A Helper Based Clustering Algorithm for Multicast Data Delivery

Jun Wei, Lixia Zhang
Computer Sciences Department, UCLA
4403 Boelter Hall, Los Angeles, CA 90095
E-mail: jun@cs.ucla.edu, lixia@cs.ucla.edu
April 17, 2000

Abstract— **Multicast distribution trees usually contain multiple bottleneck points. Due to different bandwidth constraints at these bottleneck points, multicast users experience different throughput or packet loss characteristics. It has been a great research challenge to handle the heterogeneity among multiple users in a large multicast group, where individual users may request different data transmission rate and recovery of different packet losses. We propose to cluster users into separate subgroups by common helpers. A helper is a node that can relay the data to a subgroup either at a reduced rate or by a different data encoding scheme tailored to suite the bandwidth constraints of the subgroup. Thus the helper can fulfill specific requests from the subgroup members on behalf of the original data source. Candidate helpers announce their availability by an advertisement scheme, which allows users to find the nearest helper. A novel scope control scheme is introduced to maintain low advertisement overhead. Simulation experiments show that the scope control scheme can effectively reduce the advertisement overhead under different topologies and group sizes.**

I. INTRODUCTION

IP multicast has provided efficient network layer mechanism for delivering content to large number of users. In the mean time, it imposes great challenges for multicast sender to handle multiple users. One of the challenges is associated with multiple bottleneck points of the multicast distribution tree. A multicast distribution tree connects the sender to the multiple users. Along the path from the sender to each user, one link will become the bottleneck point of that path when its bandwidth is lower than that of other links along the same path. In Figure 1, link L_c is such a bottleneck point on the paths from sender S to user R_1 and R_2. Unlike single bottleneck point along the unicast delivery path, a multicast tree may have multiple bottleneck points. For example, the tree in Figure 1 has two bottleneck points. Multiple bottleneck points introduce a number of new problems for multicast. For example, at what rate(s) should a sender transmit the data along different paths, and how should a sender adjust the transmission rate? Because different packets may get dropped at different bottleneck points, it is likely that users of the same multicast session may experience different packet losses. How should a sender handle different loss repairs? We call these problems multicast heterogeneity problems.

Some of the heterogeneity problems can be solved by clustering users into separate subgroups. A clustering example is illustrated as shown in Figure 1. The sender S uses a layered multicast delivery such as RLM [9], [15].

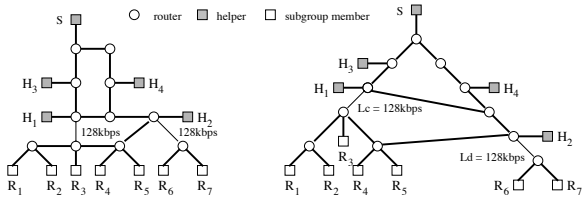

Fig. 1. (a) The topology of a multicast session with sender at S. (b) The multicast distribution tree for the multicast session in (a). There are two bottleneck points in (b). The bottleneck bandwidth is 128kbps for both.

Due to the common bottleneck link L_c, users R_1 and R_2 both suffer severe packet losses. To reduce the loss rate, they subscribe to fewer data layers than users H_1 or H_3, therefore, they cluster into a slow subgroup. A nearby user H_1 helps to relay higher data layers at a later time. Another example of clustering is the local recovery group for SRM in [8]. In the same multicast session shown in Figure 1, users R_1 and R_2 experience same packet losses due to the common bottleneck link L_c. They both join the same local recovery group. A nearby user H_1 helps them to repair the common losses.

In order to be able to relay, a helper must receive the data at a faster rate than the users of the relay subgroup. Similarly, a helper must have the requested packets before it can repair the loss packets. A helper should also be as close to the subgroup as possible in order to reduce network traffic load. In the example of Figure 1, H_1, H_2, H_3 and H_4 are all qualified to be a helper for R_1 and R_2. However, H_1 is the nearest helper. Therefore, users H_2,

H_3 and H_4 remain only as potential helpers for R_1 and R_2.

Next question is how users know whom they should cluster with. Neither common receiving rate nor loss rate is a sufficient clustering criterion. In Figure 1, users R_1 and R_6 are behind two different bottleneck links L_c and L_d, respectively. Since the bottleneck bandwidth of links L_c and L_d happens to be identical so that user R_6 sees an equal receiving rate or loss rate as R_1. Notice that R_1's helper is H_1 and R_6's helper is H_2. Therefore R_1 and R_6 should not cluster together.

Clustering around a common bottleneck link does not always result in desirable subgroups. In Figure 1, users R_4 and R_5 share the common bottleneck link L_c with users R_1, R_2, and R_3. Their nearest helper is H_2 which is also the helper for R_6 and R_7. When forming relay subgroups, they should cluster with R_6 and R_7, with neither of which they share a common bottleneck link. When forming loss recovery subgroups, they should cluster with neither R_6, R_7 nor R_1, R_2. Because they may not share the exact same loss packets with R_6 or R_7, neither do they share a common helper with R_1 or R_2.

We propose to cluster users around the common helper. Users with the *same receiving rate* and the *common helper* should cluster into the same relay subgroups, such as the two relay subgroups of $\{H_1\colon R_1, R_2, R_3\}$ and $\{H_2\colon R_4, R_5, R_6, R_7\}$. For loss recovery subgroups, we use loss fingerprint (as definded in [8]) instead of the receiving rate. Users with the *same loss fingerprint* and the *common helper* form the same loss recovery subgroups, such as the three loss recovery subgroups of $\{H_1\colon R_1, R_2, R_3\}$, $\{H_2\colon R_4, R_5\}$ and $\{H_2\colon R_6, R_7\}$. For the remainder of this report, we focus on relay subgroups. The discussion applies to loss recovery subgroups if the criterion of the *receiving rate* is replaced by the *loss fingerprint*.

Clustering around a common helper requires that each user knows each other's receiving rate as well as each other's nearest helper before the users start to form subgroups. This implies that each user must find their nearest helper first. We propose an advertisement scheme for users to find their nearest helper. Each user includes its receiving rate in an advertisement message (AM) and they multicast the AM to all of the other users. A user is a potential helper to others if its receiving rate is higher than that of others[1]. Based upon the advertisement messages from all of the others, a user determines which are its potential helpers and specifically which potential helper is closer to itself than others.

The advertisement message is to be sent periodically to protect against packet loss. The time period between two consecutive advertisement messages is called as the *AM cycle*. The AM cycle is set in such a way that the overall AM traffic load will not exceed a certain limit, e.g. 5% of the data traffic. The AM traffic load is determined by the total number of advertisement messages, which is linearly proportional to the number of users (N) of the multicast session. As N increases, the AM cycle has to increase in order to maintain the same level of AM traffic load. On the other hand, the AM cycle also determines how fast a user can find its nearest helper. During each AM cycle, every user sends its AM randomly to avoid message synchronization. Thus, a user has to wait for at least one AM cycle in order to receive advertisement messages from all the other users before it can determine which one is the nearest helper to itself. If the AM cycle is long (when N is large), the clustering process has to be delayed until the helper selection is finished.

