# Hierarchical Reliable Multicast: Performance Evaluation and Optimal Placement of Proxies

Sudipto Guha, Athina Markopoulou, Fouad Tobagi

*Abstract*— **This paper studies the use of multicast together with proxy nodes for reliably disseminating data from a single source to a large number of receivers. In order to achieve reliability, data must be retransmitted in case of loss either by the source or by special network nodes, called proxies. Each proxy is responsible for reliably delivering the data to a subgroup it is assigned. The multicast tree is partitioned into subgroups that form a hierarchy rooted at the source, hence the term Hierarchical Reliable Multicast. The performance of this approach strongly depends on the topology and the loss characteristics of the underlying tree and the location of proxies. In the first part of the paper, we study the processing and bandwidth performance of such a reliable multicast dissemination given the tree and the placement of proxies. In the second part of the paper, we develop dynamic programming algorithms that give a placement of a fixed number of proxies on an arbitrary tree that minimizes the bandwidth used for reliable transfer. The first algorithm provides an optimal solution to the multicast proxies location problem in polynomial time, in the number of nodes and proxies. The second is an approximation algorithm that gives a solution with cost within a chosen precision from the optimal, in an improved running time. An optimal and an approximate solution are also provided for the proxies location problem if unicast is used for transmissions. Applications of this dynamic programming approach to related problems are discussed.**

**Keywords: reliable multicast, proxies, performance analysis, dynamic programming, location problems, hierarchy, optimization, approximation scheme.**

Sudipto Guha is with the CIS Department, UPENN (email: sudipto@cis.upenn.com). Athina Markopoulou is currently with SprintLabs, Burlingame, CA (email: amarko@stanfordalumni.org, mail: 616 Harvard Ave, Apt#2, Menlo Park, CA 94025, USA). Fouad Tobagi is with the Electrical Engineering Department, Stanford University. This work has been conducted while the first two authors were PhD students at Stanford. The conference version, which appeared in NGC '00, [17], focused on the analytical metrics for performance evaluation. This journal version, currently submitted to Computer Communications, is extended by the algorithms for optimal and near-optimal placement of proxies.

## I. Introduction

Data dissemination applications such as the distribution of newspapers or movies, as well as the updates of stock quotes, software and web caches require reliable data transfer from one source to many receivers of potentially huge number and wide geographical distribution. In this paper, we study the use of multicast for serving such one-to-many applications.

IP multicast naturally fits such applications by constructing the multicast routing tree that allows the source to reach all receivers. The well-known strengths of IP multicast are that it saves network bandwidth by duplicating the packets only where it is necessary and that it allows for dynamic membership in a way transparent to the source. However, IP multicast provides only a a best effort delivery, while applications do have reliability requirements.

A large number of *Reliable Multicast (RM)* protocols have been developed in the last decade mainly for the purpose of ensuring reliability at the transport layer and also for ensuring some congestion control in the network, issues similar to those addressed by TCP. In this paper, we focus on the reliability aspect. Receivers send feedback about whether they received the data packets or not and the source retransmits the packets until all receivers get them.

Reliable Multicast protocols face a number of well-known problems, inherent to the nature of the one-to-many communication model, es-

pecially for large numbers of receivers. *Feedback implosion* occurs when large groups send feedback without duplicate suppression, thus unnecessarily wasting bandwidth, overwhelming the source with processing and increasing the latency in the delivery of data. Another inherent problem is the *exposure*, [21] (also called the *crying baby problem*, [7]): parts of the tree that experience high loss, expose the rest of the tree to unnecessary retransmissions. The *drop-to-zero* problem occurs when receivers with poor capabilities force the rest of the group to adjust at the slowest rate.

One successful approaches that deals with the above problems is the hierarchical proxies approach, which we call *Hierarchical Reliable Multicast (HRM)***.** This approach partitions the multicast delivery tree into subgroups that form a hierarchy rooted at the source. Each subgroup has a representative, called proxy, which keeps copies of data packets, collects the feedback from the receivers in the subgroup and locally retransmits the packets, if needed. Feedback implosion is limited because each proxy handles a smaller size of subgroup. Limiting the feedback and the retransmissions locally saves bandwidth and limits the exposure of the good parts of the tree. The recovery latency is reduced as repairs come from a proxy located close to the point of loss. Large numbers of receivers may join the group in this hierarchical scalable way.

The performance of such hierarchical schemes, strongly depends on the underlying tree topology and its loss characteristics, and on the selection of proxies. In this paper, we study two problems: the performance evaluation of a tree given a placement of proxies and the optimal placement of proxies.

The structure of the rest of the paper is as follows. In Section II we discuss work in this area and how our work relates to it. In Section III we present our model for Hierarchical Reliable Multicast. The performance evaluation of an entire multicast tree is reduced to evaluating the performance of every subgroup, in section IV. The performance measures considered are the work at the source (subsection IV-A) and the network bandwidth (subsection IV-B) needed for reliable transfer. Section V studies the optimal placement of a fixed number of proxies on the multicast tree in order to minimize a total bandwidth measure, for multicast and unicast transmissions in sections V-A and V-B respectively. In subsection V-A.1, we give an algorithm that provides an optimal placement for the multicast problem. In subsection V-A.2, we give an approximation algorithm for the multicast problem too. An example of the approximation algorithm using a realistic topology and MBONE measurements is given in subsection V-A.3. Also algorithms to find both the optimal and an approximation are given in subsection V-B for the unicast case. Possible applications of the Dynamic Programming approach to related problems are discussed in subsection V-C. Section VI concludes the paper.

## II. Related Work and Contributions

There are three bodies of work related to this paper: (i) implementation work that motivates the study (ii) performance evaluation work related to the first problem and (iii) optimization algorithms for location problems, related to the second problem we are studying.

As discussed in section III, there are many applications [9], [29] and transport protocols [16], [7], [6], [15], [28], [3],[23], [21] that follow the model of hierarchical reliable multicast by using local error recovery combined with hierarchy. They address performance evaluation by means of simulation and they place their proxies using heuristics. Our work is of analytical nature and can be used for the assessment of these schemes and for optimizing their hierarchical structure. On the other hand, measurements [27] and realistic network scenarios [6], are the input to our algorithms.

The first problem we study is performance evaluation in terms of work at the source and bandwidth. Previous work on performance analysis of reliable multicast has focused on the work at the source. The "$E[M]$" measure of

subsection IV-A was defined and analytically calculated in [1] and ever since it has been used in most performance analysis work on RM: [12], [20], [10], [19], [26],[6], [18]. Its calculation is computationally intense, so approximations [18] or simulations [6] are often used. We extend the understanding of this measure and we apply it on a realistic example. The second measure "$E[T]$" captures the bandwidth used to reach all receivers at least once. So far, it has been only counted by simulation [15] or approximated by an upper bound which is tight in simplified star topologies [12], [10], [19]. In section IV-B we calculate it analytically based on our previous work [17]. We give closed formulas in special cases and a recursive method for the general case.

The second problem we study is the optimal placement of a fixed number of proxies ($k$) in order to minimize the total bandwidth needed to reach all $(n)$ receivers at least once in an expected sense. Similar problems that look for optimal placement of facilities on graphs, known in general as *location problems* and in particular as the *K-median* problem, have been studied in length, [4], [24], [2], [14], [11] for unicast data delivery. To the best of our knowledge, the location problems for multicast transmission had not been solved so far. If unicast transmissions are used, the cost of a subtree depends only on the subtree itself. The multicast case is fundamentally different in that the cost of a subtree depends on the transmissions destined to nodes not only inside but also outside of the subtree itself; each multicast transmission is heard by all nodes whether they need it or not (*exposure* problem). We dealt with special cases, i.e. chains and uniform trees, in previous work [17].

In this paper, we provide two algorithms that solve the location problem for an arbitrary multicast tree. They both follow a dynamic programming approach, similarly to [24] and [14]. In section V-A.1, we present an algorithm that provides an optimal placement in $O(n^3k^2)$ time. In section V-A.2, we present an approximation algorithm that gives a solution with cost within a chosen $(1+\epsilon)$ precision from the optimal, in $O(nhk^2 \log n/\epsilon)$ time, where $h$ is the height of the tree. The approximation algorithm although suboptimal, has the following advantages. First it has better running time, especially for short trees that are the case in MBONE, [27]. Second, it is appropriate to use when the input of the problem itself (i.e. the links loss rates) are known within a certain precision. In addition, it is not as sensitive as the optimal algorithm to the precision of numerical calculations. Finally, it generalizes to interesting variants.

