay network. Due to the many extra features required by multicast, bandwidth dimensioning for unicast is not sufficient for multicast services. Issues of end user distribution, physical network awareness, and multicast routing etc, which are not considered in SONs, would be major concerns for MSONs. To the best of our knowledge, this paper is the first work focusing

on the design of multicast service overlay networks (MSONs).

III. PROBLEM FORMULATION

In this section, we formulate the MSON provisioning problem. To design a good MSON, we need to take into account traffic patterns (e.g., group and member distribution), group bandwidth requirements, end-to-end delay concerns, as well as multicast routing algorithms. And the objective is to minimize the network operation cost while satisfying the end user requirements.

Graph Model We model the physical network topology as an undirected graph $G = (V, E)$, where V and E denote the sets of network nodes (or routers) and physical links respectively. The total number of routers in the network is denoted as n . Each link $e \in E$ has a bandwidth capacity $c(e)$. We also denote the set of all possible paths (between any two nodes) in G as $path(E)$.

Group Model From long-term measurements, we can obtain a set of groups $\{g_i\}$ with group member distribution and bandwidth requirements [3], [6], [10], and then use this information for MSON design. To quantify the group member population clustered around a router, we use a random-weighted model [13], in which we assign to each router i a weight w_i . This weight represents the number of directly attached end users participating in multicast groups.

Problem Formulation The overlay provisioning problem can be formulated as follows: Given the set of groups $\{g_i\}$, and a physical network topology $G = (V, E)$, find a virtual topology $G' = (V', E')$ on top of G (where $V' \subset V$ and $E' \subset path(E)$), in which each $e' \in E'$ is assigned bandwidth $b(e')$, such that G' can accommodate all the groups, and the cost of G' and the average end-to-end delay of group members are minimized.

The determination of the cost of G' depends on the MSON management policy. In most cases, we can use the sum of the assigned bandwidth (purchased from underlying network service providers) to estimate the cost of bandwidth, and the number of overlay links (i.e., $|E'|$) to measure the overlay maintenance overhead.

Obviously, the MSON provisioning problem defined above is a multi-objective optimization problem, which is NP-hard due to the combinatorial explosion. To tackle this difficult problem, we use a divide-and-conquer approach: we first divide the whole problem into several sub-problems, and then provide practical solutions for each of them.

From the above definition, to obtain G' , the overlay provider needs to make three decisions: 1) determine the locations of the proxy nodes, i.e., select V' ; 2) select the overlay connections between these proxies, i.e., choose E' ; 3) compute the bandwidth to be reserved on each overlay link, i.e., assign a bandwidth $b(e')$ to each $e' \in E'$.

Following the principle of divide-and-conquer, we divide the overlay provisioning problem into three sub-problems: overlay proxy placement, overlay link selection and bandwidth dimensioning (note that this approach may incur “sub-optimality” of the solutions since its main concern is the problem manageability). We will present algorithms to solve them accordingly in the next section.

IV. PROVISIONING ALGORITHMS

A. *Overlay Proxy Placement*

Before deciding where to put proxies, we may have to solve this problem first: how many proxies a MSON should have. This again depends on the MSON management policy. In fact, the trade-off between multicast performance and the number of proxies K is clear: the larger the K , the better multicast performance. Two extreme cases are IP multicast, in which every router can replicate and forward multicast packets (i.e., $K = n$), and application layer multicast, in which there are no intermediate multicast-capable routers or overlay proxies (i.e., $K = 0$). Therefore, there are some trade-offs of choosing larger K vs. smaller K . In the following, we examine where to put proxies when K is fixed, while in Section V, we will investigate to what extent K will influence multicast performance.

When the total number of overlay proxies is bounded by K , a direct consideration for locating proxies is end-to-end delay. Since end users tend to receive data packets from the closest proxies, the locations of

overlay proxies directly affect the latency of data transmission. Intuitively, if a router is connected to a lot of users, a proxy should be placed near this router to reduce the average delay. Hence, we can minimize the total delay between users and their proxies.

In this way, the overlay proxy placement becomes an optimization problem: given the number of group members w_i ($1 \leq i \leq n$) for all n routers and the shortest distance d_{ij} between any two routers i and j ($1 \leq i, j \leq n$), find no more than K (a predefined value, and $1 \leq K \leq n$) routers as proxies, such that the weighted sum of distances from each router to its nearest proxy is minimized. We propose an ILP (Integer Linear Programming) formulation and a greedy algorithm to solve this problem.

1) *ILP Formulation:* Before presenting the ILP formulation, we define the following variables:

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

and

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

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 any router to its closest proxy weighted by the number of members attached to the router:

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

subject to the following constraints:

1) Every router subscribes to exactly one proxy:

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

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

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

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

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

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

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

The complexity of this ILP formulation is $O(n^2)$ in terms of the number of variables and the number of constraints. 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} .

2) *Greedy Algorithm*: By solving the above ILP formulation, an optimal solution can be obtained for the overlay proxy placement problem. However, due to its high computation complexity, when the number of variables or constraints is very large, the problem becomes intractable. Hence, we develop a greedy algorithm (Algorithm 1) to efficiently solve this problem.

