Towards Economically Viable Infrastructure-based Overlay Multicast Networks

Varun Khare Beichuan Zhang
{vkhare, bzhang}@cs.arizona.edu
University of Arizona

***Abstract*—Internet-scale dissemination of streaming contents (*e.g.*, live sports games) can be achieved by infrastructure-based overlay multicast networks, where multicast service providers deliver the contents via dedicated servers strategically placed over the Internet. Given the huge amount of data traffic, one of the major operation costs is the ISP cost for network access. However, existing overlay multicast protocols only consider network performance metrics in building dissemination trees without taking into account the potentially high ISP cost they may incur. This paper presents a scheme, Revenue-driven Overlay Multicast Networks (ROMaN), to assign users to different servers in order to maximize the profit derived from providing multicast services. ROMaN exploits the fact that ISP charging functions are concave by assigning users to the cheapest available servers, and dynamically adjusts the assignment to accommodate the churns of group membership. The evaluation shows that ROMaN not only can reduce ISP cost substantially, but also has shorter end-to-end delay due to smaller overlay size, and the longer a user stays in the group the better the service it will receive.**

I. Introduction

Internet delivery of live streaming contents has surged to an unprecedented level. According to a white paper from Limelight [1], video-related Internet traffic is doubling every three to four months. For instance, IPTV is becoming the next emerging market in video delivery business. Currently, large-scale dissemination of streaming contents is done via overlay networks, either peer-to-peer (P2P) or infrastructure-based overlay multicast network (*OMN* henceforth). In P2P networks, group members self-organize into an overlay to forward contents from one to another, while in *OMN* a multicast service provider (MSP) places dedicated servers strategically over the Internet, and these servers form an overlay to forward content from the source to end users. *OMN* is appealing to commercial content providers as it allows greater control over the quality of service, accounting, and access control. In recent years, *OMN* made real-world impacts in broadcasting live sports games [2], political events [3], and entertainment shows [4] to millions of users over the Internet.

In order to be economically successful, a multicast service provider wants to maximize the number of users in the groups while minimize the cost in doing so. A major part of the operation cost [5] is the charges by Internet service providers (ISP) since an MSP needs to pay ISPs for the bandwidth used for multicast. Often ISPs charge business customers based on traffic volume, the more traffic a server sends the higher the charge is. Given the huge number of users an MSP needs to support, its ISP cost can run very high. Therefore, a challenging research question is how to build and manage the *OMN* in order to minimize the ISP cost while still maintain good network performance.

ROMaN reduces ISP cost by employing a novel user assignment scheme. Generally speaking, constructing data delivery paths in *OMN* involves two steps: (1) assigning users to certain servers, and (2) organizing servers of the same multicast group into a dissemination tree rooted at the source. Prior research work in *OMN* focused on the second part by proposing various protocols to build the server dissemination tree in order to optimize network performance metrics, such as end-to-end delay. Since the server infrastructure is geographically distributed and covers the major part of the distance between the source and the users, the server dissemination tree is the right place to optimize network performance. However, to minimize ISP cost, we need to turn our attention to user assignment, which has been largely ignored by prior work. A server usually connects to a few number of other servers but a large number of end users. Therefore, the bandwidth consumed by users is the dominant component in the total traffic volume that a server sends out. Therefore, user assignment is highly related to overall ISP cost, but most prior work simply assumes that users will be connected to the nearest server without further research on the impacts on ISP cost. The main idea of *ROMaN* is to exploit the economy of scale in ISP charging, that is, the marginal unit price decreases as the total traffic volume increases. For instance, if there are two servers located in two ISPs, assigning two users to the same server usually will cost less than assigning one user to each server. We formulate the user assignment problem and develop a heuristic solution which assigns users to the *cheapest available server*. Evaluation shows that this simple user assignment policy can reduce ISP cost significantly. In addition, the average end-to-end delay from the source to users is also reduced. This is because in *ROMaN*, users of the same group tend to be assigned to the same set of servers. Therefore the size of each group's server overlay becomes smaller than that under nearest server policy and the end-to-end delay becomes shorter.

ROMaN also employs a new user movement scheme to reward better service quality to users who stay in the session longer. Multicast group membership studies [6] have shown that users can be classified as *serious viewers*, who join the group for long duration, or *casual surfers*, who join the group

briefly and leave. Surfers are of large numbers and are the main cause for the group membership dynamics. Prior work has treated all users equally and adjust the dissemination tree as users join and leave, which may cause service interruption to existing users. In *ROMaN*, surfers are connected to leaf servers of the dissemination tree while viewers are allowed to advance towards the root server along the tree. Therefore, the churns in group membership will only cause changes to the leaf servers of the tree. The service interruption to viewers is minimized, and over time, viewers will experience shorter and shorter delay as they can move towards the root. This will make the service more attractive to serious users and lead to more revenue for the multicast service provider.

The rest of the paper is organized as follows. Section II introduces some background information on ISP charging, Section III formulates the user assignment problem and proposes solutions, Section IV describes the protocol design of *ROMaN* and Section V shows the evaluation results. We discuss related work in Section VI and concludes the paper in Section VII.

II. Background on ISP Charging

An MSP pays ISPs for providing network access at server sites. The choice of how ISPs charge the MSP depends the service level agreement between them, but usually it is a function of the bandwidth usage by the MSP. The most popular means to determine the bandwidth usage are percentile-based and total-volume based charging. In the former, the charge is determined by the 95-percentile of the traffic volume over a month, while in the latter the total volume of the monthly traffic is used. Regardless of how to get the bandwidth usage, a common property of the ISP charging functions is that they are usually concave, meaning that price per Mbps drops as the purchased bandwidth goes up. Our basic design take advantage of this property without assuming either percentile-based or total-volume based schemes. For the purpose of evaluation, this paper uses a total-volume based charging function, $c(r) = (\alpha - \beta \cdot \ln r) \cdot r$ where r is measured in Mbps and the price is a monthly fee in US dollars. The same function was also used in prior work [7], [8].

