Video Multicasting with Variable Bandwidth Requirements

Michael Samidi[1] Michalis Faloutsos[1] Rajesh K. Pankaj[2] Kenneth C. Sevcik[3]

[1] Dept. of Computer Science, University of California, Riverside, CA 92521, USA
{msamidi, michalis}@cs.ucr.edu

[2] Qualcomm Inc., 6455 Lusk Blvd., San Diego, CA 92121, USA
rpankaj@qualcomm.com

[3] Dept. of Computer Science, University of Toronto, Ontario, M5S 3G4, Canada
kcs@cs.toronto.edu

Abstract

The increasing demand for point-to-multipoint connections has established multicast routing as a crucial aspect of network management. Hierarchical encoding is a newly proposed technique that offers a variety of quality levels in one encoding; a user or a switch can filter the data that will be processed further. In this paper, we examine how hierarchically encoding can increase the efficiency of multicast routing and propose polynomial algorithms for this purpose. Experimental work demonstrates the advantage of the proposed algorithms over previous solutions.

1 Introduction

Multicasting can be defined as the distribution of the same information from one to many nodes concurrently. Many emerging applications are of multicast nature, and thus multicasting has become a very active research area. The multicast problem with one source on undirected graphs becomes the Steiner tree problem. The problem is NP-complete and algorithms for the exact solution have computational complexity of at least $O(N3^M)$ [DW72], which raises the need for good approximations with polynomial complexity.

The hierarchical encoding technique enables us to use the resources more efficiently. This encoding method has received a lot of attention from the networking community [Gha89] [KV89] and is still an active research area. Hierarchical encoding is a layered way of encoding the kind of information that can appear in different quality levels such as audio and video data. In hierarchical encoding, the information of a session is divided into an enumerated set of **levels**, and higher levels can only be used on top of the lower levels. Therefore, a destination can receive a continuous sequence of such levels, including the first level. We refer to such a set of levels by the term **stream**. Note that each stream corresponds to a different quality of the data, with the lower stream being of lower quality.

Assuming a hierarchical encoding, we can offer a variety of streams within a single multicast session, and this constitutes the **heterogeneous multicast problem**. In contrast, we refer to the one-stream problem as the **homogeneous problem**. In the heterogeneous problem, we can distinguish two parameters: the cost of routing a session, which reflects the resource use, and the benefit of routing a session, which reflects the total user-satisfaction. Thus, a high level definition of the heterogeneous problem is as follows. Given a weighted directed graph, a session and a function that combines cost and benefit into a single value, we have to decide: (1) which stream to provide to each user, and (2) how that stream is to be routed to the user. Our goal is to maximize the objective function.

The motivation for this work is the absence of algorithms that include the stream selection in conjunction with the routing problem. On the one hand, previous work on routing considers one preferred stream per destination as we saw above. On the other hand, a receiver-driven stream-optimization method has been proposed by McCanne et al. [MJV96], but that work does not address the routing problem. Furthermore, this approach does not seem to apply to large networks, as shown by measurements on the MBone by Handley [Han97].

In this paper, we propose algorithms to exploit the hierarchical encoding scheme in routing multicast connections. We show that there is a benefit from stream-optimization algorithms for video multicast sessions and propose an efficient algorithm for doing so. First, we show that the proposed algorithm achieves its goal: it is sensitive to the cost-benefit interplay. Thus we offer the highest streams for low routing-cost or high benefit cases, while we reduce the offered streams as the routing cost increases or the benefit decreases.

The rest of this paper is structured as follows. In Section 2, we present our model and lists previous work. In Section 2.1, we specify our QoS requirements for the heterogeneous problem.

In Section 3, we describe our algorithms for the heterogeneous problem. In Section 4, we present our simulation results. Finally, we conclude and identify future work in Section 5.

2 Model and Previous Work

We describe briefly the most common model for multicast problems. Given an undirected weighted graph $G(V, E)$ and a subset of **participant** nodes $S \subseteq V$, we want to find the minimum weight tree that spans the participants. The number of nodes $|V|$ is N and the number of participants $|S|$ is M. For the homogeneous case, the routing cost on an edge is its weight, in other words the cost is the distance. The cost of a path is the sum of the routing cost of its edges. In this paper, when we refer to the path between two nodes, we imply the minimum cost or shortest path. Along these lines, the term "nearest" refers to the minimum cost path.

