On the Heterogeneous Postal Delivery Model for Multicasting

Chandra N. Sekharan, Shankar M. Banik, and Sridhar Radhakrishnan

Abstract: **The heterogeneous postal delivery model assumes that each intermediate node in the multicasting tree incurs a constant switching time for each message that is sent. We have proposed a new model where we assume a more generalized switching time at intermediate nodes. In our model, a child node v of a parent u has a *switching delay vector*, where the ith element of the vector indicates the switching delay incurred by u for sending the message to v after sending the message to $i-1$ other children of u. Given a multicast tree and switching delay vectors at each non-root node in the tree, we provide an $O(n^{\frac{5}{2}})$ optimal algorithm that will decide the order in which the internal (non-leaf) nodes have to send the multicast message to its children in order to minimize the maximum end-to-end delay due to multicasting. We also show an important lower bound result that optimal multicast switching delay problem is as hard as min-max matching problem on weighted bipartite graphs and hence $O(n^{\frac{5}{2}})$ running time is tight.**

Index Terms: **Min-max matching, multicasting, postal delivery model, weighted bipartite graphs.**

I. INTRODUCTION

Multicasting is an efficient communication mechanism in which a source host sends the same message to a group of destination hosts, called the multicasting group. The general strategy of accomplishing this task is to construct a rooted tree T called the multicast tree [1]–[3] that contains the source as the root and the destination hosts as the leaf nodes. A single source shortest path tree can be used as a multicast tree. The primary advantage of using the multicast is that it conserves network bandwidth. Contrasted with the unicast mechanism where separate messages are sent to each destination host from the source host, multicasting avoids sending the same message multiply over links that are common to a source and different destinations. As fewer number of messages are transmitted in multicasting, the network gets less congested. Due to limited network layer support for multicasting in the current Internet, the recent research trend is to implement multicast service in the application layer which is referred as *overlay multicast* [4]–[8]. An overlay network is a virtual network deployed over an existing network. In an overlay network, each individual link which connects two nodes can comprise of several routers and hosts in the underlying physical network.

Manuscript received May 10, 2010; approved for publication by Raouf Boutaba, Division III Editor, May 18, 2011.

C. N. Sekharan is with the Department of Computer Science, Loyola University of Chicago, Chicago, IL 60611, USA, email: chandra@cs.luc.edu.

S. M. Banik is with the Department of Mathematics and Computer Science, The Citadel, Charleston, SC 29409, USA, email: shankar.banik@citadel.edu.

S. Radhakrishnan is with the School of Computer Science, University of Oklahoma, Norman, OK 73019, USA, email: sridhar@ou.edu.

The problem of designing an efficient multicast tree for a given graph with different parameters has been addressed in the literature. Collaborative application such as video-conferencing, online games, and distributed database replication require that each destination should receive the message from a source within a specified delay bound. These applications also require that each destination should receive message from the source at approximately the same time. Given a graph with non-negative delay for each edge, an end-to-end delay bound and a delay variation bound, delay and delay variation bounded multicasting network (DVBMN) problem is defined as finding a multicast tree which satisfies the end-to-end delay bound and the delay variation bound. DVBMN problem is non deterministic polynomial (NP)-complete [9] and heuristics have been proposed by Rouskas *et. al.* [9], Kapoor *et. al*, [10], and Sheu *et. al* [11] for this problem. In our prior work [12] on multicasting, we have proposed the most efficient heuristic for the DVBMN problem.

Given a graph where each edge has a non-negative delay and a non-negative cost, Zhu *et. al* [13] have proposed a heuristic for constructing a minimum-cost multicast tree that satisfies the end-to-end delay constraint. Lee *et. al* [14] have considered delay variation and cost and proposed a scalable heuristic for designing a minimum cost multicast tree that satisfies the delay variation constraint. Bang *et. al* [15] have proposed a heuristic for constructing a multicast tree to transmit a given message of a fixed size from a source to a set of destinations which minimizes the end-to-end delay. Degree constrained multicasting is required for point-to-point networks of switching nodes where a switching node's copying ability is constrained and Bauer *et. al* [16] have proposed a heuristic for designing a degree constrained multicast tree.

