LVMSR–An Efficient Algorithm to Multicast Layered Video

Wushao Wen[1+], Biswanath Mukherjee[2], Dipak Ghosal[2], and S.-H. Gary Chan[3]
[1]Dept. of Elect. & Comp. Engineering, University of California, Davis, CA 95616, U.S.A.
[2]Dept. of Computer Science, University of California, Davis, CA 95616, U.S.A.
[3]Dept. of Computer Science, Hong Kong University of Science and Technology, Hong Kong
email:wswen@ece.ucdavis.edu

Abstract: **Layered video is a video compression technique to encode video data in multiple layers. It typically consists of a base layer and additional layers that provide enhanced video quality. The multicasting operation of a layered video may need to satisfy: *i)* bounded end-to-end delay from a source to each receiver, *ii)* minimum total cost, and *iii)* minimum delay jitter between the various video streams received by the receivers. Because different nodes may request different video quality and because of limited bandwidth on the network's links, different layers of video data may reach their destinations over different distribution trees, and not all receivers may receive all of their requested layers. The problem of computing such data distribution paths is NP-complete, which means that no optimal solution method is available. This paper presents a new heuristic algorithm called LVMSR with $O(Rn^2)$ time complexity and $O(R^2)$ message complexity, where n is the number of nodes in the network and R is thereceiver group size. Our simulation results show that the multicast data paths computed by our algorithm can always satisfy the delay constraint with reasonably small total cost.**

I. Introduction

In video multicast, a single source transmits a video stream to multiple receivers over a network. However, the network's heterogeneity and scale make the multicast communication a difficult problem. Layered-video is the solution to satisfy user heterogeneity. In layered-video encoding/decoding [6], [7], [8], a signal is encoded into a number of layers that can be sequentially combined to provide progressive refinement of received video signals. In case of insufficient network bandwidth, some data from higher layers may be dropped or are not transmitted. Users can still receive the data of the lower layers, which provide reduced quality but continuous video.

An efficient multicast-tree-construction algorithm is critical to make the layered-video multicast scheme successful. This algorithm should be able to minimize the total cost of the multicast data path, transmit the data to all destinations in bounded delay, and minimize the delay jitter in receivers. We refer to this problem as the layered-data, bounded-delay, minimum-cost problem. As we already know, the least-cost multicast tree is called a Steiner-tree [9] and the problem of finding a Steiner-tree is known to be NP-complete [10]. It is clear that the construction of a minimum-cost layered-data-transmission super-tree with delay constraint is also NP-complete. In this investigation, we present a new heuristic algorithm, *layered-video multicast super-tree routing (LVMSR)*, for multicasting layered-video. The proposed algorithm is a distributed algorithm that always satisfies the delay and bandwidth constraints. Data distribution via our constructed multicast data path has sub-optimal total cost and delay jitter.

This paper is organized as follows. Section II describes our system model and defines the problem. Some keywords are also defined in this section. Section III describes our heuristic algorithm Section IV presents our simulation model and some illustrative numerical results. Section V gives some concluding remarks.

II. Layered-video Representation and Distribution

Let us denote the number of layers generated by the layered-video encoder as L. Layer 1 is the base layer. Any lower l ($l \leq L$) layers combined together can be used to represent the original video with a different quality. The $(l+1)$th ($l < L$) layer provides enhancement of the video that is represented by the lower l layers. The $(l+1)$th layer is valid only if all the lower l layers is valid. If any data of layer j ($j \leq l$) is missed, then the $(l+1)$th layer is also invalid for decoding. In this study, we make no assumption on how the original video is compressed and encoded; we only require that the video be represented in increasingly refined layers. In practice, the base layer provides basic data for those users with minimum available bandwidth, e.g. ISDN users. Every additional layer combined with the lower layers can meet one class of user's requirement.

A. Network Model

We model the network by a connected graph $G(V, E)$. V, the node set, represents routers or switches in the network and E, the data-link set, represents communication links between nodes. There are three parameters corresponding to each communication link e, $e \in E$.

- $d(e)$ is the delay for link e. In this study, we assume that every data-link's delay is a known constant.
- $c(e)$ is the cost rate of link e. It represents the charge incurred for using the data-link and can be measured

+Correspondence author.

in terms of dollars per megabit ($\$/Mb$), or dollars per second ($\$/s$).
- $b(e)$ is the available bandwidth for link e, which can have a different value at a different time.