In this report, we introduce a novel scope-controlled advertisement message to reduce the AM overhead without increasing the AM cycle. Potential helpers send advertisement messages with a scope limit which we call *useful scope*. The useful scope assures that the AM from any potential helper does not go beyond its useful range. The useful scope certainly has an upper bound which becomes smaller as the nearest helpers are identified. Subgroup members send subgroup member reports to separate subgroup addresses instead of sending advertisement messages to all of the users. Exponential random timer based suppression is used to reduce duplicate subgroup member reports. The overall AM traffic load is greatly reduced without increasing the AM cycle. Therefore, the scope-controlled advertisement message improves the scalability of the clustering algorithm.

For the remainder of this report, section II describes the scope-controlled advertising algorithm for the helper selection. Section III describes the scope-control protocols and the suppression protocols implemented in NS simulator [11]. Section IV presents the performance studies from our simulation experiments. Section VI concludes with the impact of our helper-based clustering algorithm and the potential applicability to other distributed applications. Section VI presents the rationale for using an advertising algorithm rather than a searching algorithm to find the helper.

II. A Scope-Controlled Advertising Algorithm

If the advertisement message is sent globally by all N users in a multicast session, each user will receive $(N-1)$ advertisement messages during each AM cycle. How-

[1] For local recovery, a user becomes a potential helper for another if it has the packets indicated by the other's loss fingerprint or loss bitmap.

ever, not all of these AMs are equally important for helper selection. AMs provide no useful information if they come from a non-potential helper so that any non-potential helper should not send any AMs. In Figure 1, users R_1, R_2, R_3, R_4, R_5, R_6, and R_7 are non-potential helpers, so they do not send any AMs. The AMs from potential helpers which are too far away are neither useful nor necessary. In Figure 1, H_1, H_2, H_3, and H_4 are potential helpers for R_1. The distances from them to R_1 are 4, 5, 5, 6 hops respectively. The H_1 is the nearest helper to R_1. After R_1 discovers that the nearest helper is H_1, the other AMs from H_2, H_3, and H_4 becomes useless to R_1.

We propose a *useful scope*, S_{useful}. All potential helpers send the AM with a useful scope. The useful scope is a sufficient scope for a potential helper's advertisement message. Because a potential helper may become a real helper for users within the useful scope, but it can never become a real helper for users beyond the useful scope. In Figure 1, user R_1's nearest helper is H_1 and the distance from H_1 to R_1 is 4 hops. The useful scope for potential helpers H_2, H_3, and H_4 can be set to a value not to exceed 4 hops. Although the AMs from H_2, H_3 and H_4 are not useful to user R_1 that is beyond 4 hops, they may be useful to users with 4 hops. For example, AMs from user H_2 is useful to users R_6 and R_7 and becomes their nearest helper. The useful scope assures that a potential helper's AM does not go beyond the range where it is useful.

The useful scope do not preclude any user from finding the correct helper. Suppose the AM from H_1 was somehow lost. User R_1 may have thought (incorrectly) that H_2 or H_3 was the nearest helper because it did not receive the lost AM from H_1. In this case, the distance from H_2 or H_3 to R_1 is 5 hops, so the useful scope is set not to exceed 5 hops. Eventually, the AM from H_1 with that useful scope reaches R_1. By then, user R_1 finds the correct helper H_1.

The useful scope is determined by the distance from the helper to the farthest user of the subgroup. That distance is called as the *subgroup tree depth*. With multiple subgroups present, the useful scope should be set to a value that does not to exceed the largest subgroup tree depth. In Figure 1, the subgroup tree depths of two relay subgroups are 4 and 3 hops, respectively. The useful scope is set not to exceed 4 hops.

Instead of sending periodic advertisement messages, each subgroup user sends subgroup member report (SR) for each AM received from its helper. The SR contains the distance from the helper to itself. That distance can be determined by taking the TTL field of the advertisement message from the helper. The SR is sent only to the subgroup, so that it does not reach any users from other subgroups or any potential helpers. To avoid multiple SRs within the same subgroup, exponential random timer suppression is used to allow SRs from farther users to suppress SRs from closer users.

The helper can determine the subgroup tree depth by way of the subgroup member reports. The helper is also responsible for reporting its subgroup tree depth to all other potential helpers so that they can set the useful scope for their advertisement messages. In section -B, we show that it is necessary for the helper of each subgroup to send its AM globally without scope limit in order to prevent subgroup distribution trees from overlapping with each other.

III. Helper-Based Clustering Protocol

A. Scope-Controlled Helper Advertisement Protocol

All users should join a common multicast address to listen to advertisement messages. That common multicast address is called as the AM channel. The AM channel can be the same address as the data delivery address or a separate address. Each subgroup has a separate multicast address for its member only. Subgroup users send subgroup member report to its subgroup address. The helper of each subgroup subscribes to the subgroup address to collect subgroup member reports. A potential helper does not subscribe to any subgroup addresses.

A helper sends periodic advertisement messages to the AM channel *without* the scope limit. The helper's AM includes the helper's own receiving rate, the subgroup tree depth, and the receiving rate of its subgroup. A potential helper sends periodic advertisement messages to the AM channel *with* a scope limit. The scope is set to a value not to exceed the tree depth of the largest subgroup. The potential helper's AM includes the potential helper's own receiving rate. A subgroup member does not send any advertisement messages. Instead it sends a subgroup member report to the subgroup address for each AM received from its helper. The SR includes the subgroup member's receiving rate and the distance from the helper to itself (the subgroup member). Each SR is delayed by a random timer (delay timer) according to an exponential function. As a result, a farther member's SR suppresses a nearer member's SR.

The example in Figure 1 illustrates the scope-controlled helper advertisement protocol. For simplicity, the sender S uses a layered multicast transmission, e.g. RLM [9], [15]. The data is transmitted in 2 layers. Users with higher bandwidth connections subscribes to both data layers. Users with lower bandwidth connections subscribes to only one data layer. The subscription level is used as the measurement of the receiving rate.

The helper selection process is bootstrapped by the

sender S. Initially, all users are in one subgroup (bootstrapping subgroup) with the initial helper (bootstrapping helper) as the sender. The initial useful scope is zero. During the first AM cycle (bootstrapping cycle), there is only one helper, i.e. the bootstrapping helper or the sender. S sends a global AM to the AM channel to announce the bootstrapping helper's receiving rate, the bootstrapping subgroup tree depth, and the bootstrapping subgroup receiving rate. The sender's receiving rate is the source transmission rate, or 2 layers in this example. Initially, both the bootstrapping subgroup tree depth and the bootstrapping subgroup receiving rate are unknown. So they are set to zero. Initially, all of the users are bootstrapping subgroup members, they send no AM but SR to the bootstrapping subgroup to report the distance from S to themselves and their own receiving status. With the exponenial random delay timer, most of the duplicate SRs would be suppressed (details are discussed in section III-B). During the bootstrapping cycle, the sender collects subgroup member reports to determine the boostrapping subgroup tree depth and the subgroup receiving rates. In this example, user R_6 and R_7 are the farthest members of the bootstrapping subgroup. The bootstrapping tree depth is 7 hops. The receiving rate of the bootstrapping subgroup is 1 layer.

In the next AM, S includes the subgroup tree depth of 7 hops and the subgroup receiving rate of 1 layer. Upon receiving the AM from S in the second AM cycle, users whose receiving rates are higher than the bootstrapping subgroup receiving rate become potential helpers and leave the bootstrapping subgroup. Users H_1, H_2, H_3 and H_4 leave the bootstrapping subgroup. They begin to send advertisement messages with a useful scope of 7 hops periodically.

