Accounting Mechanism for Membership Size-Dependent Pricing of Multicast Traffic

David Breitgand[1], Danny Dolev[1], and Danny Raz[2]

[1] School of Engineering and Computer Science, The Hebrew University, Givat-Ram
91904 Jerusalem, Israel
{davb, dolev}@cs.huji.ac.il
[2] Department of Computer Science, The Technion
32000 Haifa, Israel
{danny}@cs.technion.ac.il

Abstract. A number of schemes for membership size-dependent pricing of multicast traffic have been proposed recently. The enabling accounting technology, however, is lagging behind. Retaining an exact count of the active receivers is very difficult. To target an accounting mechanism, we suggest novel monitoring algorithms that combine a hierarchical control structure with event-driven monitoring. The efficiency and tradeoffs of the proposed solutions are thoroughly studied through simulations using both synthetic, and real MBone workloads. As we show, depending on the workload and the required accuracy, we can reduce the monitoring traffic by factor of 2 to 10 compared to other known alternatives.

1 Introduction

IP multicast is an important technology that allows distributing large volumes of data to millions of receivers in an economic fashion. Usually, IP multicast tree construction is guided by receivers. The network provider bears the control overhead of joining the group's tree when the first receiver appears in the domain. The provider also absorbs the overhead of maintaining this new tree branch for as long as there exists at least one receiver in the domain.

In general, network providers are interested in encouraging both receivers and senders to use the most cost-efficient network service type (*i.e.*, unicast or multicast). The senders, however, need to assess the utility of transmitting into a specific multicast group, *i.e.*, the size of the audience that they can reach.

Recently, a number of proposals [1–3] addressed these issues via group size-dependent pricing of transmissions. However, it is very difficult to find out the exact number of host receivers in a multicast group at any given moment [4].

We argue that for applications like pricing and billing it may suffice to know at any given moment that the group size is between some two predefined threshold values. These bounds determine the group's *rank*. Traffic being sent into the group is priced depending on the rank of the group, and the price does not change until the group changes its rank.

The group size estimation is disseminated to the multicast ISPs worldwide using a peer-to-peer application level network being set up among the ISPs as is explained later. The ISPs then *locally* compute the price of transmitting a unit of bandwidth to the group, and present this price to senders in their domain. Senders are also presented with the group rank information, so that they can assess the utility of reaching a particular audience size by transmitting to a specific group while paying the price charged by their local ISP.

We assume that the charging scheme will be applied locally both to receivers and senders by their respective ISPs as in [2, 17]. However, we do not require that ISPs will be involved in additional cost apportionment process among themselves based on the local population of receivers. If they do, the question of trust should be solved, since ISPs may have incentive to exaggerate the number of their local receivers to make extra profit when reassigning the costs with other ISPs, as is the case in the spilt-edge pricing proposal in [16], or EXPRESS [3], for instance.

Additional factors such as spatial distribution of users, time of day, *etc.* may be taken into account by the pricing scheme. However, in this paper we focus specifically on the group size estimation. We propose the following generic approach to monitoring the size of a multicast group for the sake of pricing. A specific pricing scheme requires knowing the group size up to some predefined constant factor $1 \leq \gamma < \infty$. Suppose that initially the group size is known to be S_0. As long as the group size S remains such that $\frac{S_0}{\gamma} < S < \gamma \cdot S_0$, there is no need in any additional accounting activity because it brings no benefit. We say that γ defines the *low* and *high* thresholds values. Multicast ISPs need to be notified about the new group size *if and only if* either the low, or high threshold value is violated. This way each ISP knows the group size up to factor γ at all times. Given this estimation each ISP locally computes the price of transmitting a unit of bandwidth to the given group, and presents it to the local senders. Our accounting mechanism ignores membership fluctuations that are insufficient to change the rank of the group.

There is a trade-off between the accuracy of the group size approximation, and the communication cost paid for it. Thus different pricing schemes may choose different QoS levels of accounting for controlling their overhead, while still maintaining a precision that suits them.