The multicast sessions as shown in Figure 1 are motivating examples to show the importance of user assignment problem. Group Membership studies [6], [9] have reported the existence of diversity in the user population of such multicast groups. Allowing user locality to determine user assignment forces a large number of servers to participate within the dissemination tree. However controlling user assignment policies can allow the same user population to be supported by fewer session servers as seen in Session A. And since the charging functions are concave in nature, assigning more users to a server becomes cheaper compared to assigning users to separate servers thereby reducing the overall ISP cost of deploying the group.

The marginal ISP cost of a server depends upon the charging function and the its immediate load. At any given point different servers can be offering different marginal ISP costs thereby providing an opportunity to optimize ISP cost of deploying a group. In session B the same user population is supported

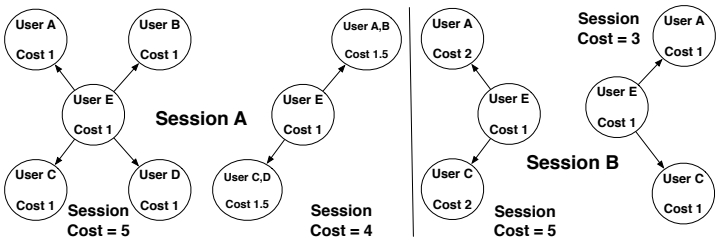

Fig. 1. Bandwidth costs of multicast groups organized in different ways.

by cheaper servers thereby reducing the overall ISP cost of deploying the group. Prior *OMN* protocols focus on optimizing network performance for users which in actuality incurs high ISP costs for deploying a multicast session. But efficient user assignment amongst servers can optimize ISP costs thereby improving profit margins for MSP serving as necessary incentive to make capital investment in such *OMN* systems.

III. User Assignment Problem

In order to facilitate a multicast session in *OMN*, users need to be assigned to dedicated servers which form an overlay network to disseminate content. The assignment of users to servers impacts (1) the cost of deploying a multicast group (2) network performance for users and (3) scalability of the *OMN* protocol. The server join policy of any *OMN* protocol dictates the user assignment amongst the servers. Prior *OMN* protocols [10], [11] propose that users statically join the *nearest server* and thereafter ask the nearest server to join groups as needed. In nearest server join policy the user locality not only determines the session servers facilitating the multicast session but also the bandwidth consumption underneath those servers. As a result the ISP cost of deploying a group gets directly tied to user locality.

Group Membership studies [6], [9] have reported the existence of spatial properties such as clustering and diversity in the user population of multicast groups. Clustering points out the skew in user population and diversity points out the large number of distinct locations where popular sessions are accessed. The nearest server join policy is unable to handle clustering which causes certain server locations to be overwhelmed thereby forcing users to be dropped in those locations. Dropping users affects the scalability of the *OMN* protocol which leads to loss in revenue for MSP. In order to alleviate the above problems we propose dynamic server join policies which can be used to distribute users amongst servers on demand. The *nearest available server* join policy allow users to join any server deployed nearby with available bandwidth and thereafter ask it to join the group. By allowing dynamically allocation of users to servers any kind of drop in users can be avoided which improves the scalability of the *OMN* protocol.

Existing approaches focus on optimizing network performance for users by carefully organizing the session servers with a given user assignment dictated by user locality. Diversity in user population forces a large number of session servers to participate in the dissemination tree. Therefore constructing a

tree for such large overlay size results in more overlay hops which increases the end-to-end delay performance for users.

ROMaN protocol solves the user assignment problem with the objective of minimizing the overall ISP cost of deploying a multicast session. In order to realize the above goal we propose another dynamic server join policy *cheapest available server* join policy which assigns users to the server with the least marginal ISP cost with available bandwidth. The problem is stated formally as: Given ISP charging function c_i and available bandwidth B_i of all the K servers deployed in *OMN*, find the user distribution u_i at each server SRV_i for N number of interested users of the multicast group where each user consumes bandwidth b, such that $\sum_{i=1}^{K} u_i = N; u_i \cdot b \leq B_i; and \sum_{i=1}^{K} c_i(u_i \cdot b)$ the ISP cost of facilitating the multicast group is minimized. From henceforth user distribution refers to physical locality of users underneath the servers and user assignment refers to the assignment of user to server in accordance with a particular server join policy.

A. Offline Dynamic Programming Solution

The *ROMaN* user assignment problem can be solved using the dynamic programming approach since the optimal solution to distribute N users amongst K servers contains within it the optimal solution to the sub-problem of distributing n users amongst k servers where n≤N and k≤K. Let cost(n, k) be the optimal cost for allocating n users amongst k servers.

$$cost(n,k) = \begin{cases} c_1(n.b) & k = 1, B_1 \geq n \cdot b \\ \infty & k = 1, B_1 < n \cdot b \\ \infty & k > 1, \sum_{i=1}^{k} B_i < n \cdot b \\ \min\limits_{0<i\leq n} \; cost(n-i, k-1) + c_k(i \cdot b) & k > 1 \end{cases}$$

We start by evaluating cost(n,1) where n=1,...,N which is the base case where no choice of servers is available since k=1. Thereafter evaluating cost(n,2) where n=1,...,N considers the necessary sub-problems to figure out the cost optimal distribution of n users amongst k=2 servers. Eventually evaluating cost(N,K) gives the optimized cost of deploying the multicast group. Tracking the user distribution of the sub-problems allows us to find the user distribution for cost(N,K), the solution to the user assignment problem. Any server with $u_i \neq 0$ serves as session server for the multicast group.

The runtime of the algorithm is $O(K.N^2)$ and the space complexity is O(K.N). The algorithm looks at all the sub-problems in the space of O(K.N). In a multicast group N changes rapidly since users are constantly trying to join and leave. Every increment, say ΔN, requires $O(K \cdot \Delta N)$ computation to evaluate the cost-optimal user assignment. Therefore the dynamic programming approach is computationally slow for *OMN* due to the group dynamics involved in multicast groups.