In this study, we assume that there is a single multicast source. Receivers may have different capacities, so they may request different layers of video data. We also assume that the source maintains separate sessions for different layers.

B. Basic Definitions

To clarify the discussion, we need to define the following keywords.

- A *super-tree* $ST(V, E)$ is a graph constructed by merging several trees that have the same root. The trees are merged to form the super-tree according to the following rules:
  1. The super-tree node set V is the union of the node sets of all trees. This can be represented by: $V = V_1 \cup V_2 \cup \cdots \cup V_n$, where V_i is the node set of tree i.
  2. The super-tree edge set is the union of the edge sets of all trees, which can be represent by: $E = E_1 \cup E_2 \cup \cdots \cup E_n$, where E_i is the edge set of tree i. In this definition, we assume that, in the resulting super-tree, edges that have the same end nodes are merged into one edge.
- A *super-ditree* $SDT(V, E)$ is a graph constructed in the same way as a super-tree except that all of the original trees used to construct the super-tree are directed trees and the edges that have the same end nodes are merged into one edge only if their directions are the same.
- A *weighted super-tree* $WST(V, E)$ is a graph constructed in the same way as a super-tree except that all of the original trees are weighted trees and the resulting edge weight is the sum of all the weights of the edges with the same end nodes when merging the corresponding edges.
- A *weighted super-ditree* $WSDT(V, E)$ is a graph that is constructed from a group of weighted directed trees according to both policies of constructing super-ditree and weighted super-tree.
- A *constrained lowest-cost path* from tree T to node v is the least cost path from any node in tree T to node v that has delay less than the maximum acceptable delay Δ from the root s of the tree T to node v. We denote the corresponding path as $P(T, v)$, the cost on such a path as $P_C(T, v)$, and the delay on it as $P_D(T, v)$.

C. Problem Specifications

For a layered-video multicast, it is important to design an efficient multicast super-tree. The multicast super-tree is not exactly a tree because different layers of video data can go through different paths to a destination. When we look at a specific layer of video data routing, the routing paths form a tree. However, the routing paths of a set of streams (a layer of video data forms one stream) forms a super-tree. A multicast super-tree must satisfy the following requirements:

- Priority transmission: Network gives higher priority to lower layers and always transmits data of a lower layer before transmitting any data of a higher layer. Furthermore, a receiver can decide how many layers of data it will receive according to its own capacity or the bandwidth available for its connection. When the available bandwidth is low, a higher-layer stream may not be delivered to some receivers even though they requested for it. We also assume that the network has some mechanisms (such as RSVP) to reserve network resources for data transmission.
- Minimum total cost: The total cost of the multicast super-tree is calculated as follows:

$$\textit{Routing Cost}: \ C = \sum_{e \in ST} c(e) \ ; \tag{1}$$

  where ST is the multicast super-tree.
- Bounded delay: The delay of a path $P(s, r)$ from source s to a receiver r ($r \in R$, R is receiver set) should always satisfy a bounded value Δ, i.e.,

$$\forall r \in R, \ \sum_{e \in P(s,r)} d(e) \leq \Delta \ . \tag{2}$$

- Minimum delay jitter: We define the delay jitter as the sum of different receiver's delay variance, which can be calculated as follows:

$$\sigma_D^2 = \sum_{r=1}^{R} \sum_{j=1}^{L_r} (D_{rj} - \bar{D}_r)^2 \frac{B_j^2}{B_r^2}, \bar{D}_r = \sum_{l=1}^{L_r} D_{rl} \frac{B_l}{B_r} \ ; \tag{3}$$

  where D_{rj} is the delay for layer j at receiver r, and $\bar{D}_r$ is the average data delay at receiver r, B_r is the total bandwidth required by receiver r, and B_l is the bandwidth requirement for layer-l
- User satisfaction: User satisfaction is measured by the fraction of the weight of the requested video layers by users and the weight of the received video layers. It is given by:

$$S = \frac{\sum_{r=1}^{R} \sum_{l=1}^{L_r} B_r}{\sum_{r=1}^{R} \sum_{l=1}^{M_r} B_r} \ ; \tag{4}$$

  where M_r is the total number of layers request by receiver r and L_r is the actual continuous lower layers received by receiver r.