Two basic communication models are used to characterize multicast operation on a network. In the first model, known as *telephone* model, a node may send a message to at most one other node in each round. In this model, both the sender and the receiver are busy during the whole sending process. The second model which is a realistic model is known as *postal* model. In the postal model, a sender may send another message before the current message is completely received by the receiver. Bar-Noy *et. al* [17] first introduced the heterogeneous postal delay model in the context of network multicasting. In their model, they consider link delays and switching time delay at each node, and further assume that the time interval between two successive message sends is equal to the switching time. Assume node u has two children v_1 and v_2 and switching time at node u is s_u. Node u sends message to v_1 at time $t = 0$ and the message arrives at v_1 at time $t_1 = \lambda_{uv_1}$, where λ_{uv_1} is the delay of the link (u, v_1). Now, u can send a message to v_2 at time $t' = s_u$. The message arrives at v_2 at time $t_2 = s_u + \lambda_{uv_2}$, where λ_{uv_2} is the

1229-2370/11/$10.00

delay of the link (u, v_2). In this model, the authors assumed that s_u is smaller than λ_{uv_1} and λ_{uv_2}. Brosh *et. al* [18] modified the heterogeneous postal model and proposed the generalized heterogeneous postal (GHP) model where $t_1 = s_u + \lambda_{uv_1}$ and $t_2 = 2s_u + \lambda_{uv_2}$. Given a graph $G = (V, E)$, a multicasting group $M \subseteq V$, a source node $s \in V$, a non-negative switching time s_i for each node $i \in V$, and a non-negative communication delay d_e for each edge $e \in E$, minimum delay multicast (MDM) problem is defined as finding a multicast scheme that minimizes the delay required for sending a message from s to all the nodes in M. As MDM problem is NP-complete [18], both Bar-Noy *et. al* [17] and Brosh *et. al* [18] have provided approximation algorithms for MDM problem.

Given a multicast tree with link delays and switching delay vectors, where all the elements in a switching delay vector are equal, Brosh [19] has provided a polynomial time algorithm using a recursive bottom-up computation to determine the ordering at each non-leaf node such that the delay of the multicast tree is minimum. In this paper, we propose a model where node u has different switching time for each child node v and the message arrival time at each child v depends on the order in which u chooses to send the messages. This model captures the heterogeneous nature of communication links and node hardware on the *overlay network*. Given a multicast tree with link delays and generalized switching delay vectors, the goal of this paper is to determine the order in which the data packets have to be sent to each of the children in the multicast tree in such a way that the maximum end-to-end delay of the multicast tree is minimum.

We will illustrate the concept of switching delays of our model using a virtual network containing hosts as nodes and two hosts are connected by a virtual link which is a multi-hop Internet connection. The hosts communicate using socket level programs using may be a connectionless protocol such as user datagram protocol (UDP). Now, let us assume a multicast tree T on the overlay network with S as the root of the tree. Also, assume that c_1, c_2, and c_3 are the children of S. Every node in the tree will use *sendTo* and *recvFrom* socket utilities to send the packet that originated from S to its children in the tree and to receive the packet sent by its parent in the tree, respectively. Node S will execute *sendTo* three times, once for each of its children in the tree. Note that each of the send places the same size data on to the kernel buffer. Now, we have three copies of the same packet in the kernel send buffer and the UDP takes the segment (containing one data packet obtained as a result of the execution of *sendTo* function) and adds its header which is then passed to Internet protocol (IP) layer. The IP layer adds its header and places the packets in the data link layer queue. The frames in the queue (corresponding to each IP packet) are sent sequentially using both the logical link control protocol and the medium access control protocol. The medium access control layer transmits to the nearest router designated for the given host by gaining exclusive access to the channel and transmitting the frame. The delay experienced by the data link layer in sending a single frame is proportional to the channel access time. The child node that receives information as a result of the second *sendTo* experiences additional delay due to the fact that the frames corresponding to the first *sendTo* have to be completed before its frame can be sent. Based on the discussion of the delays above, it is evident that the order in which the source S will issue the send to its children will decide when the children c_1, c_2, and c_3 will receive the packet from S. Let S send to the children in the following order, first to c_1, then to c_2, and finally to c_3. Let us assume that since S issues the *sendto* c_1 first, the additional delay experienced by it is 0 units. Let c_2 experience an additional delay of 3 units and c_3 experience of 5 units due to the fact that S sent the data packet using the second and third *sendTo* function statement executions at S, respectively. Generalizing this, we will define a delay vector for a child node c_1 with two other siblings to be $< c_1^1, c_1^2, c_1^3 >$, where c_1^i is the additional delay experienced when S sends the data packet to c_1 using the ith *sendTo* statement. Different switching times for different children induces the notion of ordering at the sending node and the delay of a multicasting scheme depends on the ordering at each sending node. We illustrate the scenario with an example.