B. Online Iterative Greedy Heuristic

Evaluating all the sub-problems as done in the Offline Dynamic Programming approach is not necessary when we consider the concave nature of the ISP charging functions. Servers can be relatively cheaper depending upon their respective marginal ISP cost. In order to maintain the cost efficiency

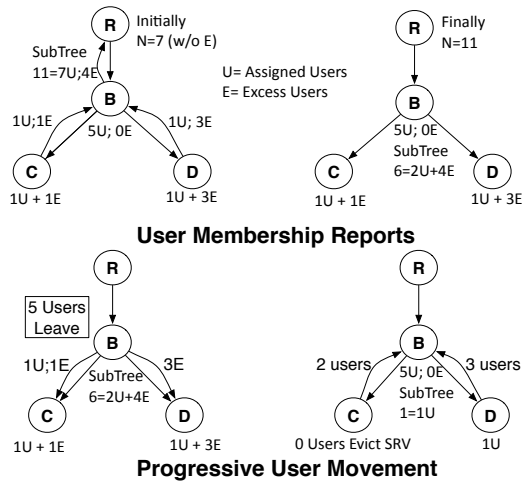

Fig. 2. Session servers regularly forward total count of users ($\triangle_{c,i}$) equal to assigned and excess users ($\triangle_{e,i}$) towards root. User deficiency at non-leaf servers satisfied by absorbing users within the sub-tree using the Progressive User Movement Scheme

of the user distribution, the root server can assign users to server with available bandwidth and least marginal ISP cost defined as the cheapest available server. The greedy strategy does not always produce the optimal cost efficient user distribution but reduces the runtime complexity significantly for handling group dynamics of multicast groups.

In order to assign users to the cheapest available server the root server maintains a list of servers ordered by their respective marginal ISP costs. The ISP charging functions intersect amongst each other pointing out the regions of contention regarding which server is cheaper for user assignment. Each pair of ISP charging function intersect at unique points i.e. ISP cost functions $c_i(x) = (\alpha_i - \beta_i \cdot \ln x) \cdot x$ and $c_j(x) = (\alpha_j - \beta_j \cdot \ln x) \cdot x$ of SRV_i and SRV_j respectively, intersect uniquely at $INT_{i,j} = e^{\frac{\alpha_i - \alpha_j}{\beta_i - \beta_j}}$. At the point of intersection the cheaper server is evaluated by comparing marginal ISP costs. For e.g. $(\alpha_i - \beta_i) - \beta_i \cdot \ln x < (\alpha_j - \beta_j) - \beta_j \cdot \ln x$ where $x = INT_{i,j}$ implies SRV_i to be relatively cheaper than SRV_j. Given the K concave cost functions associated with deployed servers the maximum number of possible intersections are $O(K^2)$. Initially the server list [SRV] is ordered by the initial marginal ISP cost of servers. For each server SRV_i an intersection list, INT_i, is maintained which captures the intersection point of c_i with other charging functions. Thereafter following recursive algorithm is used to assign n users to the [SRV] list starting at SRV_i:

$$assign(n,i) = \begin{cases} u_i += n & n < INT_{i,j} \\ u_i += n - INT_{i,j} & n > INT_{i,j} and \\ Swap SRV_i, SRV_j & c_i^{'}(INT_{i,j}) > c_j^{'}(INT_{i,j}) \\ assign(n - INT_{i,j}, i) & \\ u_i += B_i/b & n > B_i/b \\ assign(n - B_i/b, i+1) & \end{cases}$$

The root server starts by assigning users to the SRV_0 and gradually exhausts the list. The runtime of the above algorithm

is O(K^2) for sorting the initial list of servers with respect to their marginal cost. Thereafter the cost to assign any number of users is O(1) which is suitable for multicast group dynamics.

Controlling the allocation of users allows us to manage the ISP cost, spread the load amongst unused servers and increase the feasibility of future groups. By allowing all deployed servers to potentially participate and support users significantly increases the feasibility of the group. In prior *OMN* protocols the session servers are restricted by user locality which limits the maximum cumulative bandwidth available to support the group thereby reducing its feasibility.

IV. *ROMaN* PROTOCOL

In this section we present the *ROMaN* protocol which incorporates the cheapest available server join policy for the users. The root server, serving the source content for the group, commissions and/or decommissions session servers while facilitating the dissemination for the group. The root server, given the overall user membership N, calculates using the Online Greedy Algorithm *target user assignment* for each server which if maintained ensures the cost-efficiency of the group. (For groups with long session duration the root can also use the Offline Dynamic Programming approach, from time to time, to improve the target user assignment of servers to make the group even more cost-efficient.) The session servers attempt to maintain the desired target user assignment by following the Progressive User Movement Scheme. Periodically root server estimates N, but allows user assignment for γ·N to servers in the form of updated target user assignment where $\gamma \geq 1$ thereby pre-calculating cheapest available server for new users. This allows the root server to appropriately treat new users depending upon their class distinctions (surfers and viewers). In order to assist with the estimation of N, regular *user membership reports* are circulated bottom-up from the leaf session servers to the root server as shown in Figure 2. These user membership reports include $\triangle_{C,i}$ which is the cumulative number of users being serviced in the sub tree rooted at SRV_i.

The root server upon finding the leaf session servers to be near saturation point pertaining to a threshold value (=$\delta \cdot$ B_i where $\delta \leq 1$) commissions the next cheapest available server. The cheapest available server, figured by the Online approach, joins the dissemination tree following the HMTP [12] join protocol. The joining server begins by trying to join underneath the root server directly. Depending upon the available bandwidth root server either allows the join or directs the joining server to its child servers. In case the join is unsuccessful at root server the joining server retries the join at the nearest child server of the root. The joining server keeps repeating the above process in search of a parent server with available bandwidth which is closest to it. The root server supplies an initial target user assignment to the newly added session server which is updated regularly as stated earlier.