Let us recall some of heuristics for the Steiner problem. First, the union of the shortest paths between source and each destination forms a solution with a WCI of $O(M)$. We call this algorithm **Shortest Paths**. Note that Shortest Paths routes every destination separately independently of the other destinations. Second, an intuitive greedy algorithm for the multicast problem was suggested by Takahashi and Matsuyama [TM80]. We start from the root and construct the multicast tree by adding one participant at a time. At each step we connect the participant that is nearest to some node of the partial multicast tree. We call this algorithm **Nearest Participant First (NPF)**. Several other heuristics exist in the literature [TM80] [KMB81] [RSC86] [Meh88]. A comprehensive survey of the algorithms is provided by Winter [Win87]. A common measure of how good such an approximation is the **worst-case inefficiency (WCI)** of a multicast algorithm which is defined as the maximum ratio of the cost of the tree of the algorithm over the optimal one.

The heterogeneous problem has received little attention until recently. Most of the work assumes that the participants request only one stream, and thus, there is no stream optimization component. The first work was conducted by Shacham [Sha92], and the proposed algorithm, which we call **fixed-stream Shortest Paths het SP**, is an extension of the Shortest Paths algorithm. Its WCI is $O(M)$, and the complexity of the algorithm is $O(N^2)$. Maxemchuck [Max97] and Faloutsos et al. [FPS97a] proposed the same greedy algorithm independently. The algorithm groups the participants by the required stream, and routes the highest stream group using NPF. The procedure is repeated until all groups are routed. We call this algorithm **Maximum Stream Participant First (MaxStrPF)**. It has been proven that, when the cost of an edge is additive to the flow, the worst case inefficiency of this algorithm is bounded by $O(\log M)$ [FPS97a]. Furthermore, Faloutsos et al. [FBP98] proposed QoS routing protocol called QoSMIC. The protocol implements a greedy type of routing algorithm in a distributed fashion. Recently, Wong et al. [WKM00] proposed a generic clustering framework to deal with the heterogeneity by grouping approximately similar sources and receivers together with similar QoS requirements.

2.1 The General Problem

Each destination defines a range of preferred streams and participates in the session with one of them, while streams below the preferred range are called **failing streams**. Given the preferred ranges, the network manager has to decide which stream to offer to each destination. In other words, in the heterogeneous problem, the stream selection problem is added to the routing problem. Note that the two problems are correlated and can not be solved independently.

First, we have to redefine the cost of routing a session. The routing cost of an edge is defined as a function of the routing requirements of the connection. For the rest of this paper, we will use the concept of flow (in terms of bandwidth) to represent routing requirements[1]. We define the **cost function** $cost(e, f)$, which represents the cost of passing flow f on edge e. We define the cost of an edge e on which we want to route a connection of bandwidth b as: $cost_e(b) = v \cdot b$, where v is the weight of the directed edge. We define the cost of a path/session as the sum of all the cost increases of all the edges. Given a partial tree, the cost of connecting a destination with a specific stream is called **joining cost**, and is a function of the stream. Note that the actual mathematical function can be chosen to reflect the nature of the network.

Second, we have to account for the goodness that a user experiences by receiving a higher quality stream. For this, we introduce the concept of **happiness** that measures the user-satisfaction that corresponds to a certain stream. Alternatively, user-satisfaction can be interpreted as a price

[1]Clearly the use of one parameter to express these requirements is a simplification, but the task of relating resources with utilization is a significant issue by itself and extends beyond the scope of this paper.

that the user is willing to pay for a certain stream. For each destination d, a happiness value $hap_d(s)$ is associated with each stream s, and these values are non-decreasing with stream. Furthermore, for different destinations the happiness values of a stream may not be the same. For uniformity, we will assume that failing streams have a happiness value of minus infinity, and thus, every destination-stream pair has a happiness value. Naturally, the total happiness of a session is the sum of the happiness values of all destinations.