We also consider the optimal location problem for the unicast case, which is similar to the K-median problem with the difference that there are loss probabilities, instead of fixed costs, associated with links. The dynamic approach still applies and gives an optimal solution in time $O(nhk^2)$ and $(1+\epsilon)$ approximation in $O(nhk^2 \log n/\epsilon)$ time. Finally, our dynamic programming approach can also handle various related network design problems, section V-C.

## III. Model

### A. Problem Context

We are interested in applications that reliably disseminate data from a central location to a large number of receivers, whose locations are relatively fixed in time. Multicast is used as the delivery method for its bandwidth-efficiency and proxies are used to localize retransmissions and provide scalability. As a generic example, one can imagine the electronic distribution of the Wall Street Journal using multicast from the central office to remote servers and from each local server to all the receivers in its own local area. Another example is the use of Starburst Multicast MFTP by "Toys R Us"[9], to distribute software and pricing information from their data center in New Jersey to more than 900 US stores over a VSAT network. The same technology for reliable data delivery has also been used by Wal-Mart to transfer media files to 2000 stores worldwide, [9]. Updating web cache contents could be another application. [29] proposed an adaptive web caching

structure using multicast for data dissemination to the caches and a self-organizing hierarchy of caches. The methods we present could be used (i) to evaluate the performance of their structure and (ii) to optimally place their caches.

The mechanism used to serve the reliable dissemination application could be one of the proposed reliable multicast protocols or a proprietary scheme, as long as it uses multicast for data delivery, a hierarchy of proxy nodes and provides full reliability. Many existing layer-4 reliable multicast protocols follow this approach of hierarchical error recovery through proxies: RMTP [16], LBRM [7], LGMP [6], OTERS [15], TMTP [28], RMX [3].[1] In the cases where the application achieves reliability through some overlay network of servers, the topology shown in Figure 1 is a logical one, connecting the application nodes through an overlay network; the loss rates should then summarize the loss across multiple consecutive physical links.

### *B. Modeling Assumptions*

We use the model shown in Figure 1. Our modeling assumptions are the following.

1) The root is the source and all other nodes are receivers (a single-source multicast tree).
2) The topology and the loss probabilities of the links on the multicast tree are known.
3) The tree is partitioned into subgroups (or subtrees) that form a hierarchy rooted at the source. The root of each subtree acts as a proxy for this subgroup and as a simple member for the upstream subgroup.
4) Error recovery inside a single subgroup is performed as follows. The proxy multicasts the original data packet to the whole subgroup. All members send feedback back to the proxy, in some way ignored in this model. If all members of the subgroup have received the packet at least once, then the proxy multicasts the next data packet. If one or more members of the subgroup lost the packet, then the proxy retransmits it. There are two options for retransmissions:
   a) Multicast. The proxy multicasts the packet to the entire subgroup even if some members don't need it. This pure multicast scenario is the focus of the paper, subsection V-A, .
   b) Unicast. The proxy unicasts the packet only to receivers that reported loss. Pure unicast retransmissions and mixed scenarios are considered in subsections V-B and V-C respectively.
5) Subgroups are separated in the sense that transmissions from every proxy to the members of its subgroup are limited locally and do not reach members of any other subgroup.

Let us now justify our modeling assumptions one by one.

According to Assumption 1 there is a single source, which naturally fits the data dissemination applications that we consider. The problem with multiple sources can be considered as superposition of multiple single-source trees, as suggested even at the network layer in [8].

Assumption 2 states that the topology and the loss rates on the links of the multicast tree are known. One might wonder how realistic is such an assumption, especially that IP multicast provides transparent connectivity and allows for dynamic membership? The problem under study is a relatively static one, where receivers do not change frequently. In this case, it is indeed possible to acquire knowledge about the multicast tree and its loss characteristics through measurements. For example, a receiver can use the MTRACE tool to find out the multicast route and collect link loss statistics on the path toward the source. A protocol com-

[1] In their context, nodes performing proxy functionality are called designated receivers or DRs in RMTP and OTERS, log servers in LBRM, group controllers in LGMP, domain managers in TMTP or proxies in RMX. Proxies may be members of the group (in LGMP, RMTP, OTERS) or special servers (in LBRM, RMX); they may be co-located with the routers, assisted by them (in PGM [23], OTERS, LMS [21]) or at a higher layer.

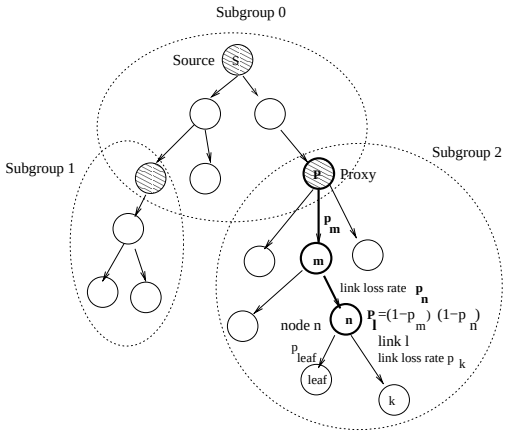



Fig. 1. Hierarchical Reliable Multicast

ponent called “Tracer”, was proposed in [13] exactly for this purpose. [15] also assumes such a backtrack capability, to track the path from the receiver to the source. Also, recent work in [22] and [25], made it possible to infer the topological structure and the loss probabilities based only on end-to-end loss prints and their correlation. Finally, actual measurements on MBONE, [27], [5] are already available.

Assumptions 3, 4 and 5 capture what hierarchical reliable multicast protocols, [16], [7], [6], [15], [28], [3], actually do: build a hierarchy of subgroups and localize the error recovery.

Assumption 4 describes the error recovery scheme inside each subgroup. The proxy is the only node allowed to send retransmissions. In practice, receivers may also send retransmissions in their neighborhood. However, it is desirable to make the appropriate receiver, i.e. the one closer to the point of loss, send the retransmission. Both LMS [21] and OTERS [15] try to find a “turning point”, i.e. the root of the subtree where the loss occurred. If there are many losses spread across different branches then a single multicast retransmission (4a) will be a repair for all. If only a few and uncorrelated receivers experience significant losses, then it makes sense for the proxy to send unicast retransmissions (4b) only to them. Some protocols, e.g. [16], have the ability to switch between (4a) and (4b) based on the number of retransmission requests . It is important to mention that dedicating the proxy as the responsible node for retransmissions, also allows for easy congestion control. In other schemes, when the original packets come from the source without coordination with local repairs, possible congestion problems may arise. However, we do not model the congestion control module of reliable multicast protocols. Finally, modeling assumption 4, matches also the problem of reliable delivery via dedicated servers, where nodes other than proxies/dedicated servers are kept simple and do not provide retransmission capability.

Assumption 5 states that subgroups are separated from each other and it captures the fact that all HRM schemes try to localize the recovery traffic among members of the same subgroup and not reach any other node. To achieve this separation, the proxy needs a way to address exclusively the members of its subgroup. In practice HRM protocols may achieve this goal exactly (e.g. by using a separate multicast address [6]) or approximately (by using the group’s multicast address and TTL scoping [28], or by using subcasting [15] and TTL scoping). However, both methods have well-known weakness. Using separate multicast addresses for subgroups does not guarantee that the retransmissions are forwarded through the

same multicast tree as the original transmission. TTL scoping lacks direction; the applicability of administrative scoping depends on the location of the receivers.

Our performance measures are the total number of transmissions needed to deliver a packet correctly to all members. They depend on the underlying tree characteristics, the topology and the link loss rates and they are the same whatever feedback mechanism one uses to report the loss. Feedback traffic is considered negligible compared to the forward traffic. Many mechanisms can be used to reduce feedback such as the use of negative acknowledgments (NAKs) with duplicate avoidance (i.e. receivers listen to the multicast group address for duplicate NAKs before sending their own requests) or summary feedback instead of per packet feedback . The appropriateness of such feedback suppression mechanisms depends on the application.

## IV. Performance Analysis of a single subgroup

In this section we consider a single subgroup, e.g. subgroup 2 in Figure 1, in which all transmissions come from the proxy. This could be the entire multicast tree if the source itself were the only proxy.