User leave operations are unpredictable but recent studies [6] conducted over live streaming workload from large content providers such as Akamai reports the existence of surfers and viewers. The session servers primarily serving surfers are more susceptible to eviction during the lifetime of a multicast session. The disruption caused by the eviction of these session servers can be prevalent if they belong to the non-leaf part of the data delivery tree. A desirable property for an overlay structure is to have minimum disruption even in the face of membership changes. We motivate the need for a user join and leave protocol which abides by the *ROMaN* user allocation scheme and minimizes change in the dissemination tree due to any membership changes.

Algorithm 1 Pseudo-code for SRV_X to satisfy deficiency of users

```
Progressive User Movement
△_e(X) local deficiency of users at SRV_X
△_E(X) cumulative excess users in sub tree of SRV_X
if △_E(X) ≥ △_e(X) then
    // Local deficiency covered by excess users in sub-tree
    Extract users from child servers depending upon △_E(child)
    △_E(X) = △_E(X) − △_e(X)
else
    if △_E(X) ≥ 0 then
        // sub-tree has some excess users
        Extract users from child servers depending upon △_E(child)
        △_E(X) = △_E(X) − △_e(X)
        Split remaining △_E(X) equally amongst child servers
    else
        Split △_e(X) equally amongst child servers
    end if
end if
```

A. *ROMaN User Join / Leave Protocol*

The root server acts as the rendezvous point for the entire group where user join requests are handled. The root server re-routes the user join request to the leaf session server SRV_I with the least amount of excess users $\triangle_{E,I}$. The excess users, $\triangle_{E,I}$, is defined as the extra users over the target user assignment being serviced by a session server. Each server maintains the aggregate excess user membership for all the leaf servers in its sub-tree. The information is aggregated bottom-up and exchanged as part of the periodic user membership reports as shown in Figure 2. By injecting users at the leaf session servers, the root server is able to maintain all the excess users at the leaf level. Initially all the users are treated as surfers and therefore injected at the leaf level with maximum delay.

The leaf session servers are sources of excess users whereas non-leaf session servers are sinks where a deficiency of users is caused by user leave operations. Any deficiency of users caused at non-leaf session server is satisfied by transferring users from the sub tree rooted at that session server. The user deficient server uses the excess user membership and cumulative user membership information of its sub-tree to split the needed user requests between its child session servers as described in the Algorithm 1 pseudo-code for Progressive User Movement Scheme.

Initially the user deficient server tries to absorb the excess users available at the leaf session servers in its sub-tree and thereafter splits the remaining user requests amongst all its

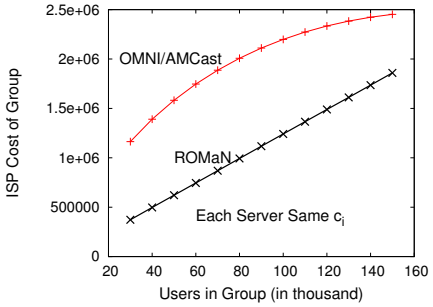

Fig. 3. *ROMaN* shows ISP cost improvement for servers deployed within an individual ISP

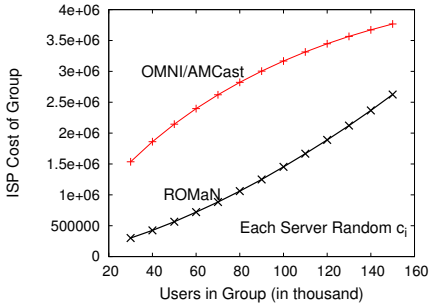

Fig. 4. *ROMaN* shows ISP cost improvement in the case of servers deployed across ISPs

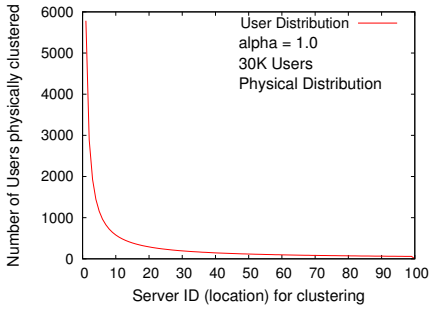

Fig. 5. Zipf distribution clusters users underneath few servers and diversifies rest

child session servers. All the user movement is implemented in a distributed fashion through local interactions between parent and child servers. The user movement is triggered at non leaf session servers but eventually ripples its way towards the leaf session servers in the sub tree as shown in the Figure 2. Each server maintains the local users served by it in a FIFO list to ensure that users with maximum session duration, i.e. local viewers, are given preference when it comes to user transfer requests towards the parent server. Such progressive movement of potential viewers towards the root server improves their network performance. Furthermore the above scheme only leaves leaf servers with any deficiency of users if at all. Therefore at any given time in the session duration only leaf servers are candidate for eviction in case they don't support any users. The leaf servers not serving any users inform the root server of their leaving after a grace period. The root server de-commissions all such leaf servers which no longer support any users. Evicting and/or adding only leaf servers at a time minimizes the disruption to the overlay tree caused by any kind of change in user membership.

B. Discussion

ROMaN does not explicitly try to reduce the end-to-end delay for users. However, as the evaluation results will show, the actual delay in *ROMaN* turns out to be shorter than that in prior *OMN* protocols. The reason lies in the size of the overlay. Prior work assumes that users connect to the nearest servers, therefore a group with a large number of users usually will have many session servers to form the overlay dissemination tree. In *ROMaN*, users of the same group tend to be clustered to fewer session servers as a result of the cheapest available server heuristic. Therefore, a smaller overlay in *ROMaN* can gives shorter delay than a bigger overlay built by prior approaches.