Therefore, the heterogeneous problem can be defined as follows. The input is an undirected graph, a source, a set of destinations with happiness values for every stream, and an objective function $Gain()$. We want to find the tree and the stream assignment for which happiness and cost maximize the objective function. The $Gain(H, C)$ function that combines the two parameters into a single objective value. By definition, the function measures the efficiency of the solution, therefore, large happiness and low cost correspond to large values of the function.

In our formulation of the problem, we make two major assumptions. First, we assume that all the levels of a session have to use the same route. If this is not necessary, we can multicast each level independently only to the appropriate destinations. However, this independence requires synchronization of the levels coming from different routes, which is not trivial. We also make the common assumption that our network is an undirected graph. Our algorithms can be extended to the directed case in a straightforward way, but their performance can differ greatly. On asymmetric graphs, even for the homogeneous problem the greedy algorithms can perform as bad as multiple point-to-point connections in the worst case [FPS96].

3 Our Heterogeneous Algorithms

Let us focus on the stream assignment problem, and consider only one destination. Trivially, the route is the shortest path between the source and the destination. All we have to do is choose one of the preferred streams. We choose the stream that maximizes the objective function, and call it **optimized stream**. More rigorously, the optimized stream is given by $\max_s [Gain(hap_d(s), c)]$ where c is the joining cost for d with stream s. We will incorporate the above procedure to do the stream selection step.

Our first algorithm for the heterogeneous problem is based on the observation that lower stream destinations can tap into the higher streams and acquire the levels they want. Therefore, we route the higher stream destinations first. Namely, at each selection step, we find the optimized stream for each unreached destination. Then, among destinations of maximum optimized stream, we choose the one nearest to the partial multicast tree. We call this algorithm the **Optimized Stream Maximum Participant First** or **OptMaxStrPF** and we present it below.

1. START from the source.
2. REPEAT until all destinations are reached
   (a) SELECT stream: For each destination d, find the stream $s_{o,d}$ that maximizes the objective function

   $$s_{o,d} : \max_{s} \left[Gain(hap_d(s), c) \right]$$

   where c is the joining cost for d with stream s.

   (b) SELECT destination: Find the set of unreached destinations that have maximum optimized stream. From this set, select the destination, d, that is nearest to the partial multicast tree.

   (c) JOIN: Include d and the related path in the tree with $s_{o,d}$.

In another approach, we observe that, at each step, we have to select a destination-stream pair. Therefore, we choose the pair whose added happiness and cost maximize the objective function. We call this algorithm **Optimized Participant First** or **OPF**.

1. START from the source.
2. REPEAT until all destinations are reached
   (a) SELECT destination-stream: Find the (destination, preferred stream) pair whose added happiness and cost maximize the objective function:

   $$(d_o, s_o) : \max_{(d,s)} \left[Gain(hap_d(s), c) \right]$$

   where c is the cost of joining d to the tree with stream s.

(b) JOIN: Include d_o and the related path in the tree offering stream s_o.

In an efficient implementation of the above algorithms [FPS97b], the complexity is polynomial and bounded by $O(M\ N^2 + M\ N\ L)$, where L is the number of distinct streams. If we restrict the maximum number of streams L to $O(N)$, the complexity of the algorithms becomes $O(M\ N^2)$.

In conclusion, we can see that the two algorithms conform with the cost sensitivity, flexibility and low complexity requirements of the introduction. They also satisfy the stream rewarding requirement, since they encourage the distribution of high streams with the use of the happiness concept. To the best of our knowledge, these are the first algorithms that address the general heterogeneous problem.

4 Experimental Results

We implemented a simulator in C++ to examine the performance of the above algorithms in practice. We used the routing data provided by the National Laboratory for Applied Network Research [NLA00]. The data represents the connectivity of autonomous systems collected on Jan 02, 2000 and it consists of 6474 nodes and 12572 edges. We generated a smaller graph from the data by reduction technique as described in [SF00]. The interesting point of the technique is the ability to maintain the characteristics of Internet topology such as power-laws as discussed in [FFF99]. The final graph has 439 nodes with 313 edges. The participants are chosen randomly, and so is the root of the session among them. We assume that the information is divided into five distinct levels. All preferred ranges include the lowest stream, while the higher preferred stream is randomly chosen. The happiness values are increasing linearly with the stream index for all destinations. We assume the ratio of consecutive streams θ to be constant for every pair of consecutive streams in a session. For our experiments, we set $\theta = 2$. This way, the required bandwidth b of stream s is an exponential function of the stream index, $b = \theta^s$. We select the proportional cost function, $cost_e(b) = v \cdot b$, where v is the weight of the directed edge as described in Section 2.1.