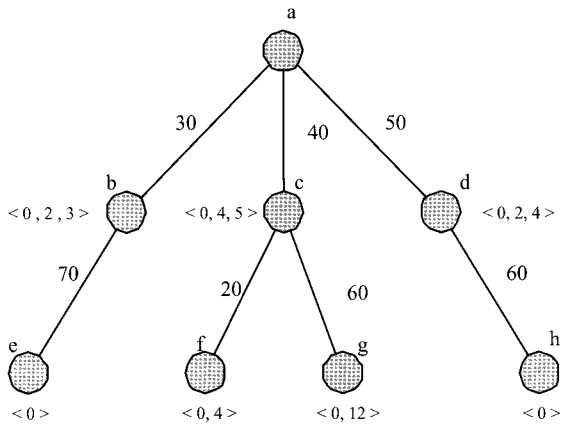

Fig. 1. Multicast tree with link delays and switching delay vectors at each node.

Fig. 1 shows a multicast tree with root node 'a' and the switching delay vectors at each node. The values on the links are the link delays. The leaf nodes of the tree are the nodes in the destination. If we consider only the link delays, the delay of this multicast tree is 110 as it is the maximum of the delays of all the paths a $\sim$ e, a $\sim$ f, a $\sim$ g, and a $\sim$ h. Now, the ordering of packet sends at each non-leaf node will cause additional delay in multicasting as shown in the switching delay vectors at each node. As seen in Fig. 1, when 'a' is sending packets to 'b', 'c', and 'd' in the order of 'b, c, d' nodes 'c', and 'd' will incur additional delay due to processing of packet for 'b' before them. The switching delay vector at node 'b' with respect to node 'a' in Fig. 1 is <0, 2, 3> means that if 'a' sends packet first to 'b', the switching delay at 'b' is 0. If 'a' sends packet second to 'b', the switching delay at 'b' is 2 and if 'a' sends the packet third to 'b', the switching delay at 'b' is 3. If the orderings of packet sends at nodes 'a' and 'c' are 'b, c, d' and 'f, g,' respectively, the delay of the multicast tree becomes 116 (this is the delay of path a $\sim$ g which is 40 + 4 + 60 + 12 = 116).

Given a multicasting tree $T = (V, E)$, a non-negative delay d_e for each edge $e \in E$, and switching delay vector for each non-root node x as $< s_1, s_2, \cdots, s_k >$ where k is the number of siblings of x, we provide a polynomial time algorithm that determines the order in which data packets need to be sent to each node in the multicasting tree so that the delay of the multicast

tree is minimum.

Our paper is organized as follows. In Section II, we formalize our problem definition and provide tools that enable us to build the optimal algorithm. The algorithm and its complexity is presented in Section III. Section IV presents our lower bound results. Conclusions are presented in Section V.