The greedy algorithm works as follows: in each step, a router r is selected as a proxy if its selection can reduce the weighted sum of distances (i.e., the objective function) by the largest amount. This procedure repeats until the maximum number of overlay proxies K is reached. In each iteration, computing the weighted sum of distance for each router requires $O(n)$ time, and the whole iteration takes $O(n^2)$ time. Therefore, the time complexity of the algorithm is $O(Kn^2)$.

This algorithm tries to minimize the objective function at each step, but it may be stuck in local minima and not reach the global minimum. In Section V, we show that this algorithm is able to achieve competitive performance in comparison with the optimal ILP solution.

Considering Overlay Proxy Deployment Cost In the above algorithms, we assumed that the maximum number of proxies (K) is pre-determined. Alternatively, the overlay provider may want to minimize the

Algorithm 1 Greedy-Proxy(R, K)

```

1:  $P \leftarrow null$  //initialize proxy set
2: for  $i = 1$  to  $K$  do
3:   //  $K$  is the number of proxies to be selected
4:   for all  $r \in R$  do
5:     //  $R$  is the set of remaining routers
6:     compute the weighted sum of distance  $d_r$  if  $r$  is selected as proxy in addition to  $P$ 
7:   end for
8:    $r_{best} \leftarrow r$  with the minimum  $d_r$ 
9:    $P \leftarrow P \cup \{r_{best}\}$ 
10:   $R \leftarrow R - \{r_{best}\}$ 
11: end for

```

sum of data delivery cost and 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 shipping cost and warehouse maintenance cost. Efficient branch-and-bound algorithms [17] and heuristic algorithms [18] can be adopted to solve this version of the overlay proxy location problem.

B. Overlay Link Selection

Once the locations of the overlay proxies have been determined, the next step is to connect these proxies into a mesh, on top of which overlay multicast trees will be constructed inside the MSON. Usually, there are two optimization goals when constructing an overlay mesh, namely, minimizing end-to-end latency and minimizing the cost of multicast trees (thus minimizing the bandwidth cost of groups) constructed on this mesh. Since these two objectives generally can not be satisfied simultaneously, we propose different approaches for them separately.

1) *Optimizing End-to-end Latency*: A naive approach to construct an overlay mesh with optimized end-to-end latency is to use *Complete Graphs*, that is, an overlay link is established for every pair of proxies.

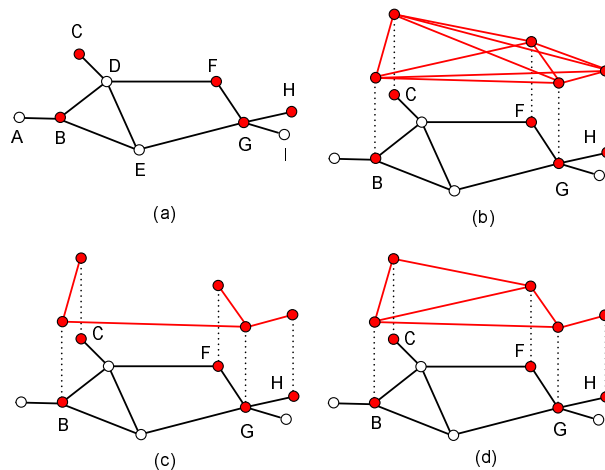


Fig. 2. An example of overlay link connection approaches. (a) Underlying physical network topology. Nodes B, C, F, G and H are proxy nodes. (b) Complete Graph. (c) 1-Minimum Spanning Tree. (d) Adjacent Connection.

In a network with n proxy nodes, this approach results in a total of $\frac{n(n-1)}{2}$ overlay links. Consequently, it induces a large amount of overlay maintenance overhead since directly connected overlay nodes usually need to periodically exchange refresh messages. An example of overlay link connection approaches is illustrated in Fig. 2. As shown in Fig. 2(a), among the 9 routers, B, C, F, G, and H are selected as overlay proxies. When the Complete Graph approach is adopted to connect the proxies (Fig. 2(b)), every pair of proxies use the shortest overlay path as the overlay link. As a result, any shortest path tree constructed on top of this overlay is essentially a star topology, in which a source proxy is connected to destination proxies directly. In other words, multicast in this case is equivalent to multiple unicast. Thus, multicast trees on Complete Graphs will have the shortest delay but the largest total cost and the highest overlay maintenance overhead.

To reduce tree cost and overlay overhead while achieving minimum delay, we can adopt an approach called *Adjacent Connection* [20] to “reduce” Complete Graphs: an overlay link is established between two overlay nodes only if the shortest IP-layer path between them does not go through any other overlay nodes. In other words, we eliminate the overlay links that are implied by (or composed of) other overlay links. For example, in Fig. 2(d), the overlay link (B, H) is removed because it is composed of the overlay links (B, G) and (G, H). In this way, the shortest overlay paths between any two proxies are exactly the

same as the network-level shortest paths, and it can create efficient multicast trees instead of multiple unicast paths. Therefore, the constructed overlay networks resemble the underlying network topologies and have high routing efficiency.