Any function can be used as an objective function. For the sake of the experiments, we chose the following function that takes as parameters the happiness Hap and cost $Cost$: $Gain(Hap, Cost) =$

$\frac{Hap}{Cost}$. In other words, we consider the network resources as our capital and the satisfaction as our revenue, then for every session we want to increase the "rate of return" on our capital.

The algorithms we compare with are the following:

1. *NPF low:* We perform NPF and offer the minimum stream of the session to all destinations. This algorithm addresses the **homogeneous problem** and offers a very conservative solution, but we use it as a point of reference.

2. *NPF hi:* We perform NPF and, to every destination, we offer its maximum preferred stream.

3. *het SP:* We perform the Shortest Paths algorithm and offer the maximum preferred stream to destinations. Recall that het SP corresponds to the algorithm suggested by Shacham [Sha92], which seems to be the only heterogeneous algorithm proposed so far.

4. *OPF:* We perform Optimized Stream Optimized Participant First as described in the previous section.

5. *OptMaxStrPF:* We perform Optimized Stream Maximum Stream Participant First as described in the previous section.

We will call het SP and NPF hi *best-effort* algorithms. We will also call OPF and OptMaxStrPF *stream-optimizing* algorithms.

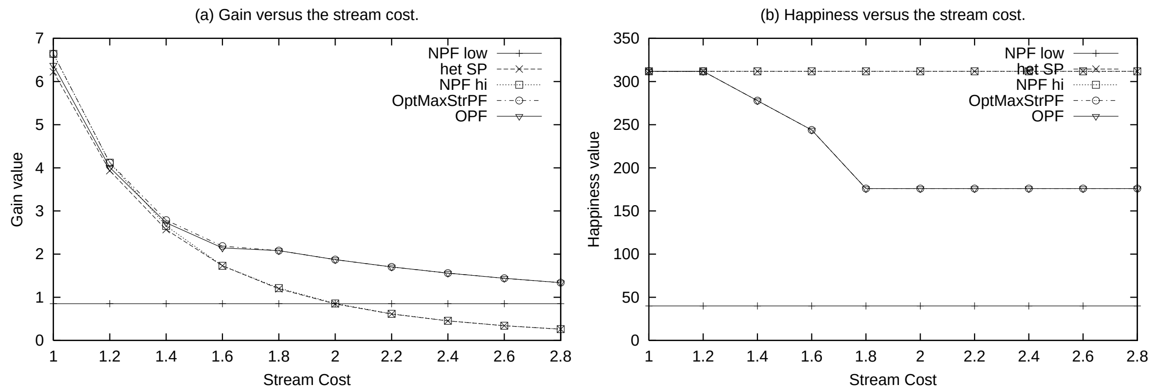

Figure 1: The Gain and Happiness values versus stream ratio with group-size 5.

The Gain and happiness of heterogeneous solutions. We show how a heterogeneous solution maximizes our gain by adapting to different stream cost values. In Figure 1(a), we plot the *Gain* values versus the stream cost parameter for group-size 5. On the one hand, we see that as stream cost increases, the *best-effort* algorithms can lead to inefficient solutions, since they are insensitive to the stream cost. Namely, their return of accommodating a session is zero, which means that we do not have a profit. Note that after a stream cost of 2, both algorithms fall below NPF low. On the other hand, the *stream-optimizing* algorithms take advantage of low stream cost values by offering higher streams and thus perform better than NPF low. Note that OptMaxStrPF and OPF gradually converges into one value eventhough OptMaxStrPF outperforms OPF before a stream cost of 1.8 by as much as $2-4\%$. This is because OptMaxStrPF offers less cost than OPF. In Figure 1(b), we plot the average total happiness versus the stream cost parameter. We see that the stream-selecting algortihms adapt to to the increase of the cost of higher streams by offering lower streams, which corresponds to lower total happiness.