ROMaN requires knowledge of available bandwidth at each deployed server in order to evaluate the marginal ISP costs accurately. The available bandwidth at each server is subject to change due to group dynamics of multiple multicast sessions. We assume this knowledge will be available from the service provider's Network Operations Command Center (NOCC), which continuously monitors the state of the server network. Removing such dependency is something which we will be looking at in the future work. Another simplification of *ROMaN* is that we assume the charging function only depends on the traffic volume. However, as discussed in an Akamai [13] patent, ISPs usually charge more for traffic that goes out of their networks than traffic that stays within their networks, the so-called "off-net" versus "on-net" traffic. Taking this into consideration will also be an important extension to *ROMaN*.

V. Evaluation

In *OMN*, MSP strategically deploys servers within network ASes to get closer to users. In order to simulate *OMN*, AS-level topological map [14] is used where individual ASes are treated as physical routers, to which servers and users can be attached. Latency between the servers is the unicast path latency between the connected ASes. Inter AS-links [14] are assigned random delay within certain range based on which inter AS unicast path latency varies between 150ms and 500ms. We compare the performance of *ROMaN* against prior *OMN* protocols such as *OMNI* [10] and *AMCast* [11] on various metrics through detailed simulation experiment.

A. ISP Cost Saving

ISP cost is an important metric which directly affects the overall profit margins. Figure 3 compares the ISP cost of deploying groups increasing in size by *OMN* protocols, with servers being present within an individual ISP. Each server is assigned the same concave charging function and fixed bandwidth. Users are evenly distributed underneath the servers to avoid the impact of spatial properties, such as clustering in user locality, upon the performance of *OMN* protocols. ISP cost of group depends upon the traffic volume at server sites which is dominated by the bandwidth consumed to support users. *OMNI* and *AMCast* are unable to influence ISP cost by tying the bandwidth consumption in supporting users to user locality. Since users are evenly distributed, each server is forced to contribute towards the overall ISP cost of group. And as group size increases the ISP cost grows logarithmically similar to the nature of the underlying charging functions at each server. In comparison *ROMaN* maintains a linear increase in ISP cost as it saturates linearly increasing number of servers. Since each server is associated with the same charging function it is cheaper to saturate a server before assigning users to other unused servers.

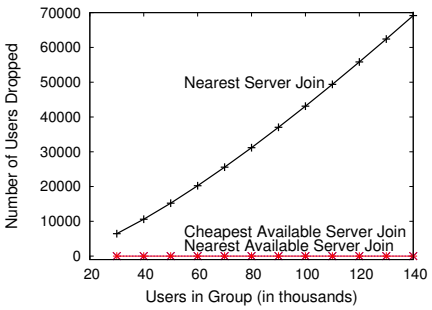

Fig. 6. Static Nearest Server Join Policy unable to handle clustering which causes user drop

Fig. 7. *ROMaN* minimizes ISP cost and avoids user drop thereby maximizing revenue generated

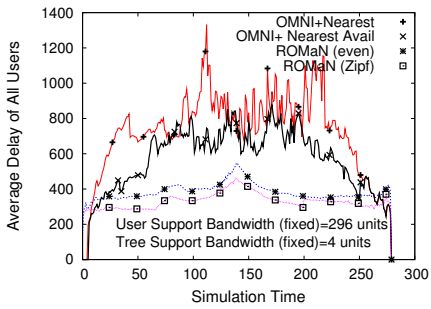

Fig. 8. *ROMaN* by reducing overlay tree size significantly improves user delay performance

Figure 4 compares the ISP cost when servers are deployed in different ISPs. Individual ISPs can assign different charging functions to the servers depending upon factors which are beyond the scope of this paper. To simulate the above fact servers are assigned charging functions generated by randomly choosing parameters such that concavity is maintained. For *OMNI* and *AMCast*, ISP cost grows logarithmically as before due to the inability to influence user assignment to servers. However *ROMaN* shows super-linear rise in ISP cost since initially user assignment is made to cheap servers but as group size increases progressively costly servers are eventually needed to facilitate the group. In either scenario *ROMaN* outperforms *OMNI* and *AMCast* when comparing ISP cost of deploying a group. *ROMaN* presents the potential to reduce ISP cost if included in the Request Re-routing mechanism [15] of MSPs to do user assignment for servers deployed within an individual ISP or several ISPs.

In Figure 3 and 4, for small group sizes the difference in ISP cost between the *OMN* protocols is significant but as group size increases the difference gradually decreases. The amount of optimization in ISP cost by *ROMaN* is dependent upon the available bandwidth at the server sites. Initially servers have more available bandwidth thereby presenting *ROMaN* with better opportunities to optimize ISP cost as reflected in the case for small group sizes. But as servers reach near saturation point due to facilitating groups of increasing size, less room is available to *ROMaN* for optimizing ISP cost as reflected in the case for larger group sizes. As a matter of fact near saturation point the difference between user assignment of *ROMaN* and user distribution followed by *OMNI* and *AMCast* is minimal due to which difference in ISP cost is minimal.

B. Scalability of OMN protocols

Scalability of an *OMN* protocol is the ability to handle spatial properties such as clustering and diversity in user locality, as reported by Group Membership studies [6], without dropping users. In order to simulate spatial properties users are distributed underneath server locations following Zipf-distribution. Zipf-distribution allows some users to be clustered and remaining spread to capture diversity as shown in Figure 5. Servers are allocated fixed bandwidth, enough to support an even user distribution, in order to capture how *OMN* protocols handle spatial properties.

Figure 6 compares user drop rate when groups increasing in size are deployed with *ROMaN* implementing the cheapest available server join policy and *OMNI* and *AMCast* implementing both the nearest server and the nearest available server join policies. The nearest server join policy quickly saturates server locations where users are clustered thereby forcing users to be dropped. As group size increases so does the effect of clustering which results in a near linear rise in user drop rate. Interestingly due to diversity all deployed servers are involved in dissemination with bandwidth to spare but the static nature of the nearest server join policy is unable to allocate dropped users to these other unused servers. Both the nearest available and cheapest available server join policies avoid dropping users by allowing dynamic assignment of users to unused servers.