We are interested in two performance measures: the number of transmissions by the proxy $M$ and the number of link-transmissions $T$. The first measure captures the amount of "work" required by the source; it is also related to the delay and bandwidth overhead introduced by retransmissions. The second measure captures the bandwidth spent across the entire multicast tree to achieve reliability. Indeed, retransmissions take place when a loss occurs; due to their multicast nature, they use bandwidth on all links of a subgroup, not only the ones that incurred the loss. The random variables $M$ and $T$ depend on the link loss rates and the topology of the tree. $M$ and $T$ although related, they are different. For example, let us assume that the proxy in Figure 1 sends the initial multicast packet (M=1) which is lost on link (m,n); so the link transmissions are T=5. If the next multicast transmission is successful, then M=2 and T=5+7=12 total link transmissions have taken place.

In this section we address the following problem. Given the topology of the subtree and the loss rates of the links, we calculate analytically the mean values of the two measures of interest (i) the expected number of transmissions from the source, $E[M]$ and (ii) the expected number of link transmissions $E[T]$, needed for all members of the subgroup to receive a certain packet at least once. The notation used in this section is summarized in Table I.

### A. Transmissions from the proxy

A commonly used measure in the performance analysis of reliable multicast is $E[M]$ , the expected number of transmissions by the proxy for a packet to reach correctly the entire subgroup. It captures the amount of work at the source and it affects the network bandwidth used.

Let us consider the subgroup 2 of Figure 1 with proxy $P$. A packet is initially multicast. If there were no loss, there would be exactly one multicast transmission. If there is loss, the proxy $P$ retransmits the packet until all the nodes in the subtree receive it at least once. [1] recursively calculated the cumulative probability $F_n(i)$ =*Pr{after i transmissions, all nodes from n and below, received the packet at least once}*, starting from the nodes and proceeding towards $P$. Thus, [1] calculated $E[M]$ at the proxy.

One can think of these calculations as a way to find the "*equivalent link*" of a subtree in a bottom-up way. An analogy from the electric circuits, could be the equivalent impedence of an entire circuit using the Norton and Thevenin rules. An entire multicast tree rooted at the source $S$ can be thought as equivalent to a link with appropriate loss rate $p_{equiv}$ such that the expected number of transmissions $E[M]$ as seen by the source is the same. Similarly to

TABLE I
NOTATION USED IN SECTION IV

| Symbol | Meaning |
|---|---|
| $M$ | number of transmissions by the source, until all nodes receive a packet at least once (deterministically) |
| $E[M]$ | mean number of transmissions $M$ |
| $T$ | number of link-transmissions, until all nodes receive a packet at least once |
| $E[T]$ | mean number of link-transmissions $T$ (deterministically) |
| $T^*$, $E[T^*]$ | number of link-transmissions, until all receivers receive a packet at least once in the expected sense ($E[times\ received] >= 1$) |
| $E[T^*]$ | mean number of link-transmissions $T^*$ |
| $1, 2, ....n$ | nodes |
| $1, 2, ...L$ | links |
| $p_i$ | probability of loss on link $i$ |
| $F_n(i)$ | $Pr${all nodes from n and below, receive the packet at least once, after i transmission} |
| $P_l$ | probability that a packet is transmitted across link l |

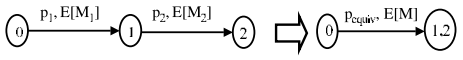
the electric circuits paradigm, all one needs to reduce an entire acyclic circuit are the rules for combining impedences in parallel or in tandem. Figure 2 shows the equivalent of two links in a row or in a star with respect to the $E[M]$ measure. Applying those rules starting from the leaves and proceeding up toward the source, leads to the calculation of $E[M]$ from the proxy.

Note however, that the computation of $E[M]$ can be exponential in the number of nodes $n$ in the general case, as we discuss in [17]. E.g. for a star with $n$ leaves, one has to find the $E[M]$ or $p_{equiv}$, considering the union of error events on links $1, 2, ...n$, i.e. $2^n$ subsets.

Let us now compare the simulated to the analytically computed $E[M]$ using a realistic example. Figure 3 shows the tree topology and the loss rates measured in [27] for a multicast session over the MBONE. This topology was used, among other studies, in [6] to evaluate the performance of different subgroups and hierarchies. Receivers first request lost packets from members of the same subgroup and if they fail, then they send a request upstream in the hierarchy. For details on the LGMP protocol, the reader is referred to [6]. The performance measure used by [6] to capture the network load was similar to $E[M]$: the total number of packets traveling in the network (including retransmissions) divided by the initial fixed number of packets sent by the source. The experiment is repeated many times (40000 original data packets), so we expect the load measured by simulation to be close to the ensemble average $E[M]$.

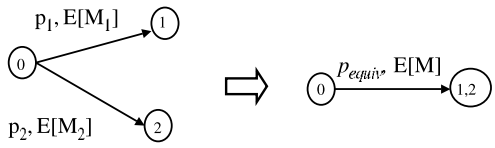
In the first scenario (I), [6] considers a flat hierarchy where all retransmissions come from the source. A second (II) and a third scenario (III) group the receivers into three separate subgroups shown in Figure 3. In scenario II, subgroups try first to locally recover from the loss and if they fail, they request the lost packet directly from the source $S$. Finally, in scenario III, subgroups $LG_1$, $LG_2$ request lost packets from the source while $LG_3$ requests lost packets from $LG_2$.

Figure 4 compares the analytical and simulation values of $E[M]$, i.e. the mean number of transmissions by the source to achieve reliable delivery, in these three scenarios. It is clear that the two values are very close. This shows that the model of Section III captures the important features of the problem (and the actual LGMP protocol [6]). This confirms that we can trust the analytically calculated measures (i) for performance evaluation and (ii) as our objective functions for the optimal placement of proxies. Furthermore, one could have come to the conclusions of [6] analytically, without the need for simulation. As a side comment, Figure 3 also demonstrates the benefit from using proxies in this specific, although limited, scenarios. Compared to the flat scenario, sub-




$$p_{equiv} = 1-(1-p_1)(1-p_2),$$
$$F_0(i) = \Pr[M(0) \le i] = 1 - p_{equiv}^i$$
$$E[M] = \frac{1}{1-p_{equiv}} = \frac{1}{(1-p_1)(1-p_2)} = E[M_1]\cdot E[M_2]$$

(a) two links in a row




$$F_0(i) = \Pr[M(0) \le i] = (1-p_1^i)(1-p_2^i) = 1 - p_1^i - p_2^i + (p_1 p_2)^i$$
$$\frac{1}{1-p_{equiv}} = E[M] = \sum_{i=0}^{\infty}[1-F_0(i)] = \sum_{i=0}^{\infty} p_1^i + p_2^i - (p_1 p_2)^i = \frac{1}{1-p_1} + \frac{1}{1-p_2} - \frac{1}{1-p_1 p_2}$$

(b) two links in a star

Fig. 2. Equivalent E[M] for reaching every node at least once

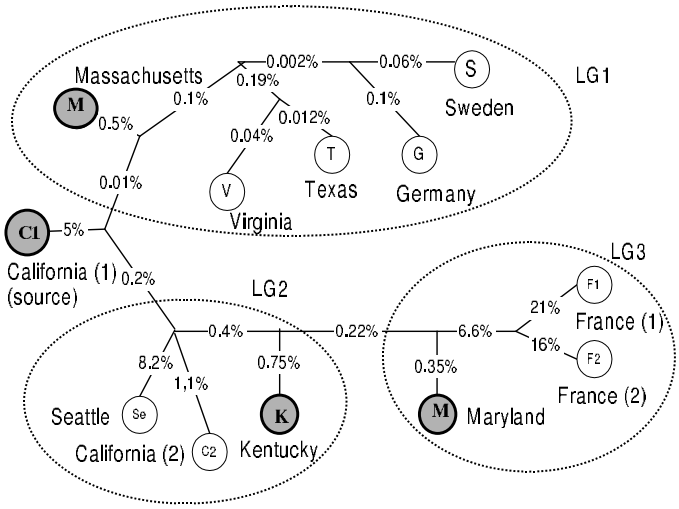



Fig. 3. Example MBONE topology

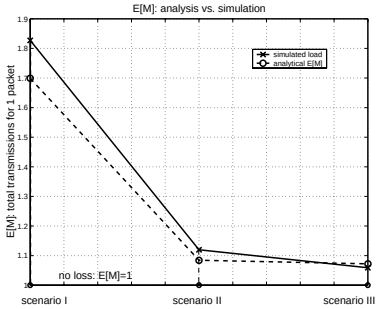



Fig. 4. Comparing the analytical to the measured E[M]

groupings I and II decrease $E[M]$ by half (to approximately $E[M] = 1$, the case for no loss).