III. Layered-video Multicast Super-tree Routing (LVMSR) Algorithm

Given a network $G(V, E)$, the LVMSR algorithm is used to compute a multicast super-tree, which can deliver as many layers of video data as possible to satisfy every receiver's requirement. The data is distributed with minimum cost while simultaneously satisfying the quality of service (QoS) requirement, i.e., bounded delay and small delay jitter.

The LVMSR problem is an optimal routing-with-delay-constraint problem. The cost requirement often conflicts with the delay requirement. As a result, the super-tree with minimum cost may have long delays for some paths, whereas a minimum-delay super-tree may incur a high cost.

A trade off must be reached to satisfy the specific requirements. Our algorithm is a heuristic that adopt some concepts of the Steiner-tree algorithm. While the Steiner-tree algorithm is used only to compute a multicast tree, our algorithm constructs a multicast super-tree that delivers more than one layer of data to the destinations.

In the following discussion, we assume that all nodes in the network have the information necessary to construct the constrained lowest-cost path from it to any other node. LVMSR is a source-oriented distributed routing algorithm. The basic idea of the LVMSR algorithm is that it first constructs the minimum-cost multicast tree under delay constraint for the base layer (layer 1), denoted as T_1. In layered-video multicast, all receivers which request for the same video in a multicast session should subscribe to the base layer, hence, this multicast tree includes the source node and all receivers. Computing such a tree is the standard conditional Steiner-tree problem and there are heuristics to solve this problem [3], [3]. After the layer 1 multicast tree is constructed, we then construct layer 2. And then layer 3, ... and finally the layer L multicast tree. The key steps of the heuristic algorithm are described below.

A. The LVMSR Algorithm

- *Step 1*: Initialize the multicast super-tree ST that only includes the source node S. Set the current layer $l = 1$.
- *Step 2*: Source marks the tree rooted at source node of ST that has enough bandwidth to transmit layer-l data. If there is more than one path from the source to any node, the lowest-cost path from the source to the node is selected. This is the initial multicast tree T_l for layer-l. For every node in the initial layer-l multicast tree, record its corresponding delay. The source then sends the initial tree information to all receivers that are not in the tree.
- *Step 3*: Every layer-l receiver that is not in the tree T_l calculates its delay-constrained, lowest-cost path from tree T_l to itself separately and sends this information to the source. The delay-constrained, lowest-cost path is calculated in a simple way. A receiver calculates the lowest-cost path and minimum delay path from every node in tree T_l to itself. Then every receiver selects the lowest-cost path under delay constraint from all the paths calculated by the above two methods. If no constrained lowest-cost path exists, the receiver will notify the source and unsubscribe to the video session.
- *Step 4*: The source selects the layer-l receiver that has the minimum cost under delay constraint to the tree T_l, and it adds the receiver and its corresponding constrained lowest-cost path to T_l. If any node in the selected constrained lowest-cost path is already in tree T_l, the node's parent is changed to the new parent in the newly-selected, delay-constrained lowest-cost path. The delay information for every node in the tree is then updated.
- *Step 5*: If all the receivers for layer-l data are not in the tree, go back to *Step 3*. Otherwise, T_l is constructed successfully. Prune all the edges and nodes that are not in any layer-l receivers' data path and the resulting tree is the multicast tree T_l for layer-l.
- *Step 6*: Merge T_l with old ST, the result is the new super-tree ST.
- *Step 7*: If $l < L$, then $l = l + 1$ and go to step 2; else ST is the required multicast super-tree. Stop.

B. Dynamic Member Management in LVMSR

The LVMSR algorithm is also suitable for dynamic member management in that it is easy to add or delete receivers at any time. To add a receiver, the new receiver requests the source to send the current multicast super-tree information. The receiver then calculates the delay-constrained, lowest-cost paths for every layer from the super-tree to itself and sends the information to the source. Then, the source includes the newly-selected data paths and receiver in the super-tree. Deleting a receiver from the multicast session can be done by excluding all the paths that are used by the receiver exclusively.

C. LVMSR Properties and Complexity Analysis

The LVMSR algorithm has following properties:

- *Property 1:* A node in a newly-selected data path is originally in the tree if and only if the delay from the source to it along the newly-selected data path is less than the delay along the old data path, and using the node as a fork node for the selected receiver violates the delay constraint [1].
- *Property 2:* The multicast tree of any layer is a tree. There is only one data path, if exists, from the source to any receiver.
- *Property 3:* The multicast super-tree, if it exists, can always satisfy the delay constraint.
- *Property 4:* The multicast super-tree constructed by the heuristic is a weighted super-ditree.
- *Property 5:* The LVMSR algorithm is a distributed algorithm, the time complexity of this algorithm is $O(Rn^2)$ and the messages needed is $O(R^2)$. Here R is the number of receivers, n is the total number of nodes in the network.