Figure 7 compares the profit margins from deploying above mentioned groups with each user generating fixed revenue. Profit margin of a group depends upon number of users being serviced and the ISP cost. By dropping users, revenue is lost due to which nearest server join policy overall generates least profit margins. The nearest available server join policy produce next best profit margins by avoiding drop in users but the ISP cost remains unchecked. *ROMaN* through cheapest available server join policy produces maximum profit by optimizing ISP cost and avoiding any user drop.

C. User Delay Performance during session duration

User delay performance is another key metric which affects the acceptability of any *OMN* protocol. Figure 8 compares the average delay of users between *OMN* protocols using different server join policies. *ROMaN* (even) and *OMNI* (Nearest server join) correspond to the case where users are evenly distributed and *ROMaN* (Zipf) and *OMNI* (Nearest Available server join) correspond to the case where users are distributed using Zipf distribution. Servers are allocated fixed tree and user support bandwidth in order to avoid their impact upon user delay performance, which we study in Section V-E. In order to simulate change in group membership users are randomly chosen to join during the join phase and thereafter leave during the leave phase. Snapshots of the average delay of users in the system is taken whenever there is more than 5% change in group membership.

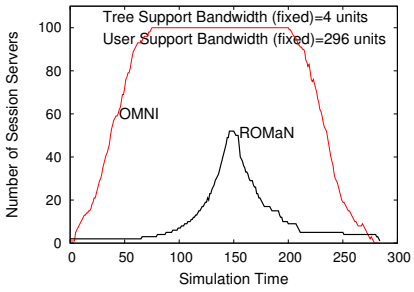

Fig. 9. *ROMaN* maintains reduced overlay tree size irrespective of the change in user membership

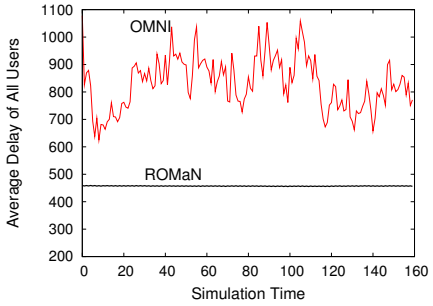

Fig. 10. *ROMaN*, by limiting surfers to leafs in overlay tree, is able to provide near-constant delay performance to all users

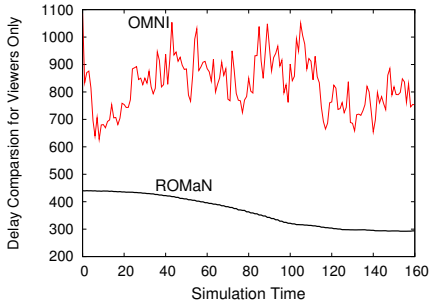

Fig. 11. *ROMaN*, by constantly moving potential viewers towards root is able to provide improved delay performance to viewers

ROMaN in either case presents nearly 50% improvement when compared to *OMNI* which focuses on optimizing user delay performance. The user delay can be split into two parts: (1) last-hop delay and (2) overlay tree delay. Overlay tree delay is definitely the dominant one and is dependent upon the number of overlay hops from root to destination. *ROMaN* by concentrating user assignment amongst fewer servers as shown in Figure 9 is able to reduce overlay hops which in-turn improves overlay tree delay. *OMNI* optimizes delay for any given user distribution underneath the session servers. However nearest server join policy when combined with diversity in user locality causes an increase in the overlay size. Therefore *OMNI* is optimizing delay on oversize overlays where the extra overlay hops make the tree delay sub-optimal even though the last-hop delay is minimized. Furthermore as users join and leave, *ROMaN* is able to dynamically adjust the overlay size which keeps in check the overlay tree delay. But with *OMNI* the overlay size remains near constant for most of the session duration since the overall user locality changes much slowly with respect to the overall user membership.

D. Delay Performance of viewers

Viewers are defined as the top 5 percentile users with longest session viewing duration. The servers with fixed dimensioned bandwidth as before are deployed with even user distribution. Initially one-third of user population randomly join the multicast session forming candidate viewers. Thereafter remaining users are randomly chosen to join session at regular intervals as candidate surfers and at the same time users are randomly chosen to leave session. The scenario simulates change in user membership at random server locations in order to demonstrate how *OMN* protocols handle it.

Figure 10 presents the average delay experienced by all users during the session duration. The user delay is influenced by the way protocols handle change in user membership. In *OMNI* user delay keeps fluctuating while in *ROMaN* it remains near constant. *OMNI* proposes local tree transformations where servers switch positions within the tree to handle the change. Each server is associated with a weight capturing the users it is servicing. Change in user membership causes these weights to shift which triggers the appropriate tree transformation. Surfers, the main cause of such change, are distributed everywhere due to the nearest server join policy thereby causing the weights underneath every server to change. Depending upon the joining and leaving behavior of surfers these weights could potentially change both frequently and constantly. Finally the local tree transformations triggered by these changing weights is the underlying cause of fluctuation in user delay. *ROMaN* due to progressive user movement avoids such fluctuations by limiting surfers to the leaf level. Moreover any change to the tree is limited to the leaf i.e. session servers are added and/or removed on-demand but only at the leaf level leaving the entire tree unaffected.

Figure 11 presents the average delay experienced by viewers during the session duration. As before in *OMNI* the viewer delay keeps fluctuating whereas in *ROMaN* delay not only remains near constant but gradually decreases. *OMNI* forces local tree transformations between servers without considering the kind of users they service due to which delay for viewers keeps fluctuating. In *ROMaN* progressive user movement allows only potential viewers to move closer towards the root as users leave from non-leaf session servers resulting in a gradual improvement in their network performance.