### B. Bandwidth analysis

However, not all transmissions $M$ by the proxy cross all links, depending on the topology and where exactly the losses happen. Let us call ***link-transmissions*** or $T$**,** the number of links crossed until a multicast packet reaches every receiver at least once, either at the first or at its later transmissions. Its expected value $E[T]$ captures the average bandwidth used per multicast packet to reach the entire subgroup. Let us first consider the special topologies of a star and chain topologies and then the more general tree topology. $E[M]$ throughout all this section should be calculated as explained in section IV-A.

*1) Star topology:* Such special topologies are realistic in cases like those measured in [27] where loss happens mainly on the links connecting the source and the receivers to MBONE and are widely used in evaluating reliable multicast protocols, [10], [12], [20]. For star topologies, every transmission from the root crosses all $L$ links and the upper bound is tight:

$$E[T] = E[M] \cdot L \quad (1)$$

*2) Chain topology:* Consider the chain topology of Figure 5, where node *0* is the proxy and nodes $1, 2, ...L$ are the receivers. Let us first consider the same loss rate $p$ on all links. On a chain topology, every transmission affects only the links until the point of loss. The next retransmission starts from the proxy. The procedure stops when the last node *L* correctly receives the packet. So, the average number of link-transmissions can be calculated as the average time to absorption by the last state *L*, using the Markov chain of Figure 5. Being in state $i$, means that the packet coming from the proxy, was dropped on the link between nodes $(i-1)$ and $i$. Let $T_i$ be the mean number of link-transmissions for reliable delivery to all $L$ nodes, required, given that the packet already arrived to node $i$. (In the Markov-chain terminology, this is the" time" to absorption by state $L$ given that we are in state $i$.)

$$T_L = 0,\ T_{L-1} = 1 + pT_0,$$
$$T_{L-2} = 1 + pT_0 + (1-p)T_{L-1},\ ...,$$
$$T_0 = 1 + pT_0 + (1-p)T_1.$$

$$E[T] = T_0 = \tfrac{1}{(1-p)^L} + \tfrac{1}{(1-p)^{L-1}} + ... + \tfrac{1}{1-p} = \tfrac{1}{p}\left\{\tfrac{1}{(1-p)^L} - 1\right\} \quad (2)$$

The same procedure can be applied to calculate $E[T]$ on a non-uniform chain. as well as to graphs with degrees greater than one. However in the latter case the number of states in the Markov chain grows with the combination of nodes. This fact motivated us to look for another method appropriate for a general tree.

*3) Tree topology:* Consider an arbitrary tree**.** Consider a link $l$, rooted at node $n$. Let $P_n$ be the probability that a transmission by the proxy results in a transmission across link $l$. This is true if and only if the packet is correctly transmitted across all links from the proxy *0* until node $n$: $P_l = Pr\{a\ packet\ arrives\ at\ node\ n\} = \prod^{n}_{i:nodes\ on\ the\ path\ from\ 0\ to\ n}(1-p_i)$ .

We now calculate the average number of link-transmissions. Given the topology we can pre-calculate all entries $(n, P_n)$. $P_n$ also equals the percentage of transmissions that affect the link from $(n-1)$to $n$, i.e. $P_n{\cdot}M$ of $M$ total transmissions affect the link$(n-1, n)$. So a link $l$ is crossed $P_l{\cdot}E[M]$ times on average. Add over all links to get the total number of links crossed on average:

$$E[T] = E[\textstyle\sum_{links\ l} P_l \cdot M] = \sum_l E[M \cdot P_l] = \sum_l E[M] \cdot P_l = E[M] \cdot \sum_l P_l \quad (3)$$

Alternatively, we can reach the same result in a more formal way, [17]: Let $l$ be a link rooted at node $n$ and $X_{il} = \begin{cases} 1, & w.p.P_l \\ 0, & w.p.(1-P_l) \end{cases}$ be the random variable indicating whether transmission $i$ crosses link $l$ or not. There are $M$ transmissions in total, where $M$ is a random variable itself. The number of transmissions

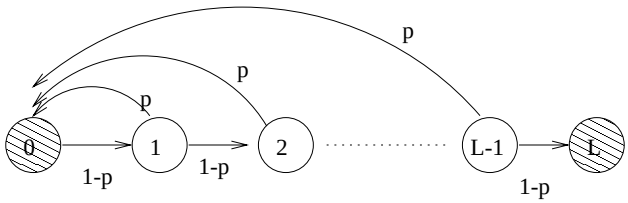



Fig. 5. a uniform chain

crossing link $l$ is $X_l = \sum_{i=1}^{M} X_{il}$ . On average: $E[X_l] = E[\sum_{i=1}^{M} X_{il}] = E[M] \cdot E[X_{il}] = E[M] \cdot P_l$. Add over all links $l = 1, 2, ...L$ and get the same result as in equation (3): $E[T] = E[M] \cdot \sum_l P_l$.

In order to verify equation (3), we apply it in the special cases of the uniform chain and the two-nodes-star and we obtain the same results that we obtained using the Markov-chain model in the chain case. Indeed, for the uniform chain of Figure 5 we get:

$$P_n = \prod_{i=0}^{i=n-1}(1-p) = (1-p)^{n-1},\ n = 1, 2, ..L-1, \quad E[M] = 1/(1-p)^L$$

$$E[T] = E[M] \cdot \sum_{n=1}^{n=L-1} P_n = \frac{1}{(1-p)^L} \cdot \sum_{n=1}^{L-1}(1-p)^{n-1} = \frac{1}{(1-p)^L} \cdot \frac{1-(1-p)^L}{1-(1-p)} = \frac{1}{p} \cdot \left\{\frac{1}{(1-p)^L} - 1\right\}$$

For a star with 2 identical receivers, we get $T_0 = 2\frac{1+2p}{1-p^2}$ using either of equations (1), (2), (3).

## V. Optimal placement of proxies

### A. Algorithm for the multicast case

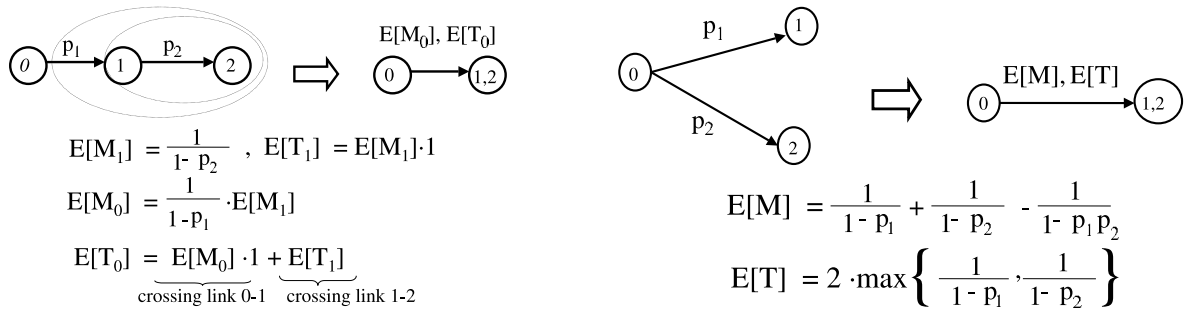
A good choice of a location for proxies becomes important to get the most benefit out of a proxy. In the example of subsection IV-A the $E[M]$ measure decreased almost by half by using an appropriate subgrouping. In this section we are looking for the best placement of a fixed number of proxies. The multicast nature of this location problem makes it fundamentally more difficult than the unicast location problem, as discussed in the Related Work section.

One approach could be to place proxies in a way to bound measure $E[M]$, i.e. the expected number of transmission required by the source and/or the proxies. We could try to (i) minimize the maximum $E[M]$ between all proxies or (ii) meet each proxy's individual bound. However, we choose to minimize a global additive measure: the total number of link-transmissions in order for all nodes to deterministically receive a packet at least once. The calculation of $E[T]$, in section IV-B, is exponential in the general case. We choose a slightly different cost function: the number of link-transmissions $T^*$ required for all members to receive a packet at least once, not deterministically, but in an expected sense. In the calculations of $E[T^*]$ we use the rules of Figure 6 for the equivalent link.[2]

We define the problem of optimal placement of proxies as follows. Given the topology of a multicast tree and its link loss rates $\{p_l\}$, place a fixed number of proxies $k$ in a way to minimize the total link-transmissions, $min\left\{\sum_{subgroup\ i=1}^{k} E[T_i^*]\right\}$, under the condition that each node is reached in the expected sense, i.e. *E[#times a node n received a packet]*$\geq 1,\ \forall n$.