IV. Illustrative Examples

In the following simulation, we generate a random network topology with 400 nodes and randomly select $N (N \leq 100)$ receivers. The network generation is based on the hierarchical-nodes and random-links idea [5]. A network plan is first divided in 10x10 grids. 20% of the grids are randomly selected as high-density areas and the other 80% are selected as lower density areas. The network plan is divided into 4 equal-sized wide areas and a wide-area center router is placed in the middle of every area. The wide-area center routers are fully connected, and they represent the backbone network. For every dense area (one grid) or every four sparse grid (2x2), an area center router is placed in its center. These are second-layer routers. A data-link is placed between any area center router and its wide-area center area router. Also, any two area centers routers may be connected. A link between two area centers is added by using the following probability p_l function [2], [5]:

$$p_l = \lambda exp(-d(u,v)/\rho D); \tag{5}$$

[1] A fork node is a connection node that is the beginning node for a newly-selected receiver's data path on the original multicast tree.

Where $d(u, v)$ is the distance between u and v, while D is the maximum distance between any two nodes that may be connected by a link. λ is used to control the data-link densities and a high λ results in high link densities. Finally, ρ is used to control the density of short data-links relative to long data-links and small ρ results in more short data-links. We set $\lambda = 10.0, \rho = 0.1$, and $D = 25\sqrt{2}$, which is the longest distance from a node to its wide-area center router. After placing all layer-2 nodes and the corresponding links, layer-3 routers and links are generated. A layer-3 router is placed randomly in a grid according to the placement probability. When a router is placed into a grid, it randomly selects a position that is not occupied by any router inside the grid. After all nodes are placed in the plane, a data-link is added between a layer-3 router and its area center router (layer-2 router). A link between layer-3 routers may also exist. The probability of a link existing between any two layer-3 nodes follows the same probability function p_l given in Equation 5, but with parameters $\lambda = 0.7, \rho = 0.2$, and $D = 20$.

The network topology generated here has 400 routers and 1182 links. The average degree in the network is about 3, which is a good approximation for network in the real world. Fig. 1 shows a network topology that is generated by our simulation. In this network, no duplicate links exist between any two nodes.

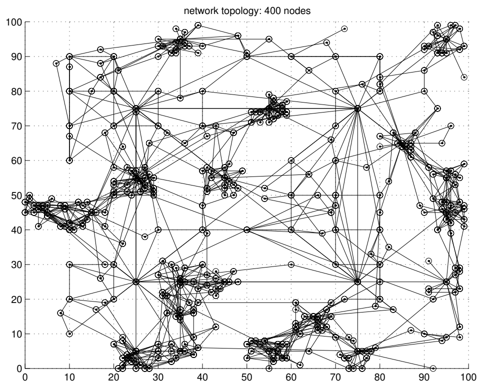

Fig. 1. Network topology in our simulation.

The cost of a data-link between any two routers is a random number $c(e)$, where $c(e) \in U(50, 70)$ for two layer-1 nodes (wide-area center node), $c(e) \in U(10, 30)$ for two layer-2 nodes (area center routers) or between a layer-1 node and a layer-2 node, and $c(e) \in U(1, 11)$ for any two layer-3 routers or a layer-3 and a layer-2 router. The delay of any data-link is assumed to be the length of the data-link. The average cost and delay for all data-links in the graph is 7.2453 and 7.5540, respectively. We assume that the total bandwidth of a data-link is 100 Mb/s between any two wide-area centers, and 10 Mb/s for all other data-links. At any time, a link's available bandwidth is a random number less than its total bandwidth. The available bandwidth for a data-link in different directions can be different.

Assume that the video data is encoded in two layers. The bandwidth requirement for each layer is 1 Mb/s. Assume also that the system has some mechanism to reserve resources when setting up data distribution paths and the available bandwidth will not change while setting up the data paths. The LVMSR algorithm successfully constructed a multicast super-tree. Fig. 2 shows the layer-1 multicast tree. There are 50 receivers in this tree. The delay constraint is $\Delta = 110$ in this simulation.