II. FORMAL DEFINITION OF THE PROBLEM

A. Problem Definitions

We define a vector T as an ordered collection of elements, namely, $< v_1, v_2, \cdots, v_k >$. For vector T, we define a bijective mapping function σ: $\{v_1, v_2, \cdots, v_k\} \rightarrow \{1,2,\cdots,k\}$ such that $\sigma(v_j) = j$, $1 \leq j \leq k$. Let $C = C_i$, $1 \leq i \leq k$, be a collection of vectors each having the same cardinality k. This implies that each C_i would look like $C_i = < v_1^i, v_2^i, \cdots v_k^i >$, $1 \leq i \leq k$. A feasible vector of representatives of C is a vector $< v_1, v_2, \cdots, v_k >$ such that $v_i \in C_i$, and $\sigma(v_i) \neq \sigma(v_j)$, $i \neq j$, $1 \leq i, j \leq k$.

Example: Let $C_1 = <0, 2, 1>$, $C_2 = <2, 0, 3>$, and $C_3 = <1, 2, 3>$. A feasible vector of representatives for the collection of sets $\{C_1, C_2, C_3\}$ is $<2, 3, 1>$, whereas $<2, 0, 1>$ is not. The following observations are easy to derive.

Proposition 1: Given a collection $C = \{C_i\}$, $1 \leq i \leq k$, of vectors, a feasible vector of representatives of C always exists.

Proposition 2: Given a collection $C = \{C_i\}$, $1 \leq i \leq k$, of vectors, there exists $k!$ possible feasible vectors of representatives. Let $\Im(C)$ denote the collection of all possible feasible vectors of representatives.

We will denote the set of non-negative real numbers by the notation $\Re^+$. The cartesian product of the set of non-negative real numbers k times will be denoted by $\Re_k^+$, i.e., $\Re_k^+ = \Re^+ \times \Re^+ \times \cdots \times \Re^+ (k \text{ times})$. Let $T = (V, E)$ be a tree with root r that represents the multicasting network of nodes. Let $Sib(v)$ denote the number of siblings of a node v of T, including itself. Trivially, $Sib(r) = 1$, for the root node. We can model the problem of multicasting as follows based on assigning labels or weights to edges of the multicast tree. For each node $v \neq r$, there is a vector called the *switching delay vector* $D(v) = < t_1, t_2, \cdots, t_k >$, where $k = Sib(v)$ and $1 \leq i \leq k$ and $t_i \in \Re^+$. The t_i's are called switching time delays. We know that $t_1 = 0$ for all non-root nodes in the tree. However, this fact is not material to the algorithm discussed here. Given a non-leaf node v, let $v_1, v_2, \cdots, v_k$ be the children of v. Let us denote the edge set $\{(v, v_1), (v, v_2), \cdots, (v, v_k)\}$ by $E(v)$. We define a feasible switching delay vector for the edge set $E(v)$ as $P_v : E(v) \rightarrow \Re_k^+$ such that $P_v = < p_1, p_2, \cdots, p_k > \in \Im(\{D(v_i): 1 \leq i \leq k\})$, where $v_1, v_2, \cdots, v_k$ are the children of v. A feasible switching delay vector P_v induces a natural labeling function f_v: $E(v) \rightarrow \Re^+$, where $f(v, v_i) = p_i$, $1 \leq i \leq k$. Intuitively, a feasible switching delay vector assigns a *label* or a *weight* p_i to each edge (v, v_i) where $< p_1, p_2, \cdots, p_k >$ is a feasible vector of representatives for the collection $\{D(v_i)\}$, $1 \leq i \leq k$. We call the functions f_v, *feasible switching delay functions*. Given a multicast tree T rooted at r and delay vectors $D(v)$ for each non-root node v, we can extend the feasible switching delay functions f_v to the whole tree T as follows: A *feasible multicast tree assignment* f_T: $E(T) \rightarrow \Re^+$ such that $f_T(u, v) = f_u(u, v)$, where $(u, v) \in E$. Essentially, a feasible multicast tree assignment assigns a label or a weight to each edge of the tree so that the collection of weights on an edge set $E(v)$ forms a switching delay vector.