To solve this problem, we follow a Dynamic Programming approach. First, we traverse the tree bottom-up and fill a table, like the one of Figure II, with entries the cost $E[T^*]$ for each subtree. Each row correspond to one subtree and a number of proxies $k_1$ allocated to this subtree. There are multiple rows for

[2]In the tandem of Figure 6a, node *1* needs to send a packet $\frac{1}{1-p_2}$ times on average to reach node *2*. For this to happen, node *0* needs to reach node *1* not just once but $\frac{1}{1-p_2}$ times and thus to send the packet $\frac{1}{1-p_2} \cdot \frac{1}{1-p_1}$ times on average. In the star scenario of Figure 6b, node *0* needs to multicast the packet $max\left\{\frac{1}{1-p_2}, \frac{1}{1-p_1}\right\}$ times to reach the weakest link once in an expected sense.




(a) two links in a star

(b) two links in a row

Fig. 6. Equivalent $E[T^*]$ , $E[M]$ for reaching every node on average at least once

each subtree, as we vary the number of proxies $k_1 \epsilon \{0, 1, 2, ...k\}$. Each column corresponds to the number of transmissions $t$ from the root of the subtree. Each entry of this table $cost(T_v, k_1, t)$ records the cost of the best possible configuration of placing $k_1$ proxies in the subtree $T_v$ given that we are going to send the message $t$ times in an expected sense. Once the table is complete, we backtrack and find the optimal allocation of the proxies.

The approach of keeping entries for each subtree and number of proxies and then backtracking through the table is typical in solving unicast location problems using Dynamic Programming, [24] [14]. However, the multicast nature of the problem forces us to keep additional entries. Indeed, the cost of a subtree when multicast is used depends not only on losses inside the subtree itself but also on unnecessary transmissions destined to other subtrees (*exposure* or *crying baby* problem). While calculating the cost of a subtree, the quality of links outside is unknown. However, it only affects the number of transmissions $t$ by the root of the subtree and the optimization can be performed with this knowledge.

There are two problems with the expected number of transmissions $t$: (i) it is not necessarily integral and (ii) it can be large. We address these problems when we discuss the running time. For the moment, let the maximum number of transmissions needed to reach all members in an expected sense be $t_m$. The columns of the table correspond to values of $t$ in the range $[1, t_m]$.

Let us now formally describe two algorithms: one that finds the placement with minimum cost in $O(n^3k^2)$ time and an approximation algorithm that achieves cost $(1 + \epsilon)optimum$, for a chosen precision $\epsilon$, in $O(nhk^2 \log n/\epsilon)$ time. Both algorithms use the dynamic programming approach outlined above (fill up a table and backtrack in it). They differ in the columns of their table: the range $[1, t_m]$ and the step of $t$ values in this range. The values of $t$ will be exact for our first algorithm and rounded up at some chosen precision for our second algorithm.

*1) Optimal solution:* Figure 7 shows a step in the dynamic programing: it adds a branch $T_R$ to a subtree $T_L$ and get the resulting tree $T_v$. Equivalently it tries to complete the rows of the table in Figure II, corresponding to subtree $T_v$, based on the already completed rows for $T_R$ and $T_L$. More formally, given $p_u$ and the best way to place $k_1$ proxies in $T_R$ and $k_2$ proxies in $T_L$, for a range of $k_1$, $k_2$ and transmissions $t$ values, we can compute the optimal placement of $k_1+k_2$ proxies in $T_v$. If the parent $v$ received the packet $t$ times, then the child $u$ received the packet $t(1 - p_u)$times.

Define COST-MULTI$(T_v, k_1, t)$ to be the minimum (expected) number of link-transmissions for the subtree $T_v$ which has $k_1$ proxies in all and its root attempts $t$ multicasts. This cost would depend on whether the root $v$

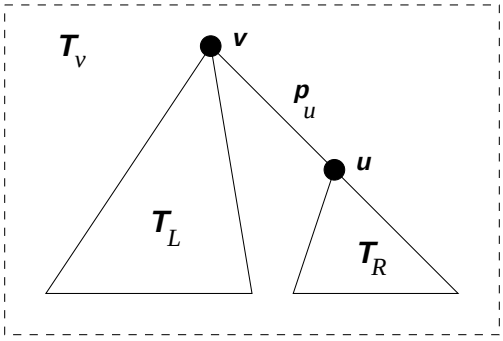



Fig. 7. A single step in the dynamic programming algorithm

has a proxy or not. It is convenient to separate these two cases.

Define ROOTED-MULTI$(T_v, k_1, t)$ to be the cost of the multicast given that the subtree $T_v$ has a proxy at its root, $k_1$ proxies in all and attempts $t$ multicasts.

Define UNROOTED-MULTI$(T_v, k_1, t)$ to be the cost of the multicast given that the subtree $T_v$ does not have a proxy at its root $v$ but does have $k_1$ proxies in all and attempts $t$ multicasts..

These costs are $\infty$ (unfeasible solutions) unless the expected number of packets received by every node is at least 1. The following computes COST-MULTI. "$min_{t'}$" means that if there is a proxy at the root of the subtree, it chooses how many times $t^{'}$ and over what configurations of proxies in its subtree, it should send the message to ensure that every node receives the message.

$$com(T_v, k_1, t) = \min[ \text{UNROOTED-MULTI}(T_v, k_1, t), \min_{t'} \{ \text{ROOTED-MULTI}(T_v, k_1, t') \}]$$

We update the tables ROOTED-MULTI and UNROOTED-MULTI for the subtree $T_v$ as shown below.[3] Their computation is similar except for the number of proxies in the subtree $T_L$ that is allowed to be 0 in the rooted but not in the unrooted case.

[3]Note that the weights are considered $w_u = 1$ here, but the notation will be used in interesting variants in subsection V-C.

$$\text{ROOTED-MULTI}(T_v, k_1, t) = \min_{k_2=0,\ldots,k_1-1}[ \text{ROOTED-MULTI}(T_L, k_1 - k_2, t) + tw_u + \text{COST-MULTI}(T_R, k_2, t(1-p_u))]$$

$$\text{UNROOTED-MULTI}(T_v, k_1, t) = \min_{k_2=0,\ldots,k_1}[ \text{UNROOTED-MULTI}(T_L, k_1 - k_2, t) + tw_u + \text{COST-MULTI}(T_R, k_2, t(1-p_u))]$$

Starting from the leaves in Figure 7 (or equivalently from the top of the table in Figure II) the dynamic program proceeds in this bottom-up fashion, until we reach the root. The final cost for the entire tree $T$ rooted at source and $k$ proxies is $\min_t$ COST-MULTI$(T, k, t)$.

*Boundary Conditions:* For a leaf node all three functions are 0 if $t \geq 1$ and $\infty$ otherwise. For any subtree $T$, if $k_1 = 0$ then, ROOTED-MULTI$(T, k_1, t)$ is $\infty$.

*Running Time:* Each row in the table corresponds to a subtree and a number of proxies $k_1 \epsilon \{0, 1, \ldots k\}$. There are $n$ subtrees and $k+1$ possible proxy values, therefore $O(nk)$ rows in the table. Each entry of tables UNROOTED-MULTI and ROOTED-MULTI can be filled in $O(k)$ time and once these are completed, each entry of table COST-MULTI takes $O(1)$ work.

The number of columns of the table, i.e. the number of possible different values of the expected number of transmissions $t$, is $O(n^2)$.

The number of transmissions needed in a subtree, is determined by the worst node, i.e. the node with the highest probability of not receiving the packet. This (worst) node may or may not belong to this subtree. There are at most $n^2$ paths and thus $O(n^2)$ paths from the worst node to its proxy for which we need to maintain a column (possible value of $t$).

Therefore, we showed that the above program finds an optimal placement of proxies in $O(n^3k^2)$ time.