When we consider the typical lifespan of a multicast group, we identify three successive periods: membership build-up, saturation, and membership termination. Membership build-up and membership termination phases are characterized by rapid membership changes. However, these two periods are short relatively to the saturation phase. Therefore, it is natural to consider an event-driven (*reactive*) monitoring scheme that generates more traffic in the first and last phases of the multicast transmission, and conserves bandwidth in the second stage. Thus, the total communication overhead of reactive group size accounting will be lower compared to a monitoring scheme based on periodic polling.

When we consider multicast, it is very natural to assume that we have a (bounded out-degree) tree structure where the leaves are the end routers that have a local count regarding the number of participants in the group. Since we

want to know the total size of the group, our aim is to calculate the sum of the values of the leaves. This can be done in a hierarchical way in the internal nodes of the tree. Every polling step consists of calculating the sum of the values up the tree and disseminating the value from the root to all the leaves if needed.

This hierarchical polling process is a natural way to overcome the feedback implosion problem, and it also saves a factor of $\log N$ messages in each polling interval, where N being the number of ISPs. The next step is to combine it with the reactive monitoring process. This leads us to constructing a hierarchical and reactive monitoring algorithm that allows to maintain the multicast membership size estimation with very low communication overhead.

Unfortunately, deployment of this mechanism would require changes to most of the existing standard multicast routing protocols. We are not proposing any changes to the existing routing protocols, but rather suggest that the accounting function to be implemented as a separate service over a peer-to-peer network of ISPs, be orthogonal to the routing functionality. In this respect, our proposal constitutes a protocol independent multicast accounting mechanism, in an analogy to PIM [15] that promotes protocol independence with respect to routing, and protocol independent multicast pricing proposal [2].

2 Related Work

The cost-effectiveness of multicast was first expressed in [1] as a simple power-law function of the active membership size. Namely, $N^{k-1} = \frac{L_m}{L_u N}$, where L_m is the total number of links in a multicast tree, L_u is the average length of the unicast route, and N is the number of leaf routers. The exponent k is called *Economy of Scale* (EoS) factor, $0 \leq k \leq 1$.

This power law suggests a straightforward multicast pricing scheme: $P_m = N \cdot (P_u)^k$ where P_u being the price of unicast, and P_m being the price of multicast. It should be noted that in order for the sender to decide which service to use, it should know the number of individual hosts, N_h, and not only N. The sender will prefer the multicast service over the unicast one as long as $P_m < P_u \cdot N_h$.

It is assumed in [1] that N, the number of leaf multicast routers, can be discovered using distributed monitoring. However, no specific accounting mechanism has been suggested, and the problem of finding N_h was left open. In this work, we provide a solution that can be used to discover both the number of leaf routers, and that of the individual hosts.

The reactive monitoring algorithms presented in [5] constitute the starting point for this research. However, these algorithms cannot be used as is for multicast group size estimation, since they ignore the network distances, the feedback implosion problem, and the statistical nature of the monitored data is different.

In [3] a new IP multicast routing protocol called EXPRESS was proposed. Part of the proposal is ECMP, the management protocol that accounts for the group size at the granularity of individual hosts. The event-driven version of ECMP is close in spirit to our hierarchical reactive monitoring algorithm of

Section 5. However, our algorithms can be used with any multicast protocol, and as we show later, is superior to ECMP in terms of bandwidth.

In [12], a fast and cost-effective algorithm for estimating the number of leaf multicast routers was presented. This algorithm leverages on the empirical properties of the realistic multicast topologies that exhibit a linear ratio between the number of high degree nodes, and the number of leaves. The algorithm as is does not help finding the number of individual hosts. However, it provides a better initial estimation for the probabilistic methods for faster convergence.

The need to know the number of active receivers in a multicast session often arises at the transport and application levels [8]. To this end, a number of probabilistic estimation methods have been proposed [?,9–11]. A comprehensive comparative study of all probabilistic estimation algorithms is beyond the scope of this paper. In Section 6, we define a generic probabilistic monitoring algorithm that embodies the functionality common to all probabilistic group size estimation methods. We use this generic algorithm as a baseline.