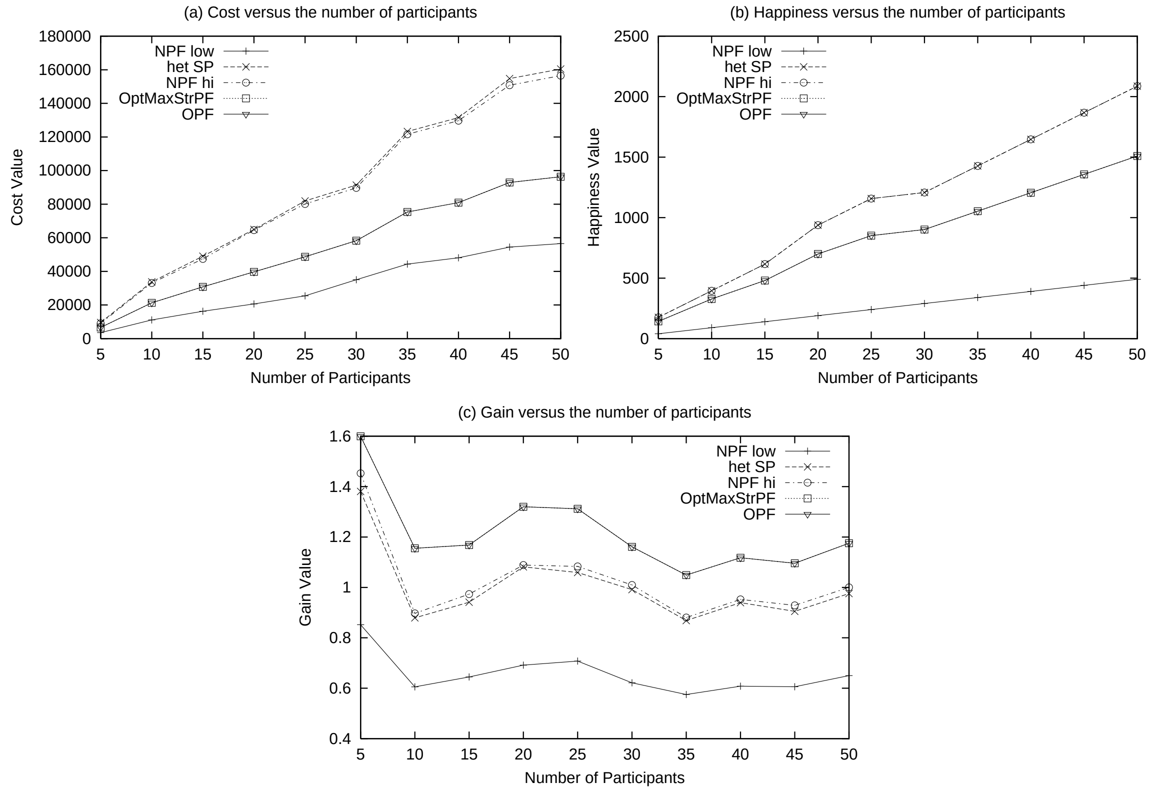

Figure 2: The general heterogeneous solution for a stream cost 2.

The general heterogeneous problem. Figure 2(c) shows that the *Gain* values versus the number of participants for a value of stream cost 2. We see again that the *stream-optimizing* algorithms outperform the *best-effort* ones in terms of the *Gain* values by as much as 20%. As the size of the multicast group increases the *stream-optimizing* algorithms offer lower cost and lower happiness than the *best-effort* ones as we can see from Figure 2(a) and Figure 2(b). Note that het SP also performs some stream optimization. However, our *stream-optimizing* algorithms manage to optimally maximize the *Gain* by offering slightly less happiness in order to reduce the cost drastically. The *Gain* comes from the improved routing less cost. Greedy approach is known to provide less expensive trees.

5 Conclusions

In this paper, we deal with the problem of merging the hierarchical encoding and the multicast routing. This leads to the formulation of the heterogeneous multicasting which is a generalization of the common multicast problem. We define efficiency by a set of requirements. This paper proposes efficient routing algorithms for the problem.