2) *Optimizing Tree Cost*: Since minimal spanning trees (MSTs) aim at minimizing the total cost of links on the tree, only overlay links with low costs will be included in this type of graphs. Thus we can use an MST as the overlay topology when trying to optimize the cost of multicast trees constructed on top of it. Though the MST approach minimizes multicast tree cost, the end-to-end delay between some proxies may be increased due to the limited set of overlay links. For example, in Fig. 2(c), an MST is constructed as the overlay network. The network-level shortest path from proxy B to F is 2 hops; however, the shortest path on the overlay network has to go through proxy G and thus has a length of 3 hops. Additionally, this approach has obviously the fewest number of overlay links (i.e., $n - 1$) and the lowest overlay maintenance overhead, but it tends to concentrate multicast traffic on a few overlay links, and it is also more susceptible to node/route failures.

To improve the fault tolerance and load balancing, a more general approach is to construct k -MST, in which k least-overlapping minimum spanning trees are used to connect overlay proxies [26]. k -MST in fact sits in between the two worlds: minimizing end-to-end delay and minimizing tree cost. We will examine the trade-offs of the presented overlay link selection approaches through simulations in Section V.

C. Bandwidth Dimensioning

Having determined the overlay topology (overlay proxies and links), we then need to decide the bandwidth required on overlay links to accommodate the groups obtained from long-term measurements.

Bandwidth dimensioning is a complex procedure since it is tightly coupled with multicast routing algorithms. Thus, we suggest a simulation-based bandwidth dimensioning approach, taking multicast routing algorithms into account. The basic idea is as follows: we compute multicast trees based on the given routing algorithm for all the groups, then by summing up the traffic volume on each 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 the bandwidth requirement of these groups on average. In reality, however, the traffic peak rate is likely to exceed the average rate frequently. Hence, the MSON provider should consider over-dimensioning the network by reserving a certain percentage of extra bandwidth. If the bandwidth is still insufficient during peak hours, the MSON can either reject some groups or lease bandwidth from higher-tier ISPs for short-term usage (possibly at a higher price). Again, there is another trade-off here: request rejection ratio and extra bandwidth reservation. The more the extra bandwidth reservation, the smaller the request rejection ratio, but the more the provisioning cost. Thus, it is important to explore good trade-off points for different applications.

V. SIMULATION STUDY

In this section, we evaluate the effectiveness and trade-offs of our overlay provisioning algorithms. We investigate the impact of the overlay proxy placement, overlay link selection, and bandwidth dimensioning algorithms on the performance of TOMA. And we also explore the trade-offs in these algorithm designs.

A. Simulation Settings

1) *Topologies and Group Models*: Considering that different topologies may affect the effectiveness of the proposed algorithms, we use two different network topologies in the simulation experiments. The first topology is abstracted from a real network, 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 are “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 represent connectivity to other peering networks and/or Internet public exchange points. This abstraction procedure results in a simplified network of 54 nodes.

For this topology, we use a weight-based model mentioned in Section III to generate group members: for each router i , we randomly attach end hosts to this router with a probability proportional to w_i . Thus, for a group with a fixed size, the average number of group members attached to a router is proportional to this router's weight. Among the 54 routers, gateway nodes and backbone nodes are assumed to be core routers 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 the access routers from which it is contracted. Exchange nodes are assigned a weight of 0.9 since they usually connect to peering networks and tend to have a large number of group members.

The second topology is a router-level topology (which is referred to as 1755 topology in this paper) with approximately 300 routers (measured by Rocketfuel Project [24]). In the 1755 topology, we use a random group member distribution model to generate group members, since we have no information to differentiate the routers. On the other hand, by using the random distribution model, we also tend to explore how member distribution models affect the overlay provisioning algorithms.

Multicast group members join multicast groups during an interval of 400 seconds. Inside the overlay domain, source-based overlay trees are constructed for data delivery between proxies; outside the overlay network, application-layer minimum spanning trees rooted at the proxy nodes are established for communication among clustered end users. Data are collected after multicast trees stabilize.

2) *Overlay Parameters:* For the overlay topology construction, we set the number of proxy nodes to 9 for the AT&T topology, which is equal to the number of exchange nodes (to which the majority of members are attached), and 20 for the 1755 topology. Unless otherwise specified, proxy nodes are placed in the locations determined by the greedy algorithm and connected using the Adjacent Connection approach as explained in Section IV-B.

3) *Multicast Performance Metrics:* To measure the quality of multicast trees, we use three performance metrics, i.e., multicast tree cost, link stress, and path length.

- *Multicast tree cost* is defined as the total number of physical links in a multicast tree, and it measures

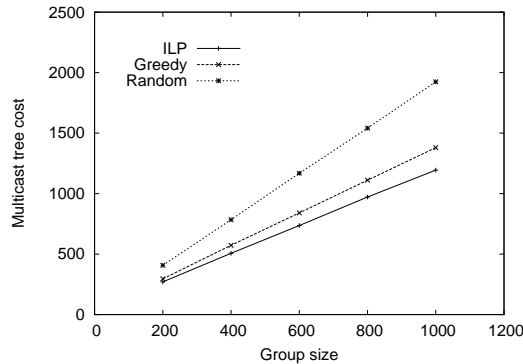


Fig. 3. Multicast tree cost for different proxy placement approaches in the AT&T topology.

the resource usage of a multicast tree.

- *Link stress* is measured by the number of redundant packets delivered on a single network link, and it quantifies the efficiency of a multicast tree.
- *Path length* is defined as the number of hops on the end-to-end path from a source to a receiver, and it measures the quality of data delivery paths.