3 Model and Problem Definition

Let $G(V, E)$ be partially synchronous network. Let $V' \subseteq V$ be a set of nodes representing the ISPs participating in a multicast group spanning tree. We assume that there exists a reliable unicast channel between each pair of nodes in V' in addition to the spanning tree of the multicast group. Each $v \in V'$, maintains a variable $x_i \in I^+ \bigcup \{0\}$, $x_i < \infty$. By $x_i(t)$ we denote the value of x_i at time $t \in I^+ \bigcup \{0\}$. The physical meaning of this variable is the number of active receivers in the domain represented by node v.

The system proceeds in rounds where all messages sent at the beginning of a round are received by their destinations by the end of the round. Each round is completed within one time unit. In practice, the actual value of the time unit may be important. This is the maximal round trip time between any two neighbors in the network. In addition, each node in V' can send or receive no more than a fixed number of messages in a round. All messages that exceed this threshold get lost. All messages are assumed to be of a fixed limited length.

Definition 1. *Let $\omega(n)$ be some function assuming non-negative integer values, and bounded from above. $f(t) = f(x_1(t), x_2(t), ..., x_N(t))$ is ω-constant function if and only if it is being constant on the semi-open intervals of length ω: $[0, \omega), [\omega, 2\omega)$, etc.*

Every node knows only its local variable x_i. The estimation of the total membership size is computed in a fully distributed manner that involves network communication among some nodes. The communication delay being inherent to any distributed algorithm, precludes knowing the exact membership count at any given moment. We compute the $O(\log^2 N)$-constant approximation of our target function $f(t) = \sum_{i=1}^{N=|V'|} x_i(t)$. In other words, we do not account for the fluctuations in the target function value if these changes last for less than $O(\log^2 N)$ time units.

Given synchronous reliable network $G(V,E)$, a collection of variables $x_1(t), x_2(t), ..., x_N(t) \in I^+ \bigcup \{0\}$, each being maintained by some $v \in V' \subseteq V$, a target function $f(t) = \sum_{i=1}^{n=|V'|} x_i(t)$, being ω-constant function, a set of threshold values $X_1, X_2, ..., X_m$, find an algorithm that guarantees that if and only if at time t_a a global threshold violation event occurs, all nodes in V' are notified about this event by time $t_a + \omega$.

Fig. 1. Problem Definition

Definition 2. *If at time t, $f(t) \leq H$, and at time t_a, $t_a > t$ and $f(t_a) > H$, t_a being minimal such time, we say that a* global upper threshold event *(or simply* cross-over*) has occurred at t_a.*

Cross-down is defined similarly. Figure 3 provides the rigorous definition of the problem. Let $C(A, \delta)$ denote the total number of messages sent by algorithm A over time δ. In this work, we compare the algorithms using this metric. *It is important to stress that we count traversing of each hop between any two neighbors in the network as a separate message.*

4 Estimation Methodology

The estimation accuracy requirements come from a specific pricing scheme. The following general methodologies are conceivable.

- **Relative to Total (R2T)** : the purpose is to show what percentage of the total maximum population is active. The total space is partitioned into percentiles that implicitly define the sensitivity parameter, γ. R2T is applicable when the maximal size is known in advance.
- **Relative to Last (R2L)**: the purpose is to capture the change of the group size relatively to its last known size. Precision of the estimation, γ, should be specified explicitly. At every given moment the estimation provides the group size up to a multiplicative factor γ. R2L is more flexible and scalable.

5 Hierarchical Reactive Monitoring Algorithm

This section briefs Hierarchical Reactive Monitoring Algorithm (HRMA) that overcomes the problems of feedback implosion and bandwidth inefficiency that are common to all flat algorithms[3]. The common structure of all flat algorithms is as follows. The monitored targets can directly communicate with a distinguished *monitor* over a reliable unicast channel. The relay can communicate with the targets using either multicast, or unicast channels. When a target detects a local

[3] A detailed presentation of the algorithm along with the thorough discussion on flat event-driven monitoring can be found in [4]

threshold event, it reports to the monitor. The latter either initiates a global poll of all targets, or discards the reported event. This structure is inherently unsuitable for our context because as the group size grows, the monitor becomes overwhelmed by the reports (the feedback implosion problem), many of which do not have global implications (bandwidth inefficiency).