Users begin to look for a nearby helper if their receiving rates are lower than the source transmission rate (announced by the sender as the bootstrapping helper's receiving rate). From the second AM cycle on, they are able to receive advertisement messages from potential helpers. When they find a potential helper closer than the bootstrapping helper, they leave the bootstrapping subgroup to join the subgroup of the newly chosen helper and send SRs to the new subgroup address. Users R_1, R_2, and R_3 find out that H_1 is closer than S, so they leave the bootstrapping subgroup to join the subgroup of H_1. Similarly, R_4, R_5, R_6, and R_7 leave the bootstrapping subgroup and join the subgroup of H_2.

After receiving an SR from R_1 or R_2, H_1 becomes a real helper. The SR also tells H_1 that the subgroup tree depth is 4 hops and the subgroup receiving rate is 1 layer. From next AM cycle on, H_1 begins to send global AMs to the AM channel to announce its (own) receiving rate of 2 layers, the subgroup tree depth of 4 hops, and the subgroup receiving rate of 1 layer. Similarly, H_2's advertisement message announces its (own) receiving rate of 2 layers, the subgroup tree depth of 3 hops and the subgroup receiving rate of 1 layer. Other potential helpers H_3 and H_4 update the useful scope to 4 hops.

The state diagram for each user is shown in Figure 2. All users start with the subgroup member state. Some

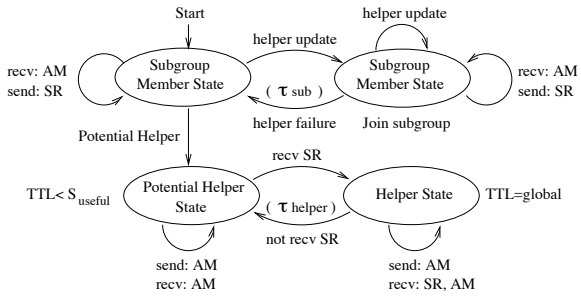

Fig. 2. The state diagram of the scope-controlled helper advertisement protocol.

users change to the potential helper state when their receiving rate is higher than that of at least one other subgroup. Some users further change to the helper state whenever they receive any subgroup member reports from their subgroup address. All users whose receiving rates are lower than the source transmission rate stay in the subgroup member state. They may switch from one subgroup to another whenever they find a new nearest helper. The switch happens at most as often as once every AM cycle.

A subgroup becomes empty when all members have left. If receiving no subgroup member report for a period of time τ_{sub} (subgroup timer), the helper goes back to the potential helper state. The bootstrapping helper (i.e. the sender) goes back to potential helper state after all users have left.

When subgroup members receive no AM from their helper for a time period of τ_{helper} (helper timer), their respective helper must have failed. Upon a helper failure, all subgroup members go back to the bootstrapping subgroup as if they had just started to join the session. When a new user joins the multicast session late, it always starts with the bootstrapping subgroup and swithes to the appropriate subgroup whenever the nearest helper is found.

The periodic advertisement messages and subgroup member reports are not sensitive to packet loss. In Figure 1, if the AM from H_1 were lost during the second AM cycle, users R_1, R_2, and R_3 would choose H_2. User H_3 is at the same distance away as H_2, but the one with smaller IP address is selected. Sometime later, when H_1's AM finally arrives, users R_1, R_2 and R_3 would switch to H_1's

subgroup.

A temporary network partition does not affect the accuracy of the helper selection. Suppose that a temporary partition separates the helper from its subgroup members. On one hand, the helper appears to fail so that subgroup members leave the subgroup address and resume the helper selection (for another helper) by joining the bootstrapping subgroup. On the other hand, all subgroup members appear to have left the subgroup so that the helper goes back to the potential helper state. When the network has recovered from the partition, subgroup members begin to receive AMs from the former helper and return to the former subgroup.

B. Suppression

Multiple subgroup member reports may appear in a subgroup of mulitple members. Each member must delay its SR by an exponentially distributed random timer [10] chosen from an interval of T_{SR}. The subgroup member report include the distance from the helper to that subgroup member. The SR from a far away member should not be suppressed by the SR from a near member. In Figure 1, the SR from R_3 can not suppress the SR from R_1 or R_2 even if R_3's delay timer goes off first. The SR from R_1 can suppress the SR from R_2 or vice versa. The suppression mechanism will assure that at least one farthest member sends out a subgroup member report per AM cycle.

The subgroup member report from near members can be further suppressed by the subgroup tree depth. The subgroup tree depth is announced by the helper of the subgroup (in the helper's AM). A subgroup member is a non-farthest member if the distance from the helper to itself is less than the subgroup tree depth. A non-farthest member does not send a subgroup member report. As a result, only the farthest members compete for sending SRs. In Figure 1, user R_3 is a non-farthest member that does not send SR. Only users R_1 and R_2 schedule a subgroup member report after each AM from H_1. This additional suppression does not affect the helper in collecting the subgroup member reports. Suppose the helper H_1 does not receive a SR within one AM cycle. Either the SR is lost or the farthest subgroup members R_1 and R_2 have left the subgroup. In the next AM cycle, H_1 announces a subgroup tree depth of zero. User R_3 is no longer a non-farthest member so that it schedules a SR.

When a potential helper happens to be on the same LAN as a real helper and its IP address is not smaller than the real helper, it will never become a real helper to any user due to the small IP address rule. That potential helper must suppress its advertisement messages by setting the useful scope to zero.

IV. Performance Studies

We have implemented the self-organized subgrouping protocols in the NS-2 [11] simulator. We define a helper convergence index H to show that the accuracy of helper selection is not affected by packet loss, helper failure, and network partition. Next, we evaluate the overhead of scope-controlled AM and SR messages in order to show that the scope control can reduce the advertisement overhead without increasing the AM cycle. We have implemented a layered multicast transmission similar to [15] in the NS simulator. In all our experiments, the sender transmits the data in 2 layers. We have configured different bottleneck links in various topologies. All users behind bottleneck links subscribe to only 1 layer, while others not limited by the bottleneck subscribe to 2 layers. Users with subscription level of 2 layers are potential helpers to those with 1 layer. The AM cycle (τ_{AM}) is set to a value much greater than the largest round trip time between the sender and any users. The SR delay timer interval (T_{SR}) is set to a value equal to half of the AM cycle (τ_{AM}) so that the SR from the farthest member can arrive at the helper before the next AM. The data is transmitted for a period of T_{TX} = 30 τ_{AM}. All measurements are taken during T_{TX} and the average results are shown. The subgroup timer (τ_{sub}) and helper timer (τ_{helper}) (see Figure 2) are set to 2 τ_{AM}.

A. Helper Convergence Index

With the absence of packet loss, each subgroup member is able to find the nearest helper within one AM cycle after the bootstrapping cycle (see section III). However, packet loss, node failure, and network partition may affect the helper selection process. We define a *helper convergence index*, H, to be the number of AM cycles (subsequent to the bootstrapping cycle) for the helper selection process to converge to the nearest helper.