B. Overlay Proxy Placement

To investigate the effectiveness of our proxy placement algorithms, we compute proxy locations based on the weights assigned to the routers using the ILP and greedy algorithms, and compare the performance of multicast groups on the resulting overlay networks. As a reference, we also include the multicast performance when proxies are randomly selected.

1) *Changing Group Size*: In the first set of simulations, we use the AT&T topology and vary the multicast group size from 200 to 1000.

The results of multicast tree cost vs. group size are plotted in Fig. 3. The ILP solution yields the lowest tree cost, since it optimally places proxies as close to exchange routers (which have a bulk of members) as possible, and thus prevents the packets from traversing the links between proxies and access routers multiple times. In contrast, the random approach chooses proxies randomly from all the routers in the network. As a result, end users may use some fairly long paths to connect to the proxies, and some

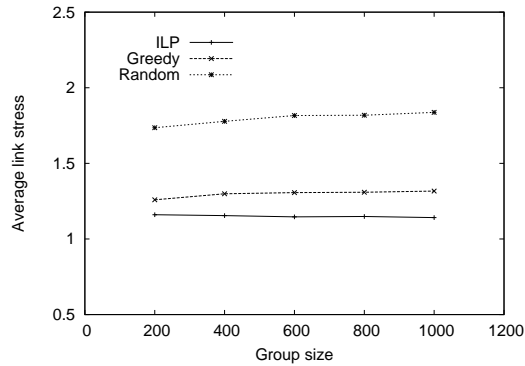


Fig. 4. Average link stress for different proxy placement approaches in the AT&T topology.

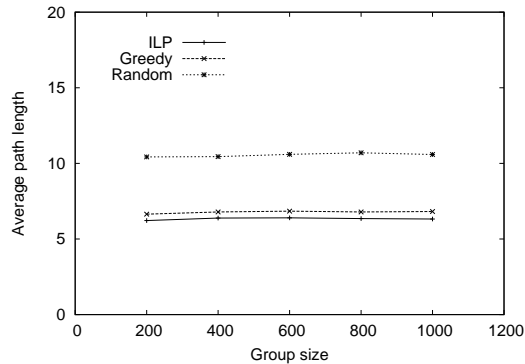


Fig. 5. Average path length for different proxy placement approaches in the AT&T topology.

physical links may be used multiple times by different users. As for the greedy algorithm, it selects some gateway routers and exchange routers, so its performance lies in between.

For the same reason, a similar trend can be observed for average link stress, which is shown in Fig. 4. Note that link stress is determined by the multicast tree cost and the number of distinct physical links in the tree, so even though tree cost increases with group size, link stress remains rather static.

The average end-to-end path length is depicted in Fig. 5. It is clear that by intelligently placing proxy nodes, both the ILP and greedy solutions reduce the average path length greatly. For example, the average path length is approximately 10.4 to 10.7 when proxies are randomly placed, and it is reduced to below 7.0 when the ILP or greedy approach is adopted. This remarkable improvement is again due to the fact that the two proposed approaches place proxies closer to end users to avoid using unnecessarily long paths for data delivery.

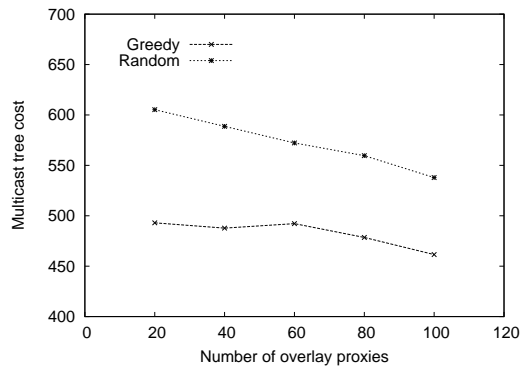


Fig. 6. Multicast tree cost for different proxy placement approaches in the 1755 topology.

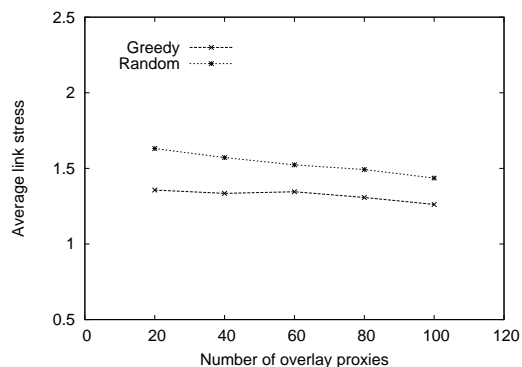


Fig. 7. Average link stress for different proxy placement approaches in the 1755 topology.

2) *Varying Number of Proxies*: Recall that the number of proxies K is an important parameter in determining multicast performance. In the second set of simulations, we increase K from 20 to 100 in the 1755 topology, and measure the corresponding multicast performance. Due to the long computation time of ILP when the number of nodes is high, we only plot the results of the greedy algorithm and the random approach.

As shown in Fig. 6 and Fig. 7, multicast tree cost and link stress decrease with the number of proxies for both greedy and random algorithms. As a larger number of proxies are deployed in the network, more nodes are capable of replicating and forwarding multicast data packets. Hence, there are fewer overlapping links in the multicast tree, which reduces the tree cost and link stress.