E. Effect of Bandwidth Dimensioning on user delay performance

The bandwidth of each server is useful to (1) service users, referred to as user support bandwidth and (2) participate in dissemination tree, referred to as tree support bandwidth. *OMNI* and *AMCast* are only concerned with the efficient usage of tree support bandwidth. The users are expected to self-organize themselves underneath the servers through either *ESM* protocol [16], [17], [18], [19], [20], [21] or *P2P* protocol [22], [23], [24] solutions. Current streaming applications expect users to be serviced directly by servers which makes it important to efficiently use both user and tree support bandwidth. As it turns out both factors affect the delay performance of users in different ways.

Figure 12 compares the average delay of users when user support bandwidth is fixed and tree support bandwidth is varied with users being evenly distributed. Tree support bandwidth affects the number of children a non-leaf node can have in

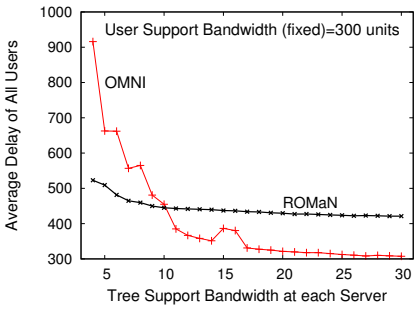

Fig. 12. Tree Support Bandwidth impacts user delay performance by reducing depth of overlay tree

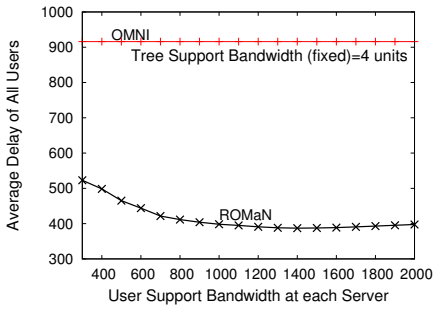

Fig. 13. User Support Bandwidth impacts user delay performance by reducing overlay tree size

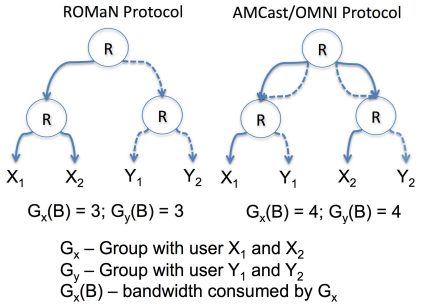

Fig. 14. Same groups G_x and G_y organized in different ways can cost extraneous Tree Support Bandwidth

the overlay tree which in-turn impacts the depth of the tree. For e.g. given a K-ary tree with k>2 has smaller depth as compared to a binary tree of the same size. In *OMNI* increasing tree support bandwidth has greater impact upon the depth of the oversize overlays which causes user delay to improve drastically. In comparison for *ROMaN* the change in user delay is less significant since the overlay size is much smaller which reduces the effect of increasing tree support bandwidth.

Figure 13 compares the average user delay when tree support bandwidth is fixed and user support bandwidth is varied. Increasing user support bandwidth allows more users to be assigned to a server. *ROMaN* is able to exploit the above fact and reduce the size of the overlay thereby improving the delay for users. In comparison *OMNI* remains unaffected by the increase in user support bandwidth since the overlay size, which is dependent upon user locality, remains unchanged.

In prior work [25], [26] bandwidth dimensioning problem deals with the overall assignment of bandwidth to servers so as to meet the group demands. As far as we know this effect of bandwidth dimensioning upon user delay experience has not been reported in any of the earlier studies. The ratio between tree support and user support bandwidth is a tuning factor which is dependent upon the requirements of applications making use of the service.

F. Rejection Rates

Rejection rate is the maximum number of groups possible within *OMN* given the bandwidth allocations of servers. The rejection rate is evaluated by deploying the same group again and again in order to create localized bottlenecks. *OMNI* faces a rejection rate of 1:19 i.e. a group is rejected due to unavailability of resources after deploying 19 groups similarly for *AMCast* it is 1:336 and finally for *ROMaN* it is 1:998. Rejection rate is dependent upon the manner in which *OMN* protocols consume tree support bandwidth for deploying groups. Consider the scenario in Figure 14 where the same groups are organized using different *OMN* protocols. In the case of *OMNI* and *AMCast*, the oversize overlays for G_x and G_y need extra overlay edges which in-turn forces unnecessary consumption of tree support bandwidth. *ROMaN* by focusing user assignment on smaller overlay size avoids the extraneous bandwidth consumption. As a result with *OMNI* and *AMCast* the overall tree support bandwidth available in the system exhausts faster when compared to *ROMaN* causing future groups to be rejected. *OMNI* furthermore exhausts bandwidth of root and nearby servers to minimize delay which in-turn makes them bottlenecks for future groups resulting in such low rejection rates. *AMCast* improves rejection rate by a balanced bandwidth allocation scheme for each group but the extra overlay edges eventually limit the number of possible groups. Only *ROMaN* through efficient usage of tree support bandwidth is able to facilitate so many groups.

VI. Related Work

IP multicast as a means to provide multicast data delivery for large-scale group communication applications is still plagued by several practical challenges [27]. End System Multicast (*ESM*) [16] was proposed as an alternative to IP Multicast but it suffers from the drawback of scalability i.e. only being able to support medium sized groups. End systems participate in the multicast group via an overlay structure constructed on an underlying mesh of end systems which is continuously improved through passive measurement of network dynamics. Scattercast [18], Yoid [19] and ALMI [20] are examples of such end-system multicast protocols which consider delay as the performance metric. More recently, application-layer multicast protocols have been proposed which exploit the underlying routing infrastructure of peer-to-peer (P2P) systems. Bayeux [23], Scribe [24], CAN-based multicast [22] are such application-layer multicast protocols which make adequate usage of the self-organizing peer-to-peer location and routing substrate provided by Tapestry [28], Pastry [29], CAN [30] respectively. In the global environment partitioned by P2P protocols, nodes interested in multicast groups form mini P2P rings to facilitate groups.