In the simulation setup, packets are subject to loss due to the buffer overflow at bottleneck links. For example, an experiment is configured with 4 TCP flows and 4 multicast flows sharing one bottleneck of bandwidth 15 Mbps on the foward path and another 4 low-bandwidth TCP flows on the reverse path. The outgoing buffer size at the bottleneck point is 40 packets. The measurement reveals that about 8.7 packets are dropped for every 1000 packets transmitted on the forward path. In addition to packet loss, we also configure node failures and temporary partitions to study the helper convergence index H.

Figure 3 shows a few sample topologies used in the experiments. We have run experiments with multicast sessions of 5-, 10-, 12-, 16-, 20-, 50- and 112-users. H is measured from the end of the bootstrapping cycle to the

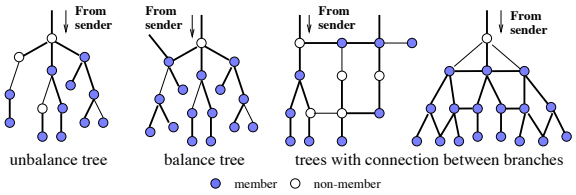

Fig. 3. Sample topologies.

time when the nearest helper is selected, in the units of the AM cycle. In Figure 4, we plot the helper convergence index H of all the users in each experiment together with the

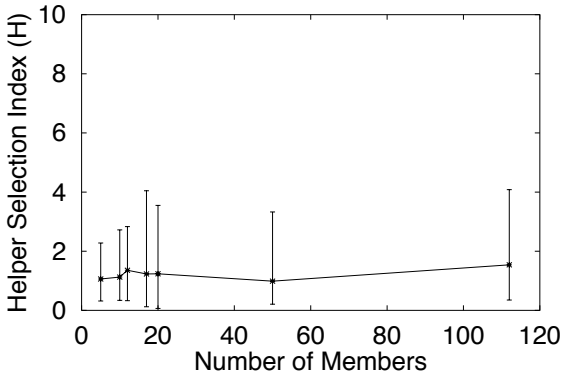

Fig. 4. Helper selection index (H).

mean value of H. On average, the helper selection process converges within one AM cycle. However, in some adverse cases, it may take over 4 AM cycles for a user to find its nearest helper. The worst scienario varies with many factors, such as packet losses, helper failure, and network partition, etc.

B. Scope-Controlled Advertising Protocol Overhead

To evaluate the overhead of the scoped-controlled advertising protocol, we take into consideration (1) the control traffic injected into the network, and (2) the control traffic concentrated at each user. The control messages include the AMs from helpers and potential helpers, and the SRs from subgroup members.

We define the weighted scope, $\mathfrak{S}_{ctrl}$, to represent the control traffic injected into the network as:

$$\mathfrak{S}_{ctrl} = \frac{S_{ctrl}}{S_{global}} \times 100\% \qquad (1)$$

S_{global} is the scope of the entire multicast session, measured by the distance between two farthest apart users. S_{ctrl} represents the scope of the control messages, e.g. the AM and the SR. When advertisement messages are sent to all others, such the global advertisement messages, $S_{ctrl}$is equal to S_{global}. The weighted scope of a global advertisement scheme is 100%. With our scope-controlled advertisement scheme, only a small number of real helpers (one per subgroup) send the AM globally. The useful scope of a potential helper's AM is usually smaller than S_{global}, so that the weighted scope of potential helper's AM ($\mathfrak{S}_{AM}$) is less than 100%. SR is sent to the subgroup whose scope never exceeds S_{global}. The weighted scope of SR ($\mathfrak{S}_{SR}$) is also less than 100%.

Next, we define N_{recv} to represent the control traffic concentrated at each user. N_{recv} is the number of control messages arriving at a user during one AM cycle. When advertisement messages are sent globally, every user will receive $(N-1)$ from all others, or N_{recv}= $N-1$. With our scope-controlled advertising scheme, only the local neighbors contribute to N_{recv} because the AMs from distant potential helpers and the SRs from other subgroups do not arrive, or $N_{recv} < (N-1)$.

B.1 Weighted Scope of the Advertisement Message $\mathfrak{S}_{AM}$

In this section, we investigate how the subgroup tree depth and the global scale of a multicast session affect the reduction in the weighted scope of the advertisement messages ($\mathfrak{S}_{AM}$). Three multicast sessions are set up with a single subgroup of each as shown in Figure 5. In all three chains,

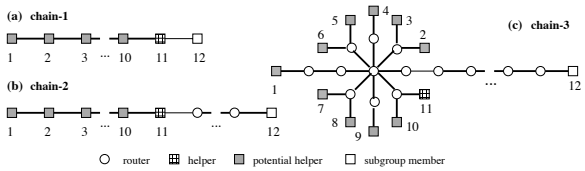

Fig. 5. The weighted scope of the advertisement messages.

user 1 is the sender and user 12 is the subgroup member. User 11 is the real helper for 12 and the rest of them are potential helpers. The average $\mathfrak{S}_{AM}$ of all users are plotted in Figure 6.

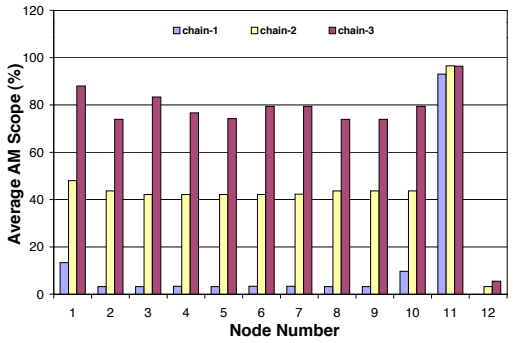

Fig. 6. The average $\mathfrak{S}_{AM}$ for (potential) helpers in Figure 5.

Figure 6 shows that our scope-controlled advertisement algorithm can reduce the weighted scope of the advertisement message ($\mathfrak{S}_{AM}$) for all potential helpers. The $\mathfrak{S}_{AM}$ of all potential helpers is below 100% for all three chains. The reduction in $\mathfrak{S}_{AM}$ depends on the useful scope and the global scale of particular topologies.

The global scale (S_{global}) is 11 hops for both chain-1 and chain-3. The subgroup tree depths of chain-1 and chain-3 are 1 hop and 10 hops, respectively. Figure 6 shows that the potential helper's $\mathfrak{S}_{AM}$ in chain-1 is much smaller than that in chain-3. On the other hand, the subgroup tree depth of chain-2 is the same as that of chain-3. The global scale in chain-2 is 20 hops, or approximately twice as much as that of chain-3. The weighted scope ($\mathfrak{S}_{AM}$) in chain-2 is lower than that in chain-3.

Figure 6 also shows that only the real helper's $\mathfrak{S}_{AM}$ (e.g. user 11) approaches 100%. The average $\mathfrak{S}_{AM}$ of the sender 1 is slightly higher than other potential helpers, because a few of the sender's AM were sent globally at the beginning when every user starts with the bootstrapping subgroup.