In Fig. 8, we observe that larger K tends to reduce the end-to-end path length, since end users can usually find closer proxies to connect to. In addition, we find that the slopes of both curves (i.e., greedy

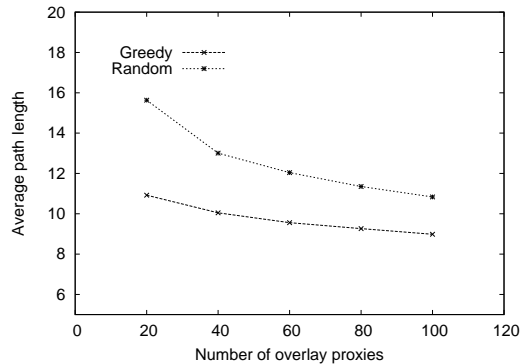


Fig. 8. Average path length for different proxy placement approaches in the 1755 topology.

and random algorithms) are smaller at lower K than those at larger K . This trend indicates that, as K increases, the reduction in path length becomes gradually smaller. In other words, the benefit of shorter end-to-end paths achieved by sacrificing proxy deployment and maintenance cost becomes less with larger K . Therefore, it is important to determine the value of K in order to balance the benefit and the cost.

C. Overlay Link Selection

In this set of experiments, we connect proxies using the three approaches described in Section IV-B, namely, Adjacent Connection (AC), Complete Graph (CG) and k -MST (Minimum Spanning Tree), and compare the performance of resulting overlays. For k -MST, we set k to be 1, 2, and 4 separately. Since the quality of the application-level multicast trees outside the overlay network only depends on the locations of the proxy nodes and the end users rather than the connections between the proxies, we only consider the performance of multicast trees between proxy nodes inside the overlay domain. We present the results of overlay multicast performance in Table I for the AT&T topology and Table II for the 1755 topology.

1) *Tree Cost and Link Stress*: As shown in these two tables, Minimum Spanning Tree and Adjacent Connection produce lower tree cost and link stress, whereas Complete Graph gives the highest values for both metrics. This is reasonable, since the Minimum Spanning Tree approach includes only overlay links with low cost, and the Adjacent Connection method removes overlay links that go through some intermediate proxies. On the other hand, as explained in Section IV-B, in the Complete Graph scheme,

TABLE I

COMPARISON OF OVERLAY LINK SELECTION APPROACHES IN THE AT&T TOPOLOGY.

Metrics	CG	AC	1-MST	2-MST	4-MST
<i>Number of Overlay Links</i>	36	27	8	16	30
<i>Multicast Tree Cost</i>	28.1	23.7	18.0	19.8	24.7
<i>Link Stress</i>	1.75	1.50	1.17	1.23	1.54
<i>Path Length</i>	3.12	3.12	4.33	3.33	3.12
<i>Minimum Cut</i>	8	4	1	2	5

TABLE II

COMPARISON OF OVERLAY LINK SELECTION APPROACHES IN THE 1755 TOPOLOGY.

Metrics	CG	AC	1-MST	2-MST	4-MST
<i>Number of Overlay Links</i>	190	76	19	38	76
<i>Multicast Tree Cost</i>	81.9	48.3	31.2	38.3	46.3
<i>Link Stress</i>	2.23	1.32	1.01	1.12	1.30
<i>Path Length</i>	4.24	4.24	6.05	4.83	4.42
<i>Minimum Cut</i>	19	1	1	2	4

a shortest path tree has the same cost as the sum of the multiple unicast paths, and thus is the highest among all the algorithms.

2) *Path Length*: From Table I and II, it is clear that, in terms of path length, Complete Graph and Adjacent Connection perform the best among all the solutions. Since multicasting in a complete graph is equivalent to multi-unicasting, Complete Graph will have the shortest end-to-end delay. Adjacent Connection yields the same delay as Complete Graph, because it attempts to remove the overlay links without affecting the routing efficiency: if the shortest path between two proxies does not go through other proxies, they will be connected directly using the shortest paths; otherwise, they will connect to the intermediate proxy nodes via overlay links. Unlike these two approaches, MST eliminates overlay links with high cost and trades path length for higher tree efficiency (See the example given in Section IV-B).

3) *Overlay Maintenance Cost*: Furthermore, we find out that the numbers of overlay links (reflecting overlay maintenance cost) in the generated overlay topologies using different approaches are different. For example, in the AT&T topology, the number of overlay links is 8 for 1-MST, 16 for 2-MST, 27 for Adjacent Connection, 30 for 4-MST and 36 for Complete Graph. This metric measures the overlay maintenance overhead. Obviously, the observed results are consistent with our analysis in Section IV-B.

4) *Robustness of Overlay*: To evaluate the robustness of each overlay topology, we measure the minimum number of overlay links whose removal partition the overlay and thus disconnects some pair of proxies. The larger the minimum cut, the more robust the topology is. This problem is known as “minimum 2-cut” or “minimum cut” in graph theory [15], and we use the simple algorithm proposed in [25] to compute the minimum cut.