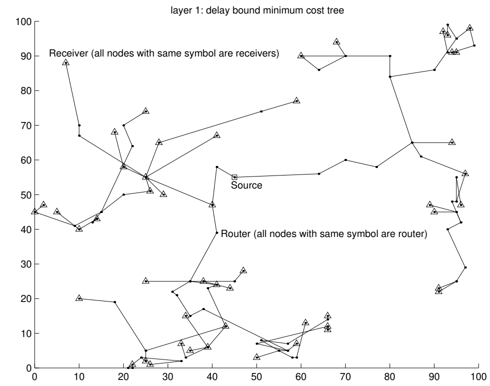

Fig. 2. Multicast tree for layer-1 data.

Fig. 3 shows the layer-2 multicast tree. For all the 50 receivers, half of them are randomly selected as receivers requesting data for both layers. Figs. 2 and 3 show that most of the nodes and links in Fig. 3 are the nodes and links in Fig. 2. The multicast trees have some long links because those links have smaller cost than the unselected ones. We also want to point out that some links overlap because more than 2 nodes may be in a line in graph. Thus, some nodes seem connected together even though they are not. It is already proved in III-C that the multicast path for every layer is a tree. All leaves in a layer's tree are receivers. But a receiver is not necessarily a leaf in the multicast tree. It could be a routing node and a receiver at the same time.

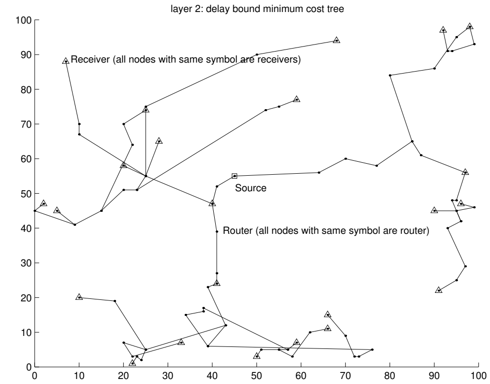

Fig. 3. Multicast tree for layer 2 data.

The multicast paths to deliver all of the data form a multicast super-tree, as shown in Fig. 4. It clearly shows that it is no longer a tree structure. Some receivers may have different data paths for different layers of data.

Using the above network model, we studied the relationship between the total cost and the delay bound Δ. In this study, we randomly select 50 nodes and half of them request both layers. The bandwidth requirements of both layers are 1 Mbps. Fig. 5 plots Δ vs. cost and it shows that, when the delay constraint is tight, we need more cost to deliver the data. The total cost reduces fast and then be-

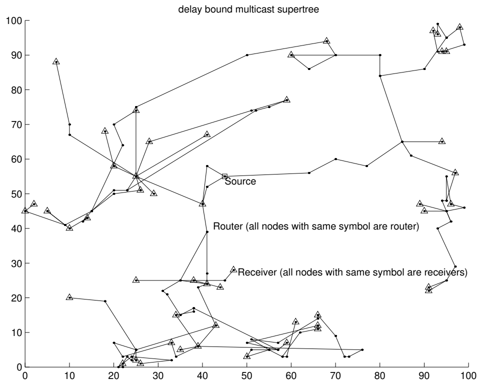

Fig. 4. Multicast super-tree for both layers.

comes relatively flat when the delay bound increases. The total cost changes little when the delay bound is large, because now, it is no longer an actual constraining factor in constructing the multicasting tree, i.e., the resulting multicast tree for every layer is equivalent to the minimum-cost multicast tree.

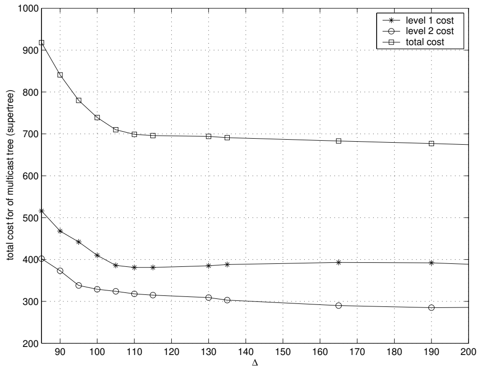

Fig. 5. Total cost vs. delay bound Δ.