B.2 Upper Bound of the $\mathfrak{S}_{AM}$

In this section, we demonstrate that the useful scope of a potential helper's AM and its the weighted scope $\mathfrak{S}_{AM}$ both have an upper bound. Two multicast trees are set up as shown in Figure 7. Both tree-1 and tree-2 have two

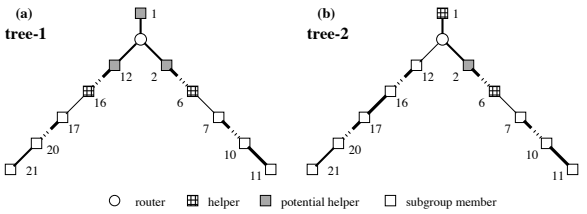

Fig. 7. Multicast sessions with two subgroups. (a) Two subgroups are of the same size. (b) The subgroup on the left branch will use the sender S as the helper.

branches and two bottleneck links in each of two branches. The global scale is twenty hops and the distribution tree depth is ten hops. User 1 is the sender. There are five potential helpers in both branches of tree-1, but no potential helper is available in the left branch of tree-2. The sender becomes the helper for the subgroup in the left branch. In other words, any subgroup is able to find a helper no farther than the sender itself. The subgroup tree depth and the useful scope both are bounded by the tree depth of the main multicast distribution tree. The average $\mathfrak{S}_{AM}$ is plotted in Figure 8. The $\mathfrak{S}_{AM}$ of potential helpers is about 25% for tree-1 and 50% for tree-2. The upper bound of the $\mathfrak{S}_{AM}$ is 50% for topologies like tree-1 or tree-2.

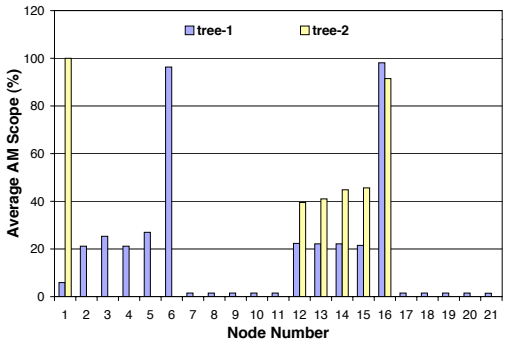

Fig. 8. The average $\mathfrak{S}_{AM}$ for (potential) helpers in Figure 7.

B.3 Suppression of Duplicate Subgroup Member Reports

In this section, we investigate the effectiveness of our suppression protocols. Two multicast sessions are set up with a single subgroup of eleven members in chain-4 and star-4 (Figure 9). User 1 is the sender and also the helper. All members are at different distances away from the helper in chain-4, while all members are at the same distance away from the helper in star-4. The subgroup scopes of both chain-4 and star-4 are the same as the global scale of the original multicast group. If a member sends out an SR for each AM received from the sender, the average weighted scope of SR ($\mathfrak{S}_{SR}$) is 100%. However, not all members send out a SR for each AM due to the suppression, thus the average weighted scope of SR of those members is less than 100%. Figure 10 shows the average weighted scope of SR ($\mathfrak{S}_{SR}$). Member 12 is the only farthest member in chain-4. It sends out one SR every AM cycle ($\mathfrak{S}_{SR}$= 100%). Other closer member's SRs are suppressed (see section III-B). In star-4, all members are furthest member, so they all competed for sending SRs. Figure 10

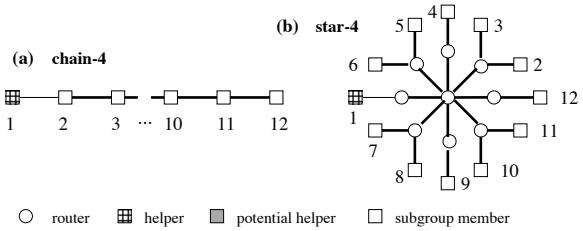

Fig. 9. Topologies with single subgroup.

shows that all farthest members have roughly equal chance to send out an SR in star-4. Duplicate subgroup member reports may still occur.

Another simple star topology is set up to study the SR suppression for multiple farthest members. All members connect to a center router. One of them is the data source

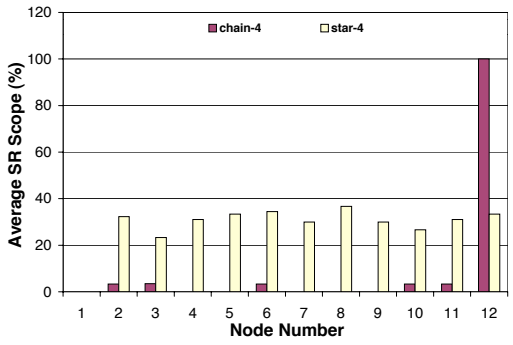

Fig. 10. The performance of suppression protocol.

and also the helper. All members are the same distance away from the helper. Simulation experiments have been done for different number of users, both with homogeneous delay and heterogeneous link delay[2]. Figure 11 shows the average number of duplicated SRs per AM cycle. The suppression can significantly reduce the number

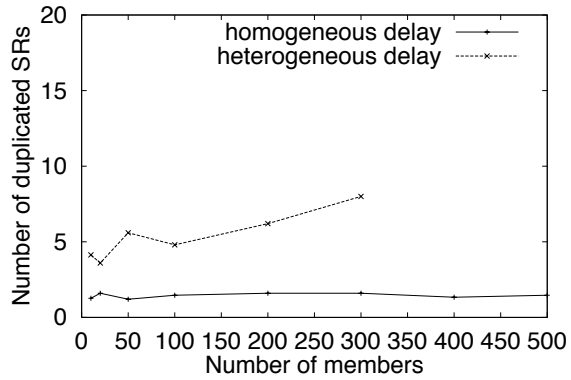

Fig. 11. The performance of suppression for subgroups with a large number of members.

of SRs both with homogeneous delay and heterogeneous delay for subgroups with a large number of members.

B.4 N_{recv}

In this section, we investigate how the useful scope affects N_{recv} and show that our scope-controlled algorithm can greatly reduce N_{recv}. The topologies of chain-2 and chain-3 in Figure 5 are used for study. Both chain-2 and chain-3 have ten potential helpers, one subgroup member, and one real helper. In experiments, the subgroup tree depth is varied in order to change the useful scope from 0 hops to 13 hops. For each useful scope, we measure the average N_{recv} of each user, and plot the mean N_{recv} of all users in Figure 12.

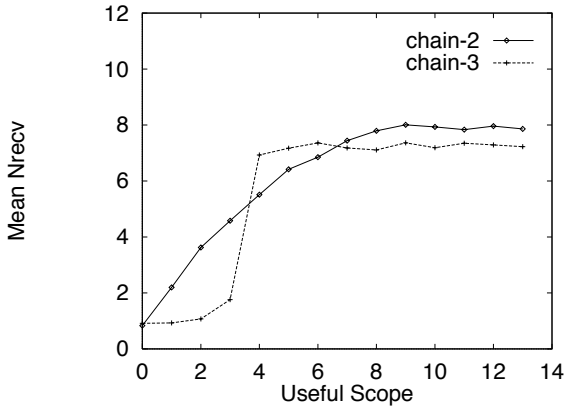

Fig. 12. The mean $N_{recv}$of all users for chain-2 and chain-3 in Figure 5.

In general, N_{recv} of a user depends on how close it is to other users. In other words, the mean N_{recv} depends largely on the specific topology. The mean N_{recv} of chain-2 gradually increases with the useful scope, while the mean N_{recv} of chain-3 dramatically rises at the useful scope of 4 hops. Most of ten potential helpers in chain-3 are 4 hops away from each other. When the useful scope is below 4 hops, their AMs can not reach each other. When the useful scope reaches 4 hops or more, the AMs can reach all others.