We consider a network represented by a graph $G = (V, E)$ with n nodes and m links, where V and E are a set of nodes and a set of links, respectively. Each link $e(i, j) \in E$ is associated with delay $d(e) > 0$. Consider a simple directed path (simply referred as a path) P from i_0 to i_k (denoted $i_0 \sim i_k$) given by $(i_0, i_1), (i_1, i_2), \cdots, (i_{k-1}, i_k)$, where $(i_j, i_{j+1}) \in E$, for $j = 0, 1, \cdots, k\text{-}1$, and all $i_0, i_1, i_2, \cdots, i_k$ are distinct. The path-delay of P is given by $d(P) = \sum_{j=0}^{k-1} d(e_j)$ where $e_j = (i_j, i_{j+1})$. Let S be a node in the network, called the source node, and $D = d_1, d_2, \cdots, d_k$, where $k \leq n\text{-}1$ be the set of destination nodes. The tree-delay of a multicast tree T that spans S and D is given by $d(T) = \max\{d(P_i)\}$ for all $1 \leq i \leq k$, where P_i is path from S to $d_i \in D$ in tree T. The objective of multicasting algorithms known in the literature is to construct the tree T that has the minimum $d(T)$.

Given a leaf node v in T, we know that there exists a unique path $P = (v_1 = r, v_2, \cdots, v_k = v)$ from root node r to v. Let f_T be a feasible multicast tree assignment. We define a *path delay* $PD(v)$ as $PD(v) = \sum_{i=0}^{k} f_T(v_i, v_{i+1})$. Given f_T, we denote the *maximum delay* of f_T by $PD_{\max}(f_T) = \max\{PD(v): v \text{ is a leaf node of } T\}$. We define an *optimal multicast tree assignment* as a feasible multicast tree assignment f_T^{OPT} such that $PD_{\max}(f_T^{\text{OPT}}) = \min\{PD_{\max}(f_T): \text{For all feasible multicast tree assignments } f_T \text{ for } T\}$. We will call $PD_{\max}(f_T^{\text{OPT}})$ or simply $PD_{\text{OPT}}(T)$, the *optimal multicasting switching delay* for T. The problem is to compute both f_T^{OPT} and $PD_{\text{OPT}}(T)$ in an efficient manner. To solve this problem, we consider the min-max matching problem on a graph and establish a relationship.

III. OUR SOLUTION

A. Min-Max Matching Problem on Weighted, Bipartite Graphs

Let $G = (X, Y, E)$ be a weighted, complete bipartite graph where X and Y are the vertex set partitions and E the edge set of G. Furthermore, let us assume that $|X| = |Y|$, and that the weights are from $\Re^+$. A *perfect matching* for G is a set of edges M of G such that no two edges of M are incident on a common vertex of G and M has maximum cardinality with this property. For G, trivially, a perfect matching having $|X|$ edges exists. The problems of computing a matching of maximum cardinality and a perfect matching are well studied in the literature [22]. We define *heavy weight* of a perfect matching M for G as $h(M) = \max\{\text{weight of edge } e: e \in M\}$. A min-max matching of G is a perfect matching N of G such that $h(N) = \min\{h(M): M \text{ is a perfect matching of } G\}$. The problem of min-max matching and its dual the max-min matching are problems of independent interest and arise in many scheduling applications. The following lemmas address the complexity of computing a min-max matching for a complete, weighted bipartite graph G.

Lemma 1: The sequential time-complexity for obtaining a min-max matching of a weighted, complete bipartite graph is the same as finding the maximum cardinality matching of a bipartite graph [20], [21].

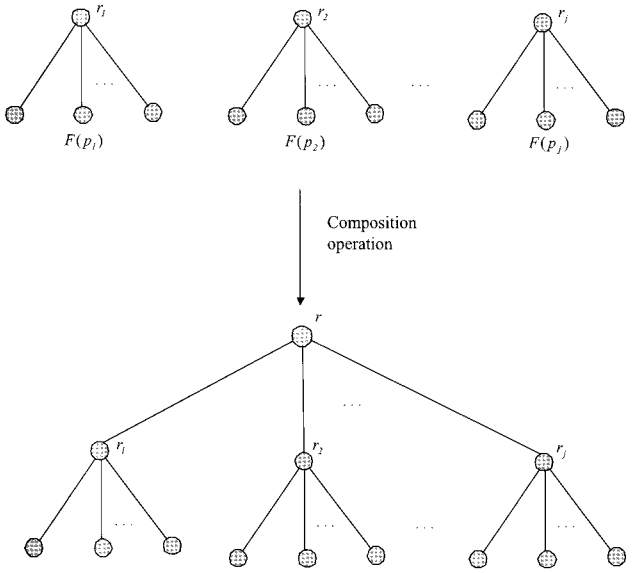

Fig. 2. Hook-up fan.