As illustrated in the tables, Complete Graph has the largest minimum cut, since it is the most dense graph and thus difficult to be partitioned. In contrast, 1-MST has the smallest minimum cut, because the removal of any link in a tree will partition it. The robustness of Adjacent Connection is topology dependent. In the AT&T topology, the proxies are relatively densely connected with one another in the network layer, so it requires at least 4 overlay links for network partition. In the 1755 topology, one proxy connects to all other proxies through the same intermediate proxy. Hence, this proxy can be isolated from other proxies by disconnecting the link between itself and the intermediate proxy.

5) *Trade-off in k -MSTs*: Another important observation is that, for minimum spanning trees (k -MSTs), the higher the k , the higher the tree cost and link stress, and the smaller the path length. This is consistent with our intuition: when we build more minimum spanning trees, additional overlay links are included, so shorter paths can usually be found, while tree cost is not optimal any more. This is another evidence of the trade-off between end user delay and tree efficiency.

Moreover, for larger k , the network has higher density and more overlay links need to be maintained. Nevertheless, the network becomes more robust, as the failure of a few links will cause the disconnection of proxy pairs.

D. Bandwidth Dimensioning

To determine the bandwidth for each overlay link, we model the traffic pattern as follows: 1) for every group, a proxy has attached end users participating in the group with a given probability as described in Section V-A; 2) groups arrive according to a Poisson process, and the average life time is exponentially distributed; 3) each group requires the same bandwidth throughout its lifetime, and the bandwidth requirements for different groups follow a certain distribution. In the simulations, we carry out simulations for both *homogeneous groups* and *heterogeneous groups*. For homogeneous groups, all the groups in the network has the same bandwidth requirement. On the other hand, for heterogeneous groups, 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)¹. We show the simulation results for homogeneous and heterogeneous groups separately.

In our study, we generate sample traces of multiple groups and use the simulation-based approach proposed in Section IV-C to determine the bandwidth assignment. For each trace of multiple groups, we compute their source-based multicast trees, determine the bandwidth required on each link at every 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 traffic pattern. A join request of a multicast group is rejected if the residual bandwidth of an overlay link is not enough to accommodate the multicast tree computed for that group.

1) *Homogeneous Groups*: Fig. 9 and 10 show the average request rejection ratio for homogeneous groups in the AT&T and 1755 topologies, respectively. In these two figures, the percentage of over-dimensioning (denoted as *od*) and the average number of active groups are varied. Due to the high computation overhead of the larger topology 1755, we use fewer groups for this topology in the simulation experiments.

¹These bandwidth requirements are typical for several common applications on the Internet, e.g., text message applications (10K), voice/audio or low-quality video applications (100K), and high-quality video applications (1M).

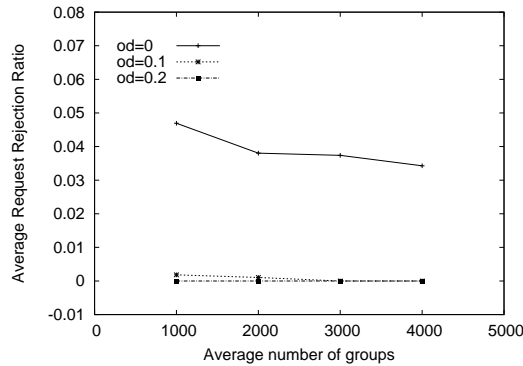


Fig. 9. Request rejection ratio for bandwidth dimensioning of homogeneous groups in the AT&T topology.

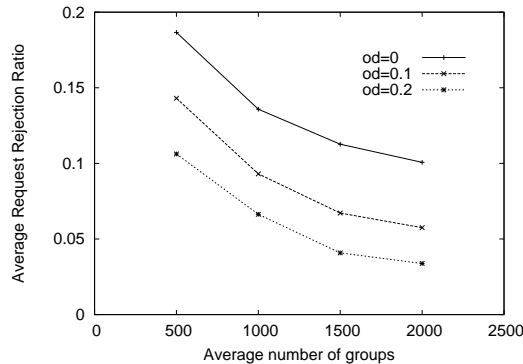


Fig. 10. Request rejection ratio for bandwidth dimensioning of homogeneous groups in the 1755 topology.

We observe four general trends in both figures. First, the rejection ratio remains very low 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 the average traffic rate instead of the peak rate. Second, the rejection ratio heavily depends on the specific topology and group membership model. For example, when there are 1000 active groups, the rejection ratio is approximately 5% for AT&T and 14% for 1755. One major reason is that the AT&T topology has a smaller number of overlay links; thus, multicast traffic is more “predictable” on each overlay link. Third, as od increases, the rejection ratio decreases, because more bandwidth is reserved. Fourth, the rejection ratio decreases when there are more groups in the network, since the sum of bandwidth requirements for a larger number of groups tend to vary less significantly. These results indicate that our bandwidth dimensioning scheme is indeed very effective.

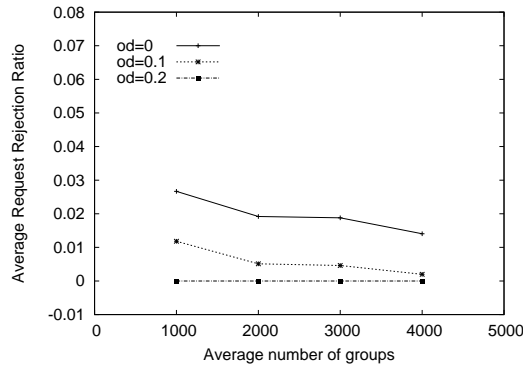


Fig. 11. Request rejection ratio for bandwidth dimensioning of heterogeneous groups in the AT&T topology.