Figure 12 shows that our scope-controlled advertisement algorithm can effectively reduce N_{recv}. N_{recv} is 11 for a global advertisement scheme. The maximum mean N_{recv} of both topologies are below 8. N_{recv} is small if the useful scope or the subgroup tree depth is small.

B.5 A Multicast Session with a Large Number of Users

In this section, we use a more realistic topology to study the weighed scope of AM/SR and N_{recv}. We set up a topology as shown in Figure 13. Several different autonomous systems (AS) attach to a 9-node backbone. Each AS consists of various different networks, such as balanced tree, unbalanced tree, trees with long haul leaves, and trees with connection between branches (mesh structure), etc. A few sample topologies of the AS are shown in Figure 3. Asymmetric links, such as satellite links, are also configured in this topology. The source is in one of the eight ASs. The number of nodes, multicast users and the tree depth are shown in Figure 13 for each AS. The global scale of this topology is 16 hops and the local scope of each AS is about 4 hops.

[2]For homogeneous delay, the delays from the sender to all subgroup members are $20ms$. For heterogeneous delay, members are divided into 10 groups, the delay from the sender to members of each group starts from $20ms$, $40ms$, ... , $200ms$.

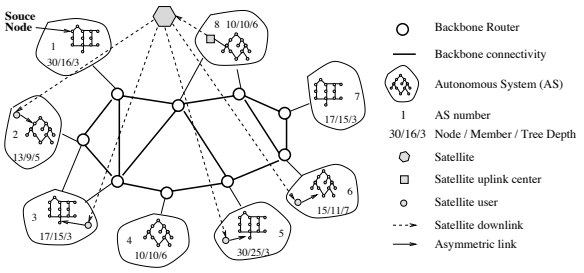

Fig. 13. A large network with asymmetric satellite links.

To comprehend how close the users are to each other in such a topology, one more user is attached to a leaf node of the multicast tree. The attached link is configured to be the only bottleneck link in order to set up a one-member subgroup. All users are potential helpers that subgroup. By varing the subgroup tree depth, different useful scope can be obtained and N_{recv} of the users can be measured. The mean N_{recv} of the users (in Table I) shows the average number of neighbor users with the range of the useful scope. For example, the mean N_{recv} is 12.2 at the useful scope of 4 hops. On average, a user receives over 12 AMs when the useful scope is 4 hops. 1 AM is from the subgroup's helper. The remaining AMs are from 11 neighbors within 4 hops. On average, 2 neighbors are within 1 hop, 11 neighbors are within 4 hops, 35 neighbors are within 6 hops, and 71 neighbors are within 8 hops, etc.

Useful scope	0	1	4	6	8
mean N_{recv}	1.08	3.29	12.2	36.2	72.0
No.(local neighbors)	0	2	11	35	71

TABLE I
MEAN N_{recv} OF ALL 112 USERS UNDER DIFFERENT USEFUL SCOPE.

To study the overall reduction in the weighted scope of AM and SR, a multicast delivery session is set up with 112 users. About 18 bottleneck links are in this particular session and 20 subgroups are to formed. The total number of potential helpers is 61, including the sender. The useful scope is 4 hops. The weighted scope of AM and SR of each users for this session with 20 subgroups is shown in Figure 14. Exactly 20 user's weighted scope ($\mathfrak{S}_{AM}$) approach 100% as shown in Figure 14. They are real helpers of 20 subgroups. The potential helper's weighted scope is approximately 25% ($\frac{S_{useful}}{S_{global}} = \frac{4}{16}$). The weighted scope of the subgroup member reports all below 100%. The near member's SR is mostly suppressed. Furthermore, the SR from one subgroup does not reach members of other subgroups or any potential helpers. Our scope-controlled

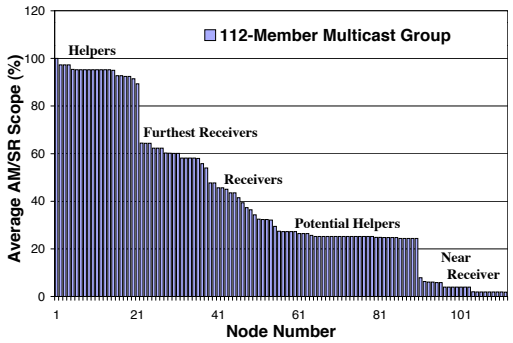

Fig. 14. The average weighted scope of AM or SR of each user in Figure 13 with 20 subgroups.

advertisement algorithm greatly reduces the control traffic injected into the network by the potential helpers and subgroup members.

N_{recv} increases with the number of potential helpers within the multicast sessions. With the scope control, each user mainly receive advertisement messages from its local neighbors within the region of the userful scope and a small number of remote helpers beyond the region of the useful scope. N_{recv} does not increase with the number of potential helpers within a multicast session. To show that, six multicast sessions are set up in the topology shown in Figure 13. The number of users in each session ranges from 50 to 187. The number of potential helpers in each session ranges from 24 to 136. The rest of the users are subgroup members and form 10 subgroups accordingly. The useful scope is 4 hops for all six sessions. The N_{recv} is measured for each user and the mean N_{recv} of all users is plotted in Figure 15 for six sessions. Figure 15 shows that

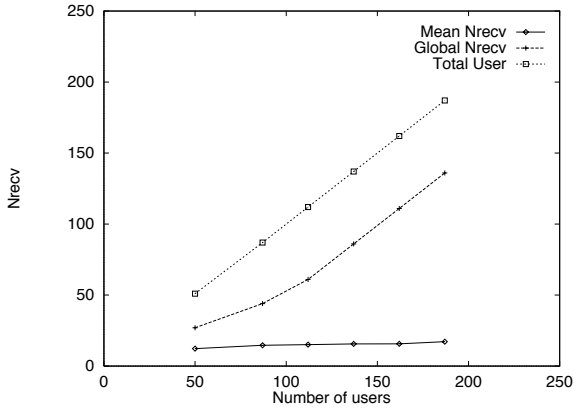

Fig. 15. The mean N_{recv} does not increase with with the number of potential helpers in the multicast sessions.

the mean N_{recv} does not increase with the number of the pontential helpers. N_{recv} for global advertisement scheme

is also plotted in Figure 15 and it increases with the number of potential helpers.

The number of remote helpers does increase when the number of subgroups increases so that N_{recv} increases with the number of subgroups. To show that, four different multicast sessions are set up with 112 users. Each session is configured with different number of subgroups, from 5, 10, 15 to 20. The total number of potential helpers is fixed as 61 and the useful scope is 4 hops in all four sessions. The mean N_{recv} of all users are plotted in Figure 16 for all four sessions. Table I says that each user has about 11

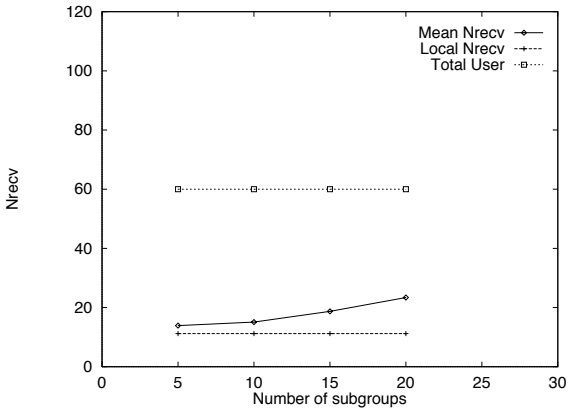

Fig. 16. The mean N_{recv} does increase with with the number of subgroups in the multicast sessions.

neighbors within the region of 4 hops for this particular topology. In addition to messages received from 11 local neighbors with 4 hops, each user received global advertisement messages from remote helpers beyond 4 hops. As the number of subgroup increases, the more remote helpers would emerge. Since some of the helpers may locate inside the local region of the useful scope, N_{recv} inscreases at a relatively lower rate than the number of the subgroup does.