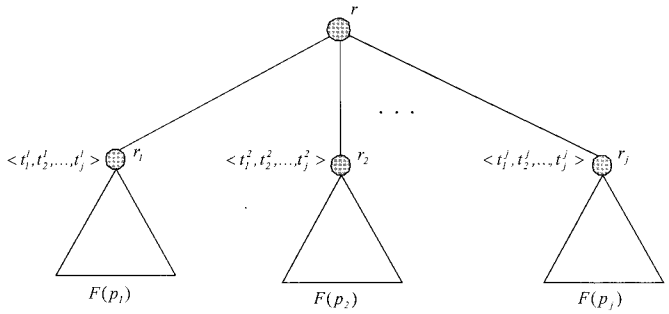

Fig. 3. Hook-up fan H with switching delay vectors.

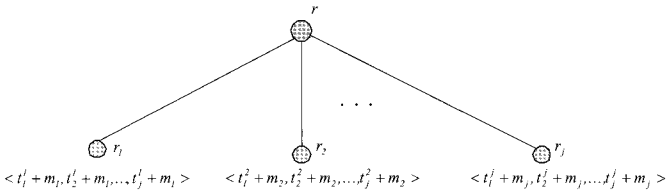

Fig. 4. Fan $F(j)$ with new switching delay vectors.

Lemma 2: Given a complete, weighted bipartite graph, a maximum weighted matching can be determined in $O(m\sqrt{n})$ time [22].

The above result of [22] was improved by [23] in 1995 to $O(\frac{m\sqrt{n}}{k(m,n)})$, where $k(x,y) = \log x / \log \frac{x^2}{y}$.

B. A Special Case of the Multicast Tree Problem

Let us consider a degenerate case *fan* of the multicast tree. A fan $T = (V, E)$ is a multicast tree with $k + 1$ nodes, where k of the $(k + 1)$ nodes are leaves attached directly to the root node. To be more descriptive, let us also say that the leaf nodes are v_1, $v_2, \cdots, v_k$ attached to the root r. Let $D_i = < t_1^i, t_2^i, \cdots, t_k^i >$, $1 \leq i \leq k$, be the switching delay vector for node v_i. We construct a weighted, complete bipartite graph $G = (X, Y, E)$ from T as follows. We let $X = \{v_1, v_2, \cdots, v_k\}$, $Y = \{1, 2, \cdots, k\}$, and the edge set $E = \{(v_i , j)$: $1 \leq j \leq k$, $1 \leq i \leq k\}$. In other words, each vertex of X is connected to all of the vertices of Y. The weight of an edge $e = (v_i, j) \in E$ is given by $w((v_i ,j)) = t_j^i$, $1 \leq i, j \leq k$.

It is fairly straightforward to see that a feasible switching delay vector of T is a vectorized representation of the set of weights in a weighted, perfect matching M of G where the ordering is from 1 through k. Secondly, because T is a fan, $f_r(T)$ is the same as f_T, where r is the root node of T and for all multicast tree assignments of f_T. Thirdly, the path delay $PD(v_i) = f_T(r, v_i)$ for each leaf node v_i. Hence given a multicast tree assignment f_T, the maximum delay $PD_{\max}(f_T)$ is the heavy weight of the corresponding weighted, perfect matching on G. In the same vein, it is easy to see that an optimal multicast tree assignment for T can be obtained by finding a min-max matching for the transformed graph G. Finally, the construction of G from T can be done in time $O(n^2)$, where n is the number of nodes in the fan. The number of edges in the bipartite graph is n^2. Based on the above remarks, Lemmas 1, and 2, the following lemma can be obtained.