*2) Approximation algorithm:* We are interested in an $(1+\epsilon)$ approximation of the minimum cost. To achieve it, we only fill up the columns of the table corresponding to values of $t$ that are powers of $(1+\delta)$ for some appropriately chosen $\delta(\epsilon)$. This makes the number of entries required at most $\frac{1}{\delta}\log t_m$ where $t_m = O(h\log 1/(1-p_m))$and $p_m$ is the maximum failure probability and $h$ the height of the tree. Notice, that the quantity $t(1-p_u)$ needs not be integral. Therefore we round it up to $t(1-p_u)^*$ to the nearest power of $(1+\delta)$, the precision we are willing to deal with. The equations of the previous section for ROOTED-MULTI, UNROOTED-MULTI, COST-MULTI should be modified to reflect this rounding. E.g. for ROOTED-MULTI:

$$\begin{aligned}\text{ROOTED-MULTI}(T_v,k_1,t) = \min_{k_2=0,\ldots,k_1-1}[\\ \text{ROOTED-MULTI}(T_L,k_1-k_2,t)+tw_u\\ +\ \text{COST-MULTI}(T_R,k_2,(t(1-p_u))^*)]\end{aligned}$$

We now show that the above program gives a $(1+\epsilon)$ approximation solution to the problem in $O(nhk^2\log n/\epsilon)$ time.

Consider the optimal solution and its cost $OPT$. Starting bottom-up, round up the expected number of transmissions of the optimal solution to powers of $(1+\delta)$. Since the height of the tree is $h$, the number of transmissions $t$ from the root will be at most $(1+\delta)^h$ times the corresponding number of transmissions in the optimal solution. Thus the cost of this solution is at most $(1+\delta)^h \cdot OPT$. If we choose $\delta = \frac{\epsilon}{2h}$, then the approximation factor is at most $(1+\epsilon)$ for $\epsilon < 1$. The solution we created has a $t$ that is a power of $(1+\delta)$ and is considered as a candidate solution for our algorithm which will provide a solution at least as good. Thus the $(1+\epsilon)$approximation of the minimum cost. However, the number of different values of transmissions is now $O(\frac{\log n}{\delta})$ which is $O(\frac{h\log n}{\epsilon})$. The number of rows is $nk$. Thus the total work is $O(nhk^2\log n/\epsilon)$.

*3) Example for the approximation algorithm:* Consider the graph of Figure 8 obtained by Figure 3 by omitting $California(2)$, $Maryland$ and the link connecting $LG_2, LG_3$ that have negligible loss rates.

Let us place two proxies. The table in Figure II shows the costs of the functions computed by the dynamic program. The row 'D & Lower' '$C(0)$' indicates the function COST-MULTI$(T_v,0,t)$, where $T_v$ is the entire subtree rooted at D. The notation '$R/U(k)$' denotes ROOTED-MULTI, and UNROOTED-MULTI for $k$ proxies. Identical functions are shown in the same row. The columns correspond to the parameter $t$, the expected number of trials. The letters in the square parenthesis (in case of $k \geq 1$) is the configuration of proxies corresponding to the solution for that entry – for the sake of brevity we present them only for COST-MULTI.

The final answer for 2 proxies is the first entry in the last row and corresponds to proxies on B and D.[4] We chose to round up $t$ in multiples of $0.05$ (additive precision instead of powers) and the dynamic program underesti-

[4]Tracing this back; this entry came from first entry in $U(2)$, which in turn came from the first entry in $C(2)$ for A and Lower. This entry comes from $C(2)$ of B and Lower; which is due to the minimum entry in the $R(2)$ row for this subtree. This means one proxy is at $B$. To trace it back further, we have to go up one column, and we have to place 1 proxy; it appears from row $C(1)$ second column for C and Lower. This appears from $U(1)$, second column; which in turn appears from $C(1)$ in D and Lower. $C(1)$ shows up from $R(1)$ entry: placing the other proxy at D.

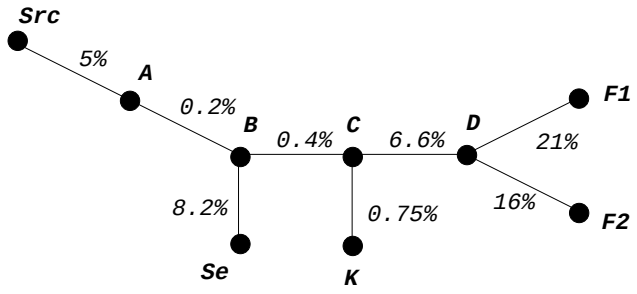



Fig. 8. A small example topology to demonstrate the computation of the approximation algorithm

mated the optimal cost by $3\%$.[5]

In case we wanted a single proxy, it is best to place it at D. This is interesting since the second column gives the minimum. The first entry shows that the source is interested in $1$ transmission, which mandates a proxy at B. However if the source is willing to transmit more, a better configuration comes from the second entry.

A more intersting example is the case where F1 is the source. Table V-A.3 shows the computation of the approximation. Consider the second last row which corresponds to having a single proxy for the whole tree.

Surprisingly, the minimum is *not* achieved by placing a proxy at $D$, but at $C$. It would have been natural to expect that $D$ would be the proxy to ensure that the $F1$–$D$ link is not overloaded. But it turned out that in that case $F2$ will act as a "crying baby" and as a result the whole network will see a lot of retransmissions.

### B. Algorithm for Unicast

If unicast is used for transmissions, the optimization problem is related to the standard $k$-*Median* problem on a tree. However in our case, probabilities of failures instead of distances are associated with links. Therefore the cost of every transmission may be different, since the transmission can fail on any edge.

*1) Approximation:* We can still apply our approach of section V-A.2, guess different values of the cost in powers of $(1+\delta)$ and get an $(1+\epsilon)$ approximation of the optimal in $O(nhk^2 \log n/\epsilon)$ time. This approach is useful in minimizing multiple criteria, see section V-C.

*2) Optimal solution:* The unicast problem is simpler than the multicast because the number of retransmissions only depends on the subtree and its path to the root. We can solve this problem optimally without guessing the number of transmissions $t$ from the root. We should just maintain a table (of size $O(nkh)$)with entries COST-UNI$(T_v, k_1, x)$ for the cost of the subtree $T_v$ with $k_1$ proxies and $x$ the immediate ancestor proxy of the root of $T_v$ ($v \neq x$).

The cost of sending a single packet from $x$ to $v$, PATH$(x, v)$ should be evaluated first in a top-down pass. PATH$(x, x) = 0$. From PATH$(x, y)$ we can compute PATH$(x, z) = \frac{1}{1-p_z}$ ( PATH$(x, y) + w_z$), where $z$ is a child of $y$, $p_z$ is the loss rate of link $(y, z)$, and $w_z$ is the cost of a transmission on this edge. Each node has at most $h$ ancestors [6] and the size of PATH is $O(nh)$.

[5]The actual cost of placing the proxies at these places is 8.95. After 1.265 expected trials from D, both F1 and F2 are expected to receive the packet. The cost of this subgroup is 2.53. We estimate it below at 2.5, since the rounding up treated an expectation more than 0.95 to be 1. Node C must send the message 1.07 times on average, in order for D to receive it with expectation 1. However, B needs to transmit 1.095 times due to Se; and this means that C transmits 1.09 times. The expected cost of this subgroup is 4.37. The subgroup between the source and B adds a cost of approximately 1.05+1.

[6]The number of subtrees is $n$ and $h$ the height. Notice that this way of formulating the dynamic program also solves the problem discussed in [14] in time $O(nhk^2)$ which is an improvement of their algorithm.