The basic idea of HRMA is as follows. Instead of using the direct unicast communication channels for reporting local values of ISP variables, and the multicast communication channel for propagating the polling requests, we organize the nodes into a logical hierarchy in which the same node may act at different levels. The inter-nodal links are realized as reliable unicast FIFO channels.

The optimal hierarchy formation and maintenance need a special study since one can look for several optimization functions. For the sake of simplicity, we assume that the logical hierarchy is a full binary tree.

The leaf nodes (ISPs) report their traps up the hierarchy, and the intermediate nodes act as local relays deciding whether to propagate the trap further toward the root depending on the total values reported by their children.

Each node has values l and h that hold the current low and high thresholds for the subtree rooted at this node. The root also has an array of threshold values, *thresh*, as determined by the ranking service.

Initially, a converge-cast is performed to compute the total number of active receivers at time 0, S_0. The root calculates the leaf thresholds as $l = \frac{S_0}{N \cdot \gamma}$, $h = \frac{S_0 \cdot \gamma}{N}$, and multicasts them on the tree. Initially, the intermediate nodes do not have l and h. They compute them upon receiving traps from their children.

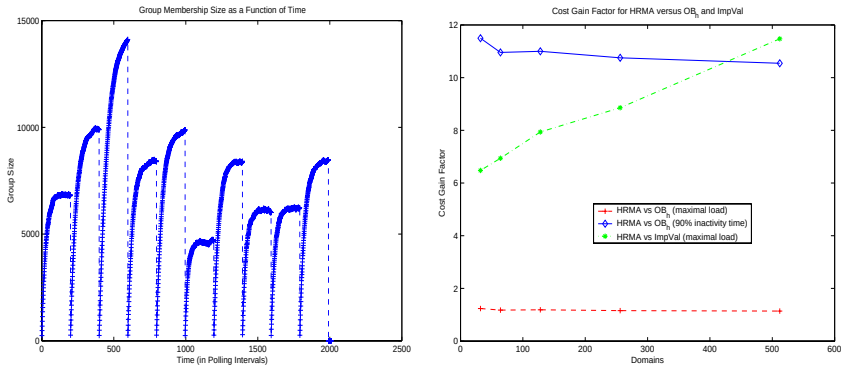
When a node sends a trap to its father in the tree, we say that it starts verifying the threshold condition. This process ends either at some intermediate node, or at the root of the whole tree. A node stops the propagation if it discovers that neither high, nor low threshold of the subtree being rooted at it is violated. In this case, the node does not send more messages. The leaf nodes in its subtree will discover that no global event has occurred after a timeout. If the root discovers that a global threshold event has happened, it sends an *update* message down the hierarchy informing the ISPs about the new group size estimation. It is not difficult to verify that if the root informs about the global event, then there was at least one local threshold violation. In the worst case, any local event is verified after $O(\log^2 N)$ time units. Thus, *HRMA* correctly solves the membership size monitoring problem defined in Figure 3.

6 HRMA Performance

First, we establish a baseline algorithm against which the cost gain factor of *HRMA* will be computed. To this end, we define a *Hierarchical Straightforward Monitoring Algorithm*, denoted OB_h that uses the hierarchical structure as explained in Figure 6. We argue that evaluating against OB_h, also is indicative for other known alternatives. Obviously, $C(OB_h, \delta) = 2(N-1)(\frac{\delta}{2\log^2 N} + e_\delta)$, where e_δ being the number of TVEs detected by the root. It is easy to observe that $C(HRMA, \delta)$ may be higher than $C(OB_h, \delta)$.

The leaf nodes initiate a converge-cast of their local values towards the root of the logical hierarchy every $2\log^2 N$ time units. The root checks whether a global TVE occurs, and if so, multicasts the new group size using the same hierarchy.

Fig. 2. Hierarchical Straightforward Monitoring Algorithm (OB_h).