C. Discussion

In principle, our scope-controlled advertisement scheme requires that all users be willing to help others. In fact, this requirement can be relaxed. If a potential helper does not desire to serve as a helper, it simply stops sending advertisement messages. A subgroup member can always find an appropriate helper from those potential helpers who are willing to help.

The current MBONE has implemented an administrative scope mechanism at border routers to prevent local multicast traffic from leaking out. The multicast packets for the local region are given the scope of 16 hops. Any multicast packets with an TTL less than 16 are dropped by the border routers. This administrative scope control does not interfere with our scope-controlled advertisement scheme. If the useful scope is less than 16 hops, the administrative scope does prevent any potential helper's AM from going across administrative boundary. This is equivalent to the situation where a potential helper does not desire to help any subgroups outside its local region.

V. Related Work

Many research works in reliable multicast areas have applied clustering for different purposes. For example, users cluster together for loss recovery, such as in local recovery for SRM [8], RMTP [7], LGC [2], TMTP [17], LBRM [3], SHARQFEC [4]. A special user (helper) is chosen to handle loss repairs for each subgroups. Another example for clustering is in SOT [5]. Slow users group together to receive the transcoding stream at a lower rate.

The local recovery groups [8] and the transcoder groups [5] suggest to cluster users around bottleneck links. The local recovery groups are built based on the common loss fingerprint and the transcoder groups are built based on the common loss bitmap. Users behind the same bottleneck links experience the same loss fingerprint or loss bitmap. However, as been discussed in section I, clustering around a common bottleneck link may not always result in the desirable subgroups.

Both the local recovery groups and the transcoder groups employ the uniform random timer based suppression mechanism to reduce the number of duplicate helper announcement messages [8] or transcoder search messages [5]. The uniform random timer is very sensitive to the delay measurement. On the other hand, we use an exponential random timer based suppression mechanism [10] in our helper based clustering protocol. The exponential random timer based suppression mechanism is not sensitive to the delay measurement and is more effective in reducing the number of duplicate messages.

The uniform random timer based suppression mechanism requires the delay measurement between two users. The delay measurement is obtained by way of the global session messages of the SRM in [8] or some external synchronized clocks as suggested in [5]. The global session message mechanism does not scale well to multicast sessions with a large number of users.

A hierarchical scheme (scalable session message [13]) is proposed for SRM to reduce the overhead of global session messages. Statistically, a small number of users send out session message globally to every user, while most of the other users send session messages only to local neighbors. This hierarchical scheme is similar to our scope-controlled helper advertisement scheme. All potential helpers send advertisement message with a useful scope. Only a small number of real helpers send advertise-

ment messages globally.

Clustering is only the first step. Next, subgroups must find a nearest helper for either loss recovery or transcoding stream. SHARQFEC [4] requires that the helper response to the zone (subgroup) users quickly. The users must find the helper with the shortest delay. SOT [5], TMTP [17] and LBRM [3] all employ a multicast search mechanism to find the nearest helper. The helper who responses to the search message first becomes the closest helper. Both the delay measurement and the search mechanism assume that the path from the helper to the subgroup user is symmetric. In section -A, we also show that the search mechanism is not suitable for networks which contains asymmetric paths.

RMTP [7] and LGC [2] both employ the advertisement mechanism. However, the designated receivers (helper) are manually configured in RTMP, which makes it difficult to adapt to network changes. LGC proposes an expanded advertisement approach. Advertisement messages are sent with a scope expanding from a small value. The scope can still expand to global scale. On the other hand, the useful scope in our scope control algorithm certainly has an upper bound and will shrink (not expand) as soon as the nearest helper is found.

A totally different approach has been proposed to allow clustering based on end user preference [16]. A matchmaker can collect the preference from all users before it can make optimal decision for all. However, it relies on exchanging preference messages among all users. The global message exchange faces the same scalability problem as the global session message.

All above approaches are end-to-end solutions that require no router modification. Another set of approaches, such as subcast [12], AIM [6] and PGM [14], etc., rely on additional support from special routers. For example, routers above the bottleneck link can forward repair packets only to certain interfaces in order to reach users behind the bottleneck link. With additional support from special routers, it becomes easy to identify the bottleneck links for efficient loss recovery.

VI. Summary

We have developed a helper-based clustering algorithm. Instead of clustering around common bottleneck links, users find the nearest helper first and then they cluster around the common helper. We have employed an advertising algorithm for helpers to announce their availabilities and for users to find the nearest helper. No assumption on network path symmetry is made. We have introduced a novel scope control algorithm to apply a useful scope to the advertisement messages of potential helpers. The useful scope assures that an advertisement message does not travel beyond where it is useful. The useful scope has an upper bound and becomes smaller after the nearest helpers are found. Separate subgroup addresses and effective exponential random timer based suppression mechanisms insure a small number of subgroup member reports per subgroup. Our helper-based clustering protocol is robust and not sensitive to packet losses, node failures, and network partitions.

We have implemented the helper-based clustering protocol in the NS simulator. Our simulation experiments show that the useful scope can effectively reduce the traffic load injected into the network by the advertisement messages. Large reduction in the advertisement traffic load can be obtained in the multicast sessions with small useful scopes. The useful scope of a multicast session is bounded by the distance from the sender to the farther member. The advertisement traffic load can be reduced to only a small fraction of what it would have been without the scope control.

The advertisement messages from distant potential helpers do not reach any users beyond the range of the useful scope. The subgroup member report from one subgroup does not reach any members of other subgroups or any potential helpers. The control traffic concentrated at each user mainly come from local neighbors within the region of the useful scope and remote helpers beyond the region of the useful scope. One study [1] has discovered that only a small number of major bottleneck links exist in most of the current MBONE multicast sessions. This finding implies that a small number of subgroups may appear in a practical multicast session. The control traffic concentrated at each user is mainly determined by the number of local neighbors.

Our scope control algorithm works with the current MBONE administrative boundary control. It does not rely on any router modifications.

Our helper-based clustering approach can also apply to other applications, such as content distribution, where heterogeneity is also a great challenge. The clustering eases the difficulty of handling heterogeneous clients from different locations. The helper server can relay the content during the distribution session as well as serve as a replicate server after the distribution is finished.

Appendix

Appendix Two mechanisms are usually used in finding the nearest helper. One is the search mechanism and the other is the advertisement mechanism. The search mechanism requires that a user send out search messages. The helper who is closer to the user than others receives the search message first and thus responds first. However, the

search mechanism assumes the path from the helper to the user is symmetric. On the other hand, the advertisement mechanism does not make such assumption. In following sections, we demonstrate that the advertisement mechanism is more appropriate for networks with asymmetric paths, such as satellite links, than the search mechanism. We also demonstrate that the advertisement algorithm can prevent subgroups from overlapping with one another.