TABLE II

TABLE FOR THE DYNAMIC PROGRAM (APPROXIMATION SCHEME)

| **Subtree** | **Function** | **Expected Trials** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | t=1 | 1.05 | 1.1 | 1.15 | 1.2 | 1.25 | 1.3 | 1.35 | 1.4 |
| D & F1 (not F2) | R/U | | | | 1.15 | 1.2 | 1.25 | 1.3 | 1.35 | 1.4 |
| D, F1,F2 | R/U | | | | | | 2.5 | 2.6 | 2.7 | 2.8 |
| | C(0) | | | | | | 2.5 | 2.6 | 2.7 | 2.8 |
| | C(1)/C(2) proxy | 2.5 [D] | 2.5 [D] | 2.5 [D] | 2.5 [D] | 2.5 [D] | 2.5 [D] | 2.5 [D] | 2.5 [D] | 2.5 [D] |
| C & K | R/U | 1 | 1.05 | 1.1 | 1.15 | 1.2 | 1.25 | 1.3 | 1.35 | 1.4 |
| C,K,D,F1,F2 | U(0)/R(1) | | | | | | | 5.1 | 5.3 | 5.5 |
| | U(1)/R(2)/U(2) | | 4.6 | 4.7 | 4.8 | 4.9 | 5 | 5.1 | 5.2 | 5.3 |
| | C(0) | | | | | | | 5.1 | 5.3 | 5.5 |
| | C(1) proxy | 5.1 [C] | 4.6 [D] | 4.7 [D] | 4.8 [D] | 4.9 [D] | 5 [D] | 5.1 [C] | 5.1 [C] | 5.1 [C] |
| | C(2) proxies | 4.6 [C,D] | 4.6 [C,D] | 4.6 [C,D] | 4.6 [C,D] | 4.6 [C,D] | 4.6 [C,D] | 4.6 [C,D] | 4.6 [C,D] | 4.6 [C,D] |
| B & Se | R/U | | 1.05 | 1.1 | 1.15 | 1.2 | 1.25 | 1.3 | 1.35 | 1.4 |
| B & Lower | U(0)/R(1) | | | | | | | 7.7 | 8 | 8.3 |
| | U(1)/R(2) | | 6.7 | 6.9 | 7.1 | 7.3 | 7.5 | 7.7 | 7.8 | 7.9 |
| | U(2) | | 6.7 | 6.8 | 6.9 | 7 | 7.1 | 7.2 | 7.3 | 7.4 |
| | C(0) | | | | | | | 7.7 | 8 | 8.3 |
| | C(1) proxy | 7.7 [B] | 6.7 [D] | 6.9 [D] | 7.1 [D] | 7.3 [D] | 7.5 [D] | 7.7 [B] | 7.7 [B] | 7.7 [B] |
| | C(2) proxies | 6.7 [B,D] | 6.7 [B,D] | 6.7 [B,D] | 6.7 [B,D] | 6.7 [B,D] | 6.7 [B,D] | 6.7 [B,D] | 6.7 [B,D] | 6.7 [B,D] |
| A & Lower | U(0)/R(1) | | | | | | | 10.3 | 10.7 | 11.1 |
| | U(1)/R(2) | 8.7 | 7.75 | 8 | 8.25 | 8.5 | 8.75 | 9 | 9.05 | 9.1 |
| | U(2) | 7.7 | 7.75 | 7.8 | 7.85 | 7.9 | 7.95 | 8 | 8.05 | 8.1 |
| | C(0) | | | | | | | 10.3 | 10.7 | 11.1 |
| | C(1) proxy | 8.7 [B] | 7.75 [D] | 8 [D] | 8.25 [D] | 8.5 [D] | 8.75 [D] | 9 [B] | 9.05 [B] | 9.1 [B] |
| | C(2) proxies | 7.7 [B,D] | 7.75 [A,D] | 7.75 [A,D] | 7.75 [A,D] | 7.75 [A,D] | 7.75 [A,D] | 7.75 [A,D] | 7.75 [A,D] | 7.75 [A,D] |
| Tree from Src | C(1) | 9.7 | 8.8 | 9.1 | 9.4 | 9.7 | 10 | 10.3 | 10.4 | 10.5 |
| | proxy | [B] | [D] | [D] | [D] | [D] | [D] | [B] | [B] | [B] |
| | C(2) proxies | 8.7 [B,D] | 8.8 [A,D] | 8.85 [A,D] | 8.9 [A,D] | 8.95 [A,D] | 9 [A,D] | 9.05 [A,D] | 9.1 [A,D] | 9.15 [A,D] |

$$\text{COST-UNI}(T_v, k_1, x) = \min[$$
$$\quad \text{PATH}(x, v) + \text{ ROOTED-UNI}(T_v, k_1)$$
$$\quad \text{UNROOTED-UNI}(T_v, k_1, x)]$$

$$\text{ROOTED-UNI}(T_v, k_1) = \min_{k_2=0,\ldots,k_1-1}[$$
$$\quad \text{ROOTED-UNI}(T_L, k_1 - k_2) +$$
$$\quad \text{COST-UNI}(T_R, k_2, v)]$$

$$\text{UNROOTED-UNI}(T_v, k_1, x) = \min_{k_2=0,\ldots,k_1}[$$
$$\quad \text{UNROOTED-UNI}(T_L, k_1 - k_2, x) +$$
$$\quad \text{COST-UNI}(T_R, k_2, x)]$$

*Theorem 1:* The dynamic program shown above solves the problem of unicast with link failures on trees optimally in time $O(nhk^2)$. (We omit the proof).

### C. Extensions

Our dynamic programming approach generalizes to several interesting variants.

*Expected Minimum Cost and Delay:* Apart from the failure probability, we can also assign a parameter $w_u$ specific to each link. If this

TABLE III

TABLE SHOWING $F1$ AS SOURCE

| Subtree | Function | Expected Trials | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | t=1 | 1.05 | 1.1 | 1.15 | 1.2 | 1.25 | 1.3 | 1.35 | 1.4 |
| A & S | U/R/C | | 1.05 | 1.10 | 1.15 | 1.2 | 1.25 | 1.3 | 1.35 | 1.4 |
| B,Se | U/R/C | | 1.05 | 1.10 | 1.15 | 1.2 | 1.25 | 1.3 | 1.35 | 1.4 |
| B & Lower | U(0)/C(0)/R(1) | | 3.15 | 3.3 | 3.45 | 3.6 | 3.75 | 3.9 | 4.05 | 4.2 |
| | U(1)/R(2) | | 3.15 | 3.25 | 3.35 | 3.45 | 3.55 | 3.65 | 3.75 | 3.85 |
| | C(0) | | 3.15 | 3.3 | 3.45 | 3.6 | 3.75 | 3.9 | 4.05 | 4.2 |
| | C(1)/C(2) | 3.15 | 3.15 | 3.15 | 3.15 | 3.15 | 3.15 | 3.15 | 3.15 | 3.15 |
| | proxy | [B] | [B] | [B] | [B] | [B] | [B] | [B] | [B] | [B] |
| C,K | U/R/C | 1 | 1.05 | 1.10 | 1.15 | 1.2 | 1.25 | 1.3 | 1.35 | 1.4 |
| C & Lower | U(0)/R(1) | | 5.25 | 5.5 | 5.75 | 6 | 6.25 | 6.5 | 6.75 | 7 |
| | U(1)/R(2)/U(2) | 5.15 | 5.25 | 5.35 | 5.45 | 5.55 | 5.65 | 5.75 | 5.85 | 5.95 |
| | C(0) | | 5.25 | 5.5 | 5.75 | 6 | 6.25 | 6.5 | 6.75 | 7 |
| | C(1) | 5.15 | 5.25 | 5.25 | 5.25 | 5.25 | 5.25 | 5.25 | 5.25 | 5.25 |
| | proxy | [B] | [C] | [C] | [C] | [C] | [C] | [C] | [C] | [C] |
| | C(2) | 5.15 | 5.15 | 5.15 | 5.15 | 5.15 | 5.15 | 5.15 | 5.15 | 5.15 |
| | proxies | [B,C] | [B,C] | [B,C] | [B,C] | [B,C] | [B,C] | [B,C] | [B,C] | [B,C] |
| D & F2 | U/R/C | | | | 1.15 | 1.2 | 1.25 | 1.3 | 1.35 | 1.4 |
| D & Lower | U(0)/R(1) | | | | 7.75 | 8.05 | 8.35 | 8.65 | 8.95 | 9.2 |
| | U(1)/R(2) | | | | 7.55 | 7.65 | 7.75 | 7.85 | 7.95 | 8.05 |
| | U(2) | | | | 7.45 | 7.55 | 7.65 | 7.75 | 7.85 | 7.95 |
| | C(0) | | | | 7.75 | 8.05 | 8.35 | 8.65 | 8.95 | 9.2 |
| | C(1) | 7.75 | 7.75 | 7.75 | 7.55 | 7.65 | 7.75 | 7.75 | 7.75 | 7.75 |
| | proxy | [D] | [D] | [D] | [C] | [C] | [D] | [D] | [D] | [D] |
| | C(2) | 7.55 | 7.55 | 7.55 | 7.45 | 7.55 | 7.55 | 7.55 | 7.55 | 7.55 |
| | proxies | [C,D] | [C,D] | [C,D] | [B,D] | [C,D] | [C,D] | [C,D] | [C,D] | [C,D] |
| Whole tree | C(1) | | | | | | 9 | 9.05 | 9.1 | 8.95 |
| | proxy | | | | | | [D] | [D] | [D] | [C] |
| | C(2) | | | | | | 8.8 | 8.85 | 8.9 | 8.85 |
| | proxies | | | | | | [B,D] | [B,D] | [B,D] | [B,D] |

parameter represents delay, then we can use almost the same program to minimize for minimum expected delay. The only change needed is to take maximum of the delay for $T_L$ and the link plus $T_R$, instead of the sum. This parameter $w_u$ can also be used to allow network manager to favor some links, e.g. congested or expensive ones.