(a) Sample Synthetic Membership trace (b) *HRMA* : average of 200 runs

Fig. 3. Performance of HRMA

An alternative baseline is offered by popular probabilistic polling techniques. A generic probabilistic estimation procedure communicates directly with the receivers, and not with their ISPs. The algorithm proceeds in rounds. In each round, an increasing probability of response is multicast by the dedicated node, usually the source. The receivers respond with the advertised probability. Each such response constitutes a single Bernoulli trial. Based on the number of the responses (that are not aggregated), an estimation of the total number of receivers is performed. Although there are many more details involved, this *Generic Probabilistic (GeP)* algorithm captures the essence of the methodology. *GeP* avoids the problem of feedback implosion. However, it is easy to see that the communication cost of *GeP* approximately equals that of OB_h. Indeed, in our model, the cost of multicasting the response probability in *GeP* equals the cost of converge-cast in OB_h, and the cost of informing the members about the TVE is the same in both techniques. Thus, comparing *HRMA* to OB_h is indicative of *GeP*.

Yet another alternative is ECMP of EXPRESS. It is again easy to observe that the communication cost incurred by ECMP is *at least* that of OB_h when the maximal silence period of ECMP is set to $O(2\log^2 N)$. Thus, comparing *HRMA* to OB_h is sufficient is indicative of the relative ECMP performance.

In this work, we focus on the short-lived multicast sessions. The reason for this is that shorter sessions are less stable, and, therefore are more challenging. Following [6, 13] we model the inter-arrival and stay times of the multicast receivers as exponentially distributed variates with means μ_{arr}, μ_{stay}. Throughout

the simulations we assume that users are uniformly distributed over ISP domains. We assume a fixed ratio of the users over domains, thus having a knowledge of the total population size. The thresholds are computed using R2T estimation methodology with the group ranks being 15-percentiles.

Figure 6(a) shows a sample synthetic trace of the group membership for 512 multicast domains, 15360 potential receivers acting according to the exponential model above, and being uniformly distributed among the domains with the average user/domain ratio being 30. There are 10 different multicast sessions per trace. All sessions are of equal length. This is the maximal load scenario. In reality the membership not necessarily drops to 0 at the end of each session, and there exist larger periods of stability during, and between the sessions. The basic scenario can be changed by adding stability periods to the trace that freeze the membership activity for the certain percentage of the total trace duration.

Figure 6(b) shows the average performance of *HRMA* versus OB_h under the maximal load condition, and for 90% inactivity time trace. The same figure also shows the performance of *HRMA* versus ImpVal (the most efficient flat event driven algorithm of [5]) under the maximal load. Under a very high probability of violating a local threshold, *HRMA* produces a message cost-gain factor of 14%-20%. Notice that this cost-gain remains constant with the number of domains.

The relative cost-gain of *HRMA* versus ImpVal grows with the number of domains. *HRMA* does not save communication on multicasting the notifications about the global TVEs. It saves communication by suppressing the non-significant local threshold violation events though. In our traces, *HRMA* pays $\approx 1.5 * N$ messages per round. This is due to the fact that some edges are traversed twice: one message to solicit the value, and another one to report the value itself. In the same circumstances, ImpVal pays $\approx 2 * N \log N$ messages per round. Thus, the expected improvement is around $1.33 * \log N$ and is indeed expected to grow with the number of domains, N. These are very important observations suggesting that *HRMA* is scalable.

7 Adaptive HRMA

Basic *HRMA* is not adaptive. Namely, if there exist local threshold events, but there is no global TVE, then the verification process stops at some level in the hierarchy, However, *HRMA* learns no lesson from this, and if the situation remains similar also for the next step, the unnecessary verification will be repeated.

There is an apparent trade-off between leaving the local thresholds in the subtree intact in case of a false alarm in this subtree, and changing them. The latter requires multicasting in the scope of this subtree, and therefore increases the communication cost of the algorithm in this round. However, if it succeeds in tuning the thresholds in such a way that no false alarms would follow, the total communication cost of the algorithm will be reduced over time. To this end, we define two simple modifications to the basic *HRMA*.

- **Oblivious Adaptive HRMA:** When a false alarm is detected at time t by the root of some subtree (an intermediate node i), this node computes

$\alpha_1 = f_i(t)/l$, and $\alpha_2 = h/f_i(t)$ where $f_i(t)$ being the sum of the leaf variables of the subtree rooted at i, and l, h being the lower and higher local thresholds of this node respectively. Then, i multicasts the UPDATE message containing α_1, α_2, and indicating that there was no global threshold event. The nodes in the subtree recalculate their local thresholds using these values as they would upon reception of this message from the global root. However they do not change their current estimation of the group size.