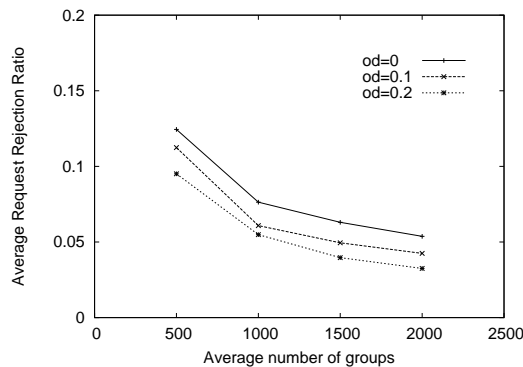


Fig. 12. Request rejection ratio for bandwidth dimensioning of heterogeneous groups in the 1755 topology.

2) *Heterogeneous Groups*: We plot the results for heterogeneous groups in the AT&T and 1755 topologies in Fig. 11 and 12, respectively. These two figures show the same trends as the figures for homogeneous groups. However, when comparing the absolute values of the rejection ratio for homogeneous groups with those for heterogeneous groups, we find that there are some variations. In particular, heterogeneous groups usually have a lower rejection ratio than homogeneous groups. For heterogeneous groups, a larger amount of bandwidth can be saved if high-bandwidth groups are rejected. In this way, by sacrificing a smaller number of high-bandwidth groups, more groups can be accommodated. We also observe that, in the special cases when bandwidth is sufficient for accepting most groups (e.g., when od is 0.1 or 0.2 in the AT&T topology), more groups can be admitted for homogeneous groups. This is because the homogeneous groups have the same bandwidth requirement, whereas high-bandwidth groups in heterogeneous groups have more stringent bandwidth requirement and are more difficult to satisfy.

E. Summary

Based on the simulation and analysis results, we can draw the following conclusions. 1) The performance of TOMA is significantly affected by the overlay provisioning algorithms. 2) For overlay proxy placement, the greedy algorithm is a practical and efficient solution. It is also important to determine the number of proxies in order to balance the benefit of improved multicast performance and the cost of deploying and maintaining proxies. 3) For overlay link selection, we need to consider the trade-off between end user delay and tree cost. MST and Adjacent Connection produce better results than Complete Graph. 4) Slight over-dimensioning helps to improve the performance of the proposed simulation-based bandwidth dimensioning approach. In comparison with homogeneous groups, lower rejection ratio is achieved for heterogeneous groups by sacrificing high-bandwidth groups.

VI. CONCLUSIONS

In this paper, we have studied the MSON design problem. Our contributions can be summarized as follows.

- We formulate the difficult MSON provisioning problem, and divide it into three sub-problems: overlay proxy placement, overlay link selection, and bandwidth dimensioning.
- For each of these sub-problems, we propose several algorithms: an ILP formulation and a greedy algorithm for overlay proxy placement; Complete Graph, K-MST, and Adjacent Connection for overlay link selection; and a simulation-based approach for bandwidth dimensioning. We analyze the performance of these algorithms and identify the trade-offs among them.
- Through extensive simulations, we find that the performance of TOMA is significantly affected by different overlay provisioning algorithms.
- Based on our simulation and analysis results, we make the following suggestions for MSON design: the greedy algorithm is a very effective solution for overlay proxy placement; MST and Adjacent Connection produce better results than Complete Graph; over-dimensioning is needed for bandwidth

dimensioning in order to reduce the join request rejection ratio.

Future Work To make MSONs a reality, there are a large course of work to be done. In the future, we plan to study the impact of various network topologies and multicast group models (such as different group lifetime distributions) on the effectiveness of the proposed algorithms. In addition, we would consider the design of peer-to-peer trees in local clusters when MSONs are used as the backbone service overlays, and investigate the feasibility of using existing end system based approaches such as Narada [9] and NICE [4] in this case.

In short, we hope that our work in this paper could shed some light on the challenging MSON design problem and inspire some future interesting work in this direction.

REFERENCES

- [1] *AT&T IP Backbone*, 2001. <http://www.ipservices.att.com/backbone/>.
- [2] K. Almeroth. The evolution of multicast: From the MBone to inter-domain multicast to Internet2 deployment. *IEEE Network*, 14(1):10–20, Jan./Feb. 2000.
- [3] K. Almeroth and M. Ammar. Multicast group behavior in the internet’s multicast backbone (MBone). *IEEE Communications*, 35(6):124–129, June 1997.
- [4] S. Banerjee, C. Kommareddy, and B. Bhattacharjee. Scalable application layer multicast. In *Proceedings of ACM SIGCOMM*, pages 205–217, Aug. 2002.
- [5] 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*, volume 2, pages 1521–1531, Apr. 2003.
- [6] R. Chalmers and K. Almeroth. Modeling the branching characteristics and efficiency gains of global multicast trees. In *Proceedings of IEEE INFOCOM*, volume 1, pages 449 – 458, Apr. 2001.
- [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*, volume 2, pages 795 – 804, Mar. 2000.
- [9] Y.-H. Chu, S. G. Rao, and H. Zhang. A case for end system multicast. In *Proceedings of ACM Sigmetrics*, pages 1–12, June 2000.
- [10] J.-H. Cui, M. Faloutsos, D. Maggiorini, M. Gerla, and K. Boussetta. Measuring and modelling the group membership in the Internet. In *Proceedings of the ACM SIGCOMM/USENIX Internet Measurement Conference (IMC 2003)*, pages 65–77, Oct. 2003.