Diametrically opposite to these approaches is the *OMN* architecture where strategically deployed servers act as proxies facilitating multicast groups. The advantage to this architecture is end users send or receive only one copy of data packets during session, and the work of duplicating packets is shifted from data sources to servers. Researchers [25], [31], [26] have treated the deployment of such architectures as a

network design problem taking into account ISP operational costs. They have divided it into three sub-problems: server placement, overlay link selection and bandwidth dimensioning. MSON [25] identifies these sub problems and proposes separate solutions. TOMA [31] proposes a two-tier overlay multicast architecture which advocates MSON as the backbone service domain with the end users being present in the access domain forming small clusters underneath deployed servers. But *OMN* protocols focus on deploying multicast groups once the initial *OMN* infrastructure has been deployed with due consideration given to particular performance metrics. *AMCast* [11] and *OMNI* [10] protocols are state-of-art *OMN* protocols used to organize the servers to facilitate the multicast service for end-users. *AMCast* works through a centralized algorithm which optimizes tree support bandwidth usage at server sites in order to maximize the number of groups that can be deployed. *OMNI* presents a distributed algorithm to modify the overlay tree with the objective of maintaining optimized user delay performance in the face of change in network conditions and user memberships.

VII. Conclusion and Future Work

We presented cost-efficient user assignment scheme which minimizes the ISP cost of deploying groups within the *OMN* infrastructure. Decoupling the users locality from session servers allows us to control the overall ISP cost of deploying multicast groups and provide a mechanism to balance the user load amongst the servers. By considering more dynamic server join policies *ROMaN* protocol along with prior *OMN* protocols can handle any kind of skew in the user population. By optimizing ISP cost and avoiding any user drop *ROMaN* protocol is able to provide higher profit margins to the MSP. By reducing the overlay size *ROMaN* protocol is able to reduce the overall delay experienced by users in almost half. The progressive user movement scheme allows *ROMaN* protocol to provide a stable network performance to its end users. Distinguishing users allows *ROMaN* to reward viewers with performance benefits and limit the disruptive behavior of surfers. Treating users separately upon the basis of session duration is something which hasn't been considered by any previous work as far as we know. We have conducted experiments under realistic user distribution settings as reported by previous studies. The *ROMaN* protocol performs well across a range of user distribution settings and continues to present improved revenue results for the overall system and better overall network performance for its end users.

References

[1] M. Gordon, "The Internet Streaming Media Boom," White paper, Limelight Networks, http://www.limelightnetworks.com/index.

[2] "March madness On-Demand," http://www.sportsline.com/info/about/press/2008/feb08mmod.

[3] "The great Obama traffic flood," http://asert.arbornetworks.com/2009/01/the-great-obama-traffic-flood.

[4] "Online video: Must-free TV," CNN Money, 2006, http://money.cnn.com/2006/04/10/news/companies/abconline_free/index.

[5] G. Pallis and A. Vakali, "Insight and Perspective for Content Delivery Networks," in *Communications of the ACM*, 2006.

[6] K. Sripanidkulchai, B. Maggs, and H. Zhang, "An analysis of Live Streaming workloads on the Internet," in *Proceedings of Internet Measurement Conference (IMC)*, 2004.

[7] Y. Zhu, C. Dovrolis, and M. Ammar, "Combining multihoming with overlay routing," in *Proceedings of IEEE INFOCOM*, 2007.

[8] P. M. Ferreira, "Implications of decreasing bandwidth price on allocating traffic between transit and peering aggreements," Master's thesis, Massachusetts Institute of Technology, 2002.

[9] J. hong Cui, M. Faloutsos, D. Maggiorini, M. Gerla, and K. Boussetta, "Measuring and modelling the group membership in the Internet," in *Proceedings of Internet Measurement Conference (IMC)*, 2003.

[10] 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*, 2003.

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

[12] B. Zhang, W. Wang, S. Jamin, D. Massey, and L. Zhang, "Universal IP multicast delivery," *Journal on Computer and Telecommunications Networking*, 2006.

[13] "Akamai powering a better Internet," http://www.akamai.com/.

[14] "Internet Topology Collection," http://irl.cs.ucla.edu/topology.

[15] R. D. Day, "Meta content delivery network system," *US Patent 7185052*.

[16] Y.-H. Chu, S. G. Rao, and H. Zhang, "A case for End System Multicast," in *Proceedings of ACM Sigmetrics*, 2000.

[17] Y. hua 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*, 2001.

[18] Y. Chawathe, "Scattercast: an architecture for Internet broadcast distribution as an infrastructure service," *Ph.D Thesis, U.C. Berkeley*, 2000.

[19] P. Francis, "Yoid: Extending the Internet multicast architecture," Cornell, Tech. Rep., 2000.

[20] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An application level multicast infrastructure," in *Proceedings of USITS*, 2001.

[21] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. Jr, "Overcast: Reliable multicasting with an overlay network," in *Proceedings of OSDI*, 2000.

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

[23] 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*, 2001.

[24] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron, "Scribe: A large-scale and decentralized publish-subscribe infrastructure," *IEEE Journal on Selected Areas in communications (JSAC)*, 2002.

[25] L. Lao, J.-H. Cui, and M. Gerla, "Multicast service overlay design," in *Proceedings of SPECTS*, 2005.

[26] S. Shi, J. Turner, and M. Waldvogel, "Dimensioning server access bandwidth and multicast routing in overlay networks," in *Prceedings of NOSSDAV*, 2001.

[27] C. Diot, B. N. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture," IEEE Network, Special Issues on Multicasting, 2000.

[28] B. Zhao, J. Kubiatowicz, and A. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," UC Berkeley, Tech. Rep. UCB/CSD-01-1141, 2001.

[29] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *Proceedings of IFIP/ACM Middleware*, 2001.

[30] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content addressable network," in *Proceedings of ACM SIGCOMM*, 2001.

[31] L. Lao, J.-H. Cui, and M. Gerla, "Toma: A viable solution for largescale multicast service support," in *Proceedings of IFIP*, 2005.