– **Conscious Adaptive HRMA:** The main difference from Oblivious is that the root of the subtree computes the chance of suppressing local TVEs of no global significance. It compares the distance of the current value of the sum for its subtree with the median of the interval defined by its l, and h threshold values. The chances of suppressing false alarms are highest when the current value of the subtree is close to the median, and decrease as it gets close to either l, or h. Accordingly, Conscious Adaptive *HRMA* sends UPDATE message with the higher probability in the former case, and lowers the probability of sending an UPDATE message in the latter case.

Figure 8(a) demonstrates the adaptivity trade-off. For a fixed number of domains, we vary the trace duration. As one would expect, for shorter sessions the adaptivity heuristics are inferior to the basic form of *HRMA*. However, as sessions become longer, they become advantageous over the basic *HRMA*. Oblivious Adaptive *HRMA* is always inferior to the the Conscious and and Basic variants when sessions are short. This is natural since Oblivious Adaptive *HRMA* needs more time to compensate for the extra costs of updating the thresholds.

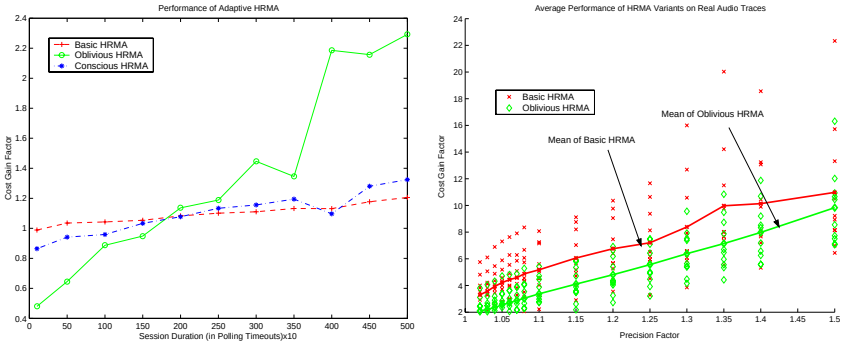
8 Validation with Real Traces

In this section we evaluate *HRMA* vs. Ob_h using the real membership traces collected using *mlisten* tool [6]. The purpose of this study is to inspect the behavior of *HRMA* when statistical assumptions on the user behavior used so far are not necessarily preserved.

Most multicast groups that were present in the traces were quite small ranging from a few users to a few tens of users at any given moment, and have not exhibited particularly high activity.[4] However, this makes the data even more interesting. So far our algorithms have been evaluated on the synthetic data that simulated very large groups (thousands of highly dynamic users) spanning hundreds of domains. This may be way too far from the current state of the Internet. The question that we asked was whether the hierarchical reactive monitoring will be useful also for estimating the size of very small groups in such a way that it would be useful for pricing applications.

To this end, we used R2L estimation methodology that defines only a single pair of thresholds $H \stackrel{\text{def}}{=} \gamma * f(0)$, $L \stackrel{\text{def}}{=} f(0)/\gamma$, where $f(0)$ is the total size of the group as measured at time 0. The local thresholds l, h are computed as before. If at some time instance t a global TVE is detected, *HRMA* re-calculates L, H

[4] See [7] for discussion on the reasons.



(a) Adaptive HRMA on the synthetic dataset (b) HRMA on the real traces

Fig. 4. Adaptive Heuristics; Validation on real traces

using $f(t)$. This way we always know the size of the group with precision γ as being set by the pricing application.

Out of $\approx$ 200 different traces we found only 11 traces with considerable activity. All these traces were comprised out of $\approx$ 256 ISPs. We analyzed only the first $300,000$ seconds ($\approx$ 80 hours) of these traces. Figure 8(b) illustrates performance of *HRMA* variants for various values of γ. One thing to notice is that on the average Conscious Adaptive *HRMA* obtained exactly the same results as the Basic *HRMA*. This means that the distribution of values in the tree was always such that the values of the subtrees were far from the median of the intervals defined by their local thresholds. We do not have an explanation for this. Also, it would be difficult to come to a decisive conclusion without inspecting more traces. However, the traces we used represent a certain type of real life traffic, and *HRMA* performance on these traces is very satisfactory. The cost-gain factor of *HRMA* versus other known techniques represented by OB_h ranges from 5 to 10 on the average. As one would expect, the variance is high because the sample is too small. The variance is lower when threshold values are lower which generates more threshold events irrespective of the nature of the trace. It gets higher when thresholds grow. The variance diminishes as the thresholds grow even more since large thresholds preclude any local violation events.