We formulate and tackle the general heterogeneous problem. Apart from the routing problem, we have to solve a stream selection problem that attempts to maximize the user's happiness. We have to strike the balance in the trade-off between the happiness and the cost of a session. The experimental study here is a proof of concept that the proposed algorithms can provide an advantage.

In conclusion, network management can benefit from the use of hierarchical encoding in routing. We believe that our simple and efficient algorithms are a step in this direction. We will do more elaborate simulations customized to the specifications of typical applications such as video-conferencing and video distribution.

FUTURE WORK. We have started to examine how we can incorporate our *stream-optimizing* algorithms into Internet protocol. The QoSMIC is the natural choice since its routing implements the greedy algorithms and is flexible since it is receiver-driven protocol.

ACKNOWLEDGMENTS. The authors would like to thank Manolis Skordalakis and Christos Faloutsos. We also thank the anonymous reviewers for their constructive comments.

References

[DW72] S. E. Dreyfus and R. A. Wagner. The Steiner problem in graphs. *Networks*, 1:195–207, 1972.

[FBP98] M. Faloutsos, A. Banerjea, and R. Pankaj. QoSMIC: Quality of Service Multicast Internet protoCol. *ACM SIGCOMM*, pages 144–153, Aug 31-Sept 4, Vancouver BC 1998.

[FFF99] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the Internet topology. *ACM SIGCOMM*, pages 251–262, Sep 1-3, Cambridge MA, 1999.

[FPS96] M. Faloutsos, R. Pankaj, and K. C. Sevcik. Multicasting in directed graphs. *18th Biennial Symposium on Communications, June 2-5*, pages 29–32, 1996.

[FPS97a] M. Faloutsos, R. Pankaj, and K. C. Sevcik. Multicast routing with heterogeneous quality. *The Fourth IEEE Workshop on the Architecture and Implementation of High Performance Communication Systems (HPCS'97), June*, 1997.

[FPS97b] M. Faloutsos, R. Pankaj, and K. C. Sevcik. Video multicasting with hierarchical encoding: The fixed and the general problem. *University of Toronto, Technical Report CSRI-367*, 1997.

[Gha89] M. Ghanbari. Two-layer coding of video signals for VBR networks. *IEEE Journal on Selected Areas on Communications*, 7(5):771–781, 1989.

[Han97] M. Handley. An examination of the MBone performance. Unpublished manuscript, 1997.

[KMB81] L. Kou, G. Markowsky, and L. Berman. A fast algorithm for Steiner trees. *Acta Informatica, Springer-Verlag*, 15:141–145, 1981.

[KV89] M. Karlsson and M. Vetterli. Packet video and its integration into the network architecture. *IEEE Journal on Selected Areas on Communications*, 7(5):739–751, 1989.

[Max97] N. F. Maxemchuck. Video distribution on multicast networks. *IEEE Journal of Selected Areas in Communications*, 15(13):357–372, April 1997.

[Meh88] K. Mehlhorn. A faster approximation algorithm for the Steiner problem in graphs. *Information Processing Letters*, pages 125–128, 1988.

[MJV96] S. McCanne, V. Jacobson, and M. Vetterli. Receiver-driven layered multicast. *ACM SIGCOMM*, pages 117–130, August 1996.

[NLA00] NLANR. National Laboratory for Applied Network Research. Raw Routing Table Information, http://moat.nlanr.net/Routing/rawdata/, Jan 02, 2000.

[RSC86] V. J. Rayward-Smith and A. Clare. On finding Steiner vertices. *Networks*, 16:283–294, 1986.

[SF00] J. Sun and M. Faloutsos. Small realistic graphs for simulations. Technical Report In progress, University of California Riverside, Computer Science, 2000.

[Sha92] N. Shacham. Multipoint communication by hierarchically encoded data. *Proc. IEEE INFOCOM*, pages 2107–2114, 1992.

[TM80] H. Takahashi and A. Matsuyama. An approximate solution for the Steiner problem in graphs. *Math. Japonica*, 24(6):573–577, 1980.

[Win87] P. Winter. Steiner problem in networks: a survey. *Networks*, 17:129–167, 1987.

[WKM00] T. Wong, R. Katz, and S. McCanne. An evaluation of preference clustering in large-scale multicast applications. *Proc. IEEE INFOCOM, March 26-30*, 2000.