- [11] C. Diot, B. Levine, J. Lyles, H. Kassem, and D. Balensiefen. Deployment issues for the IP multicast service and architecture. *IEEE Network*, 14(1):78–88, Jan./Feb. 2000.
- [12] 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.
- [13] A. Fei, J.-H. Cui, M. Gerla, and M. Faloutsos. Aggregated Multicast with inter-group tree sharing. *Proceedings of NGC2001*, Nov. 2001.
- [14] P. Francis. *Yoid: Extending the Multicast Internet Architecture*. White paper, <http://www.aciri.org/yoid/>.
- [15] O. Goldschmidt and D. S. Hochbaum. Polynomial algorithm for the k-cut problem. In *Proceedings of IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 444 – 451, Oct. 1988.
- [16] 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*, pages 197–212, Oct. 2000.
- [17] B. M. Khumawala. An efficient branch and bound algorithm for the warehouse location problem. *Management Science*, 18(12):B718–B731, Aug. 1972.
- [18] A. A. Kuehn and M. J. Hamburger. A heuristic program for locating warehouses. *Management Science*, 9(4):643–666, July 1963.
- [19] L. Lao, J.-H. Cui, and M. Gerla. TOMA: A viable solution for large-scale multicast service support. In *Proceedings of IFIP Networking*, pages 906–917, May 2005.
- [20] Z. Li and P. Mohapatra. The impact of topology on overlay routing service. In *Proceedings of IEEE INFOCOM*, volume 1, pages 408–418, Mar. 2004.
- [21] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel. ALMI: An application level multicast infrastructure. In *Proceedings of the 3rd USENIX Symposium on Internet Technologies and Systems*, pages 49–60, Mar. 2001.
- [22] S. Shi and J. S. Turner. Routing in overlay multicast networks. In *Proceedings of IEEE INFOCOM*, volume 3, pages 1200 – 1208, June 2002.
- [23] S. Shi, J. S. Turner, and M. Waldvogel. Dimensioning server access bandwidth and multicast routing in overlay networks. In *Proceedings of NOSSDAV'01*, pages 83–92, June 2001.
- [24] N. Spring, R. Mahajan, and T. Anderson. Measuring ISP topologies with Rocketfuel. *IEEE/ACM Transactions on Networking*, 12(1):2 – 16, Feb. 2004.
- [25] M. Stoer and F. Wagner. A simple min-cut algorithm. *Journal of the ACM*, 44(4):585–591, July 1997.
- [26] A. Young, C. Jiang, M. Zheng, A. Krishnamurthy, L. Peterson, and R. Y. Wang. Overlay mesh construction using interleaved spanning trees. In *Proceedings of IEEE INFOCOM*, volume 1, pages 396–407, Mar. 2004.

Li Lao received her B.S. degree from Fudan University, China in 1998. She received her M.S. and Ph.D. degrees in Computer Science from University of California, Los Angeles in 2002 and 2006, respectively. She joined Google Inc in April 2006. Her research focuses on multicasting, overlay network management, multicast modeling and performance evaluation.

Jun-Hong Cui received her B.S. degree in Computer Science from Jilin University, China in 1995, her M.S. degree in Computer Engineering from Chinese Academy of Sciences in 1998, and her Ph.D. degree in Computer Science from UCLA in 2003. Currently, she is an assistant professor in the Computer Science and Engineering Department at University of Connecticut. Her research interests are in computer networks and data communications. They cover the protocol design, network modeling and performance evaluation of the Internet, wireless networks, sensor networks, peer-to-peer networks, and overlay networks. Currently, her research mainly focuses on multicasting and QoS support in wired and wireless networks, algorithm and protocol design in large-scale mobile wireless sensor networks, and mobility modeling and management in mobile adhoc networks. Please see <http://www.cse.uconn.edu/~jcui/> for her recent projects and publications.

Mario Geral received a graduate degree in engineering from the Politecnico di Milano in 1966, and the M.S. and Ph.D. degrees in engineering from UCLA in 1970 and 1973, respectively. After working for Network Analysis Corporation from 1973 to 1976, he joined the Faculty of the Computer Science Department at UCLA where he is now Professor. His research interests cover the performance evaluation, design and control of distributed computer communication systems; high speed computer networks; wireless LANs, and ad hoc wireless networks. He has worked on the design, implementation and testing of various wireless ad hoc network protocols (channel access, clustering, routing and transport) within the DARPA WAMIS, GloMo projects and most recently the ONR MINUTEMAN project. He is also conducting research on QoS routing, multicasting protocols and TCP transport for the Next Generation Internet (see www.cs.ucla.edu/NRL for recent publications).