We have also studied the average cost per layer per receiver for our algorithm. In this simulation, about half of the receivers in every test requests both layers. The group size is changed from 10 to 100. Fig 6 plots the average cost rate per layer per receiver vs. group size. The three lines correspond to average cost ($\$/Mb$) per receiver for layer 1, layer 2, and both layers, respectively. Fig 6 shows that, when the group size is small, the average cost is high. This is because every receiver has a long data path from source as the number of links per receivers is large. When the group size increases, the average cost first decreases very fast and then it is relatively flat. The average cost rate per receiver is the smallest for layer-1 data and the largest for layer-2 data because our algorithm gives priority for layer-1 multicast-tree construction. We also want to point out that, even though the average cost rate per link is 7.25/Mb in our network topology and the average number of links per receiver is much larger than one because there are some routing nodes in a multicast super-tree, the average cost rate per receiver for a large group size is only about 6.5, 7.7, 10.0 ($\$/Mb$ per receiver) for layer 1, layer 2, and both layers, respectively. The average cost rate per receiver for layer 1 is less than the average of a link's cost rate (7.25), and the average cost rate per receiver per layer is only a little bigger than the average cost rate per link. This confirms that our algorithm is an effective algorithm in finding the minimum-cost, delay-constrained multicast super-tree.

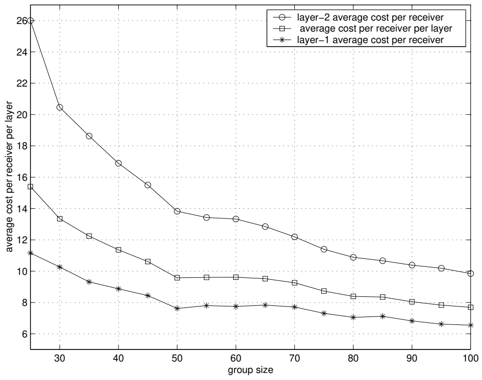

Fig. 6. Average cost per layer per receiver vs. group size.

V. Conclusion

This paper presented a new distributed algorithm (LVMSR) with $O(Rn^2)$ time complexity and $O(R^2)$ message complexity to construct layered-video multicast data-paths. The multicast data-paths form a multicast super-tree. The super-tree constructed by this algorithm can always satisfy the delay constraint while providing small delay jitter and low cost. By using the LVMSR algorithm, data from different layers may be routed via different paths in case of insufficient bandwidth which can improve the QoS of the received video. Our simulation results show that the algorithm performs well in randomly-generated network topologies.

References

[1] N. F. Maxemchuk, "Video distribution on multicast networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 3, pp. 357-372, April 1997.

[2] X. Jia, "A distributed algorithm of delay-bounded multicast routing for multimedia applications in wide area networks," *IEEE/ACM Transactions on Networking*, vol. 6, no. 6, pp. 828-837, Dec. 1998.

[3] M. Parsa, Q. Zhu, and J. J. Garcia-Luna-Aceves, "An iterative algorithm for delay-constrained minimum-cost multicasting," *IEEE/ACM Transactions on Networking*, vol. 6, no. 4, pp. 461-474, Aug. 1998.

[4] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, "Multicast routing for multimedia applications," *IEEE/ACM Transactions on Networking*, vol. 1, pp. 286-292, June. 1993.

[5] B. M. Waxman, "Routing of multipoint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617-22, Dec. 1988.

[6] S. McCanne, M. Vetterli, and V. Jacobson, "Low-complexity video coding for receiver-driven layered multicast," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 6, pp. 983-1001, Aug. 1997.

[7] N. Shacham, "Multipoint communication by hierarchically encoded data," *Proc. IEEE INFOCOM ' 92*, pp. 2107-2114, vol. 3, May 1992.

[8] D. Taubman, and A. Zakhor, "Multirate 3-D subband coding of video," *IEEE/ACM Transactions on Image Processing*, vol. 3, no. 5, pp. 572-88, Sept. 1994.

[9] S.L. Hakimi, "Steiner's problem in graphs and its implications," *Networks*, vol. 1, no. 2, pp. 113-133, 1971.

[10] R. M. Karp, "Reducibility among combinatorial problems," *Complexity of ComComputer Computations*, R. E. Miller and J. W. Thatcher, Eds. New York, NY: Plenum, pp. 85-103, 1972.

[11] D. Eppstein, "Finding the k shortest paths," *SIAM Journal on Computing*, vol. 28, no. 2, pp. 652-673, 1998.