9 Conclusions

We proposed a protocol-independent group size accounting mechanism. It allows explicitly to control the trade-offs between the accuracy of accounting and expected benefit from it, and between the cost-effectiveness of the multicast and unicast. As simulations show, the communication costs it incurs are low. On the average, it reduces management bandwidth by factor of 2 to 10 compared to other existing techniques while yielding acceptable precision.

Acknowledgments

We thank K. Almeroth, and R. Chalmers for sharing MBone traces with us.

References

1. J. Chuang and M. Sirbu, "Pricing Multicast Communication: a Cost Based Approach," in *INET'98*, Geneva, Switzerland, July 1998.
2. T. Henderson and S.N. Bhatti, "Protocol independent multicast pricing," in *10th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'00)*, The Univ of N. Carolina, USA, June 2000.
3. H. Holbrook and D. Cheriton, "Multicast channels: Express support for large-scale single-source applications," in *ACM SIGCOMM'99*, USA, Sept. 1999.
4. D. Breitgand, , D. Dolev, and D. Raz, "Hierarchical reactive monitoring of multicast membership size," Tech. Rep. TR2002-45, School of Engineering and Computer Science, Hebrew Univ. of Jerusalem, July 2002.
5. Mark Dilman and Danny Raz, "Efficient reactive monitoring," *IEEE Journal on Selected Areas in Communications (JSAC), special issue on recent advances in network management*, Apr. 2001.
6. K. Almeroth and M. Ammar, "Multicast group behavior in the internet's multicast backbone (mbone)," *IEEE Communications*, June 1997.
7. K. Sarac and K. Almeroth, "A Long-Term Analysis of Growth and Usage Patterns in the Multicast Backbone (MBone)," in *INFOCOM'00*, Israel, Mar. 2000.
8. Dan Rubenstein, Jim Kurose, and Don Towsley, "A study of proactive hybrid fec/arq and scalable feedback techniques for reliable rea-time multicast," *Computer Communications Journal*, vol. 24, no. 5–6, pp. 563–574, Mar. 2001.
9. C. Liu and J. Nonnenmacher, "Broadcast audience estimation," in *IEEE INFOCOM'00*, Tel-Aviv, Israel, Mar. 2000, vol. 2, pp. 952–960.
10. T. Friedman and D. Towsley, "Multicast session membership size estimation," in *IEEE INFOCOM'99*, New York City, NY, USA, Mar. 1999, vol. 2, pp. 965–972.
11. Sara Alouf, Eitan Altman, and Philippe Nain, "Optimal on-line estimation of the size of a dynamic multicast group," in *INFOCOM'02*, NY, USA, June 2002.
12. D. Dolev, O. Mokryn, and Y. Shavitt, "On Multicast Trees: Structure and Size Estimation," in *INFOCOM'03*, San-Francisco, USA, Apr. 2003.
13. Robert C. Chalmers and Kevin C. Almeroth, "Modelling the Branching Characteristcs and Efficiency Gains in Global Multicast Trees," in *INFOCOM 2001*, Anchorage, Alaska, USA, Apr. 2001.
14. Robert C. Chalmers and Kevin C. Almeroth, "Developing a Multicast Metric," in *GLOBECOM 2000*, San-Francisco, CA, USA, Dec. 2000.
15. D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei, "Protocol independent multicast sparse mode (PIM-SM): Protocol specification," June 1997, RFC2117.
16. S. Herzog, S. Shenker, and D. Estrin, "Sharing the cost of multicast trees: An axiomatic analysis," in *SIGCOMM'95*, Cambridge, USA, Aug. 1995, pp. 315–327.
17. B. Briscoe, "The direction of value flow in connection-less networks," in *NGC'99*, Pisa, Italy, Nov. 1999.