Lemma 3: Given a multicast fan T, a special case of a tree, an optimal multicast tree assignment for T and the corresponding optimal multicasting switching delay can be found in $O(n^{\frac{5}{2}})$ time, where n is the number of nodes in T.

C. Hook-up Fans

We will use the notation $F(p)$ for a fan with p leaves, having $(p + 1)$ nodes including the root. Given a collection of vertex-disjoint fans $F(p_1)$, $F(p_2), \cdots$, $F(p_j)$ with roots r_1, $r_2, \cdots$, r_j respectively, a hook-up fan is defined as the composition of the collection of fans $F(p_i)$, $1 \leq i \leq j$, such that the hook-up fan is a tree $T = (V, E)$ satisfying the following properties.

1) $V(T) = \bigcup_{i=1}^{j} V(F(p_i)) \bigcup r$ where V denotes the vertex set and r the root of T.
2) The edge set of T, $E(T) = \bigcup_{i=1}^{j} E(F(p_i)) \bigcup \{(r, r_i)$: $1 \leq i \leq j\}$.

Diagrammatically, the hook-up fans obtained by the composition operation looks as shown in Fig. 2.

D. Optimal Multicast Tree Assignment for a Hook-up Fan

We know from the previous section how to compute an optimal multicast tree assignment for a fan. In this section, we will show a method to obtain an optimal multicast tree assignment for a hook-up fan. Consider a hook-up fan H with switching delay vector as shown in Fig. 3. The switching delay vectors at nodes r_i are indicated in Fig. 3 as $D(r_i) = < t_1^i, t_2^i, \cdots, t_j^i >$, for $1 \leq i \leq j$.

Let m_1, m_2, $\cdots$, m_j be the optimal multicasting switching delays for fans $F(p_1)$, $F(p_2), \cdots, F(p_j)$, respectively. We know that these can be obtained by using Lemma 3. Let $f_{F(p_i)}$, $1 \leq i \leq j$ be the corresponding optimal multicast fan assignments. We transform the hook-up fan to a fan $F(j)$ as shown in Fig. 4 along with new switching delay vectors. The switching delay vectors for the fan in Fig. 4 are $D(r_i) = < t_1^i + m_i, t_2^i + m_i, \cdots, t_j^i + m_i >$ for $1 \leq i \leq j$. We now compute an optimal multicast tree assignment $f_{F(j)}^{\text{OPT}}$ for fan $F(j)$ and the cor-

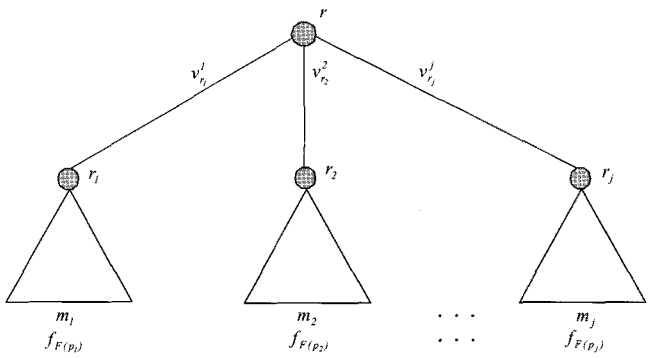

Fig. 5. Fan $F(j)$ with feasible switching delay vectors.

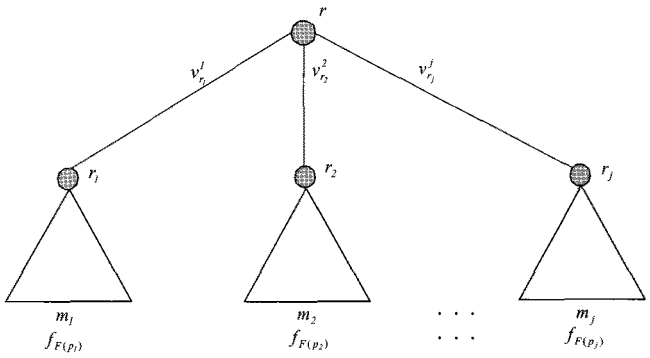

Fig. 6. Hook-up fan H with optimal multicast tree assignments.