A. Asymmetric Networks

Asymmetric paths have become popular since the satellite networks are integrated into the Internet. Satellite users can receive broadcast data through the satellite downlink from any place that is covered by the satellite (see Figure 13). Currently, a satellite user connects to the satellite uplink center through a terrestrial network. Suppose a number of satellite users join a multicast distribution session and the data comes down to them through the satellite downlink. When a bottleneck occurs between the multicast sender and the satellite uplink center, all of the satellite users seek a helper. If the search mechanism were to be used to find the nearest helper, each of these satellite users would find a helper from the terresterial networks they connect to. Each of the selected helpers would send a copy of the same data to the satellite uplink center. This is not a desirable behavior because one copy of the data is enough for the satellite uplink center to send to all satellite users. The helper advertisement mechanism can avoid multiple copies of the same data to be sent to the satellite uplink center. All of the satellite users find the same helper by way of the helper advertisement mechanism. Only one copy of the same data is sent to the satellite uplink center towards all the satellite users.

B. Preventing Subgroup Overlap

If multiple subgroups appear within one multicast session, these subgroup distribution trees should not overlap one another. Otherwise, two subgroup trees may share some common link(s). If two helpers of such subgroups happen to transmit the same packets, two copies of the same packets may go through the shared link twice. Subgroup overlap is not easy to detect. Thus it is difficult to resolve when it happens.

We now show that the global advertisement messages from the subgroup helper can prevent two subgroups from overlapping one another. Not to lose the generality, suppose two helper trees for subgroup A and B share a common link L_c as shown in in Figure 17(a).

According to the assumption, some of the users behind the shared link L_c belong to the subgroup A while others belong to the subgroup B, *concurrently*. Two representative users are selected: user i belongs to the subgroup A and user j belongs to the subgroup B. The advertisement messages from both helper A and B arrive to user i and j by way of link L_c. Since user i is in the subgroup A, the distance from B to user i, D_{Bi}, must be greater than the distance from A to user i, D_{Ai}. The difference in the distance is $\Delta_i^{BA} = D_{Bi} - D_{Ai} > 0$. Δ_i^{BA} is equal to Δ_C^{BA}, which is measured from a node C at the junction of two subgroup distibution trees. Thus, Δ_C^{BA} is *greater* than 0. However, if the same calculation is applied to user j, Δ_C^{BA} becomes *smaller* than 0. This contradicts what has been derived for user i. Therefore, the initial assumption is not correct. All of the users behind link L_c should belong to either subgroup A or subgroup B, *but not both at the same time*. Link L_c can only be in either subgroup A or subgroup B. The global advertisement messages from helpers can prevent subgroup overlap.

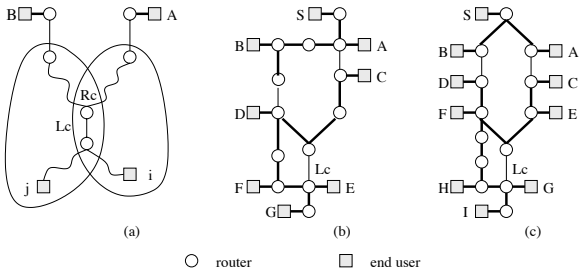

Fig. 17. Multiple subgroup trees are disjoint by way of advertisement messages. Two helper trees are rooted at B and A respectively. The main multicast trees are not shown. (a) shows the assumption that two subgroup trees share one common link L_c. (b) and (c) give real life example of multiple disjoint subgroups.

REFERENCES

[1] Mark Handley. A congestion control architecture for bulk data transfer. Viewgraph for Reliable Multicast Meeting of IRTF, September 1997. Obtained via: ftp://north.east.isi.edu/~mjh/slides/rm-bulk-data.ps.gz.

[2] Markus Hofmann. A generic concept for large-scale multicast. In *Proceedings of International Zurich Seminar on Digital Communication (IZS'96)*, Springer Verlag, February 1996.

[3] Hugh W. Holbrook, Sandeep K. Sighal, and David R. Cheriton. Log-based receiver-reliable multicast for distributed interactive simulation (dis). In *Proceedings of ACM SIGCOMM'95*, pages 328–341, Boston, MA, August 1995.

[4] Roger Kermode. Scoped hybrid automatic repeat request with forward error correction (sharqfec). In *Proceedings of ACM SIGCOMM'98*, Vancourver, Canada, September 1998.

[5] Isidor Kouvelas, Vicky Hardman, and Jon Crowcroft. Network adaptive continuous-media applications through self organized transcoding. In *Proceedings of Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 98)*, Cambridge, UK, July 8-10 1998.

[6] Brian Neil Levine and J.J. Garcia-Luna-Aceves. Improving internet multicast with routing labels. In *Proceeding of IEEE Interna-*

tional Conference on Network Protocols (ICNP), pages 241–50, Octorber 1997.

[7] John C. Lin and Sanjoy Paul. Rmtp: A reliable multicast transport protocol. In *Proceedings of IEEE Infocom'96*, pages 1414–1424, San Francisco, CA, March 1996.

[8] Ching-Gung Liu, Deborah Estrin, Scott Shenker, and Lixia Zhang. Local error recovery in srm: Comparison of two approaches. *IEEE/ACM Transactions on Networking*, 6(6):686–699, December 1998.

[9] Steven McCanne, Van Jacobson, and Martin Vetterli. Receiver-driven layered multicast. In *Proceedings of ACM SIGCOMM'96*, pages 117–130, Stanford, CA, August 1996.

[10] Joerg Nonnenmacher and Ernst Biersack. Optimal nulticast feedback. In *Proceedings of IEEE Infocom'98*, San Francisco, CA, March 1998.

[11] *UCB/LBNL/VINT Network Simulator, Version 2*. URL: http://www-mash.cs.berkeley.edu/ns.

[12] Christos Papadopoulos, Guru Parulkar, and George Varghese. An error control scheme for large-scale multicast. In *Proceedings of IEEE Infocom'98*, San Francisco, CA, March 1998.

[13] Puneet Sharma, Deborah Estrin, Sally Floyd, and Lixia Zhang. Scalable session messages in srm. Technical report, University of Southern California, February 1998.

[14] Tony Speakman, Dino Farinacci, Steven Lin, and Alex Tweedly. *Pretty Good Multicast (PGM) Transport Protocol Specification*. Cisco Systems, January 1998. Internet Draft.

[15] Lorenzo Vicisano, Luigi Rizzo, and Jon Crowcroft. Tcp-like congestion control for layered multicast data transfer. In *Proceedings of IEEE Infocom'98*, San Francisco, CA, April 1998.

[16] Tina Wong, Randy Katz, and Steven McCanne. A preference clustering protocol for large-scale multicast applications. In *Proceedings of Networked Group Communication Workshop 99*, Pisa, Italy, Nov. 1999.

[17] Rajendra Yavatkar, James Griffioen, and Madhu Sudan. A reliable dissemination protocol for interactive collaborative applications (tmtp). In *Proceedings of ACM Multimedia'95*, November 1995.