*Mixed Strategies:* A proxy may locally choose its own strategy, between multicast and unicast retransmissions. If there are a few isolated bad links, unicast retransmissions to them would save the rest of the group from exposure. If the loss happened on the shared part of the path, multicast might be better. Even this mixed scheme can be accommodated by the dynamic programming strategy, by recording the best between the multicast and unicast cost in the table.

*Variable Number of Proxies:* The larger the number of proxies the better the performance. At the limit, if all nodes become proxies, the scheme degenerates to hop-by-hop acknowledgment and the least bandwidth is used. However, proxies are costly entities and one would like to use a relatively small number of them and get adequate performance. In the example of section V-A.3, optimally placing one proxy decreased the cost by $25\%$, while the additional decrease from placing the second proxy is only $1\%$. The problem studied in this paper, considers a fixed number of proxies $k$. However, the table of the dynamic program contains anyway the minimum cost $T^*$ for all number of proxies $k_1 : 1, 2, ...K$. By looking at the values $T^*(k_1)$ and one could choose an appropriate $k_1$ beyond

which the incremental decrease in cost $T^*$ is small.

## VI. Conclusions

In this paper, we study the hierarchical reliable multicast problem. In the first part of the paper we partitioned an entire group into subgroups and evaluated the performance of each subgroup in terms of two appropriate measures. In the second part, we studied the problem of optimal placement of proxies in order to minimize a bandwidth cost function. An optimal and an approximation algorithm were given for both the multicast and the unicast problem. We focused on the multicast case, which had not been solved before as an optimization problem. We also applied our algorithm on a realistic example obtained from MBONE measurements. We outlined how our approach can be applied to a variety of related problems.

## References


[1] P. Bhagwat, P. Mishra, S. Tripathi, "Effect of Topology on Performance of Reliable Multicast Communication", *in Proc. of IEEE INFOCOM '94, pp.602-609*, Toronto, Ontario, Canada, March 1994.

[2] M. Charikar, S. Guha, E. Tardos, D. Shmoys, "A Constant factor approximation for $K$-Median Problem" , *in Proc. 31st Annual Symposium on the Theory of Computing, pp. 1-10*, Atlanta, Georgia, USA, March 1999.

[3] Y. Chawathe, S. McCanne, E. Brewer, "RMX: Reliable Multicast in Heterogeneous Networks", *in Proc. IEEE INFOCOM '00, pp.795-804*, March 2000, Tel-Aviv, Israel.

[4] M. Daskin, "Network and Discrete Location Theory", *J.Wiley, 1995*.

[5] M. Handley, "An examination of MBONE Performance", *Research Report ISI/RR-97-450*, USC/ISI, Jan. 1997.

[6] M. Hofmann, "Impact of Virtual Group Structure on multicast performance", *in Proc. 4th Int. COST 237 Workshop, pp. 165-180*, Lisboa, Portugal, Dec. 1997.

[7] H. Holbrook, S. Singhal, D. Cheriton, "Log-Based Receiver-Reliable Multicast for Distributed Interactive Simulation" , *in Proc. ACM SIGCOMM '95, pp.328-341*, Cambridge, MA, USA, August 1995.

[8] H. Holbrook, D. Cheriton, "IP Multicast Channels: Express support for Large scale single-source applications", *in Proc. ACM SIGCOMM '99, pp.65-68*, Cambridge, MA, USA, Sept. 1999.

[9] Starburst Communications, Press Releases, *http://cad.ntu-kpi.liev.ua/~netlib/Nets/Mbone/-www.starburst.com-/*

[10] S. Kasera, J. Kurose, D. Towsley, "A comparison of server and receiver-based local recovery approaches for scalable reliable multicast", *in Proc. IEEE INFOCOM '98, pp.988-995*, San Francisco, CA, USA, March 1998.

[11] P. Krishnan, D. Raz, Y. Shavitt, "The Cache Location problem", *in IEEE Trans. on Networking, 8 (5): pp. 568-582*, Oct. 2000.

[12] M. Lacher, J. Nonnenmacher, E. Biersack, "Performance comparison of centralaized versus distributed error recovery for reliable multicast", *in IEEE Trans. on Networking, Vol.8, No.2, pp. 224-239*, April 2000.

[13] B. Levine, S. Paul, J.J. Garcia-Luna-Aceves. "Organizing multicast receivers deterministically by packet-loss correlation", *in Proc. ACM Multimedia '98, pp.201-210*, Oslo, Norway, Sept. 1998.

[14] B. Li, M. Golin, G. Italiano, X. Deng, K. Sohraby, "On the optimal placement of web proxies in the Internet", *in Proc. IEEE INFOCOM '99, pp.1282-1290*, New York, NY, USA, March 1999.

[15] D. Li, D. Cheriton," OTERS: A Reliable Multicast Protocol", *in Proc. ICNP '98, pp. 237-245*, Austin, TX, USA, Oct. 1998.

[16] J. Lin ,S . Paul, "RMTP: A Reliable Multicast Transport Protocol", *in Proc. IEEE INFOCOM '96, pp. 1414-1424*, San Francisco, CA, USA, March 1996.

[17] A. Markopoulou, F.Tobagi, "Performance analysis of Hierarchical Reliable Multicast", *in Proc. of ACM NGC' 00, pp. 27*-35, Stanford, CA, USA, Nov. 2000.

[18] J. Nonnenmacher, E. Biersack, "Performance Modeling of Reliable Multicast Transmission", *in Proc. of IEEE INFOCOM '97, pp. 471-479*, Kobe, Japan, Apr. 1997.

[19] J. Nonnenmacher, E. Biersack, D. Towsley, "Parity-Based Loss Recovery for Reliable Multicast Transmission", *IEEE Trans. on Networking, 6(4): pp. 349-361*, August 1998.

[20] J. Nonnenmacher, M. Lacher, M. Jung, E. Biersack, G. Carle, "How bad is Reliable Multicast without Local Recovery?", *in Proc. IEEE INFOCOM '98, pp. 972-979*, San Francisco, CA, USA, March 1998.

[21] C. Papadopoulos, G. Parulkar, G. Varghese, "An Error Control Scheme for Large-Scale Multicast Applications", *in Proc. IEEE INFOCOM '98, pp.1188-1196*, San Francisco, CA, USA, March 1998.

[22] S. Ratnasamy, S. McCanne,"Inference of Multicast Trees and Bottleneck Bandwidths using End-to-end Measurements", *in Proc. IEEE INFOCOM '99, pp.353-363*, New York, NY, USA, March 1999.

[23] T. Speakman, J. Crowcroft, J. Gemmell, D.Farinacci, S.Lin, D.Leshchiner, M.Luby, T.Montgomery, L.Rizzo, A.Tweedly, N. Bhaskar, R.Edmonstone, R.Sumanasekera, L. Visicano, "PGM Reliable Transport Protocol Specification", *IETF RFC 3208*, Dec. 2001.

[24] A. Tamir, "An $O(pn^2)$ algorithm for the p-median and related problems on tree graphs", *Oper. Research Letters*, 19: *pp. 59-94*, Feb. 1996.

[25] D. Towsley, "Network tomography using end-to-end multicast measurements", *Proc. of ACM NGC '00, pp. 93-101*, Stanford, CA, USA, Nov. 2000.

[26] D.Towsley, J.Kurose, S.Pingali, "A Comparison of sender and receiver-initiated reliable multicast protocols", *in IEEE JSAC, Vol.15, No3, pp.* , April 1997.

[27] M. Yajnik, J. Kurose, D. Towsley, “Packet Loss Correlation in the MBONE”, *in Proc. IEEE Global Internet Conference ’96, pp.94-99,* Nov.1996.

[28] R. Yavatkar, J. Griffioen, M. Sudan, A Reliable Dissemination Protocol for Interactive Collaborative Applications , *in Proc. ACM MULTIMEDIA ’95, pp.333-344,* Boston, MA, USA, Nov.1995.

[29] L. Zhang, S. Floyd, V. Jacobson, “Adaptive web caching”, *NLANR Web Cache Workshop*, Boulder, CO, June 1997.