responding optimal multicasting delay $PD_{\text{OPT}}(F(j))$. Let $f^{\text{OPT}}_{F(j)} = < l_1, l_2, \cdots, l_j >$. We know that each l_i is of the form $v^i_{r_i} + m_i$, $1 \leq i \leq j$. Secondly, $< v^1_{r_1}, v^2_{r_2}, \cdots, v^j_{r_j} >$ is a feasible switching delay vector for edge set $E(r)$ in H. Based on this, we will re-work the solution obtained on $F(j)$ as a solution for the original hook-up fan H as indicated in Fig. 5. Let $f_r(r_i) = v^i_{r_i}$, $1 \leq i \leq j$.

Lemma 4: For the hook-up fan H in Fig. 5, the multicast tree assignment f_H given by f_r and $f_{F(p_i)}$, $1 \leq i \leq j$ is a feasible multicast tree assignment.

Proof: $f_{F(p_i)}$, $1 \leq i \leq j$ are feasible multicast fan assignments for $F(p_i)$. $< v^1_{r_1}, v^2_{r_2}, \cdots, v^j_{r_j} >$ is a feasible switching delay vector. In the reminder of this section, we will show that f_H is also an optimal multicast tree assignment for H. We need a few results before that. Let H be a hook-up fan as shown in Fig. 6 with an optimal multicast tree assignment as indicated. $u_i = PD_{\max}(f_{F(p_i)})$, $1 \leq i \leq j$. Let $t_s + u_s = PD_{\text{OPT}}(H)$, where $s \in \{1, 2, \cdots, j\}$, without loss of generality. In other words, there could be more than one path from r with the same value for optimal delay. We break ties arbitrarily and pick one such indexed by. □

Lemma 5: Given H as in Lemma 4, u_s is optimal for $F(p_s)$, i.e., $u_s = PD_{\text{OPT}}(F(p_s))$ where $s \in \{1, 2, \cdots, j\}$.

Proof: Suppose u_s is not optimal for $F(p_s)$. Then, there exists an optimal assignment for $F(p_s)$ such that the optimal multicasting switching delay $v_s = PD_{\text{OPT}}(F(p_s))$. Clearly, then $v_s < u_s$. It is clear that using this new assignment for $F(p_s)$, we could construct another feasible assignment for H. Let us call this new feasible assignment for H, f^{OPT}_H[new]. In f^{OPT}_H[new], we have new values for the path delays originating at r and ending at leaves of $F(p_s)$. In particular, the maximum path delay of $u_s + t_s$ becomes $v_s + t_s$. We know that $v_s + t_s < u_s + t_s$. Two possibilities exist for the optimal assignment of H.

1) $v_s + t_s > u_i + t_i$, $i \neq s$, $1 \leq i \leq j$ or
2) $\exists q \in \{1, 2, \cdots j\}$, $q \neq s$ such that $u_q + t_q > u_i + t_i$, $1 \leq i \leq j$ and $i \neq s$ and $u_q + t_q > v_s + t_s$.

In case (i), we have a new min-max value $(v_s + t_s) < (u_s + t_s)$. And this is a contradiction. In case (ii), there is a new min max delay on a different path. In this case, $u_q + t_q > v_s + t_s$ and $u_q + t_q < u_s + t_s$. Hence $u_q + t_q$ is a maximum that is less than the optimal value $u_s + t_s$. Again, this is a contradiction. □

Lemma 5 is crucial because it suggests that we could have sub-optimal solutions for all but one fan and still get an optimal solution or assignment for a hook-up fan. The next lemma extends this idea and shows that any optimal assignment for a hook-up fan can be made to consist of optimal assignments for all fans in a hook-up fan.

Lemma 6: For a hook-up fan H, and an optimal multicast tree assignment f^{OPT}_H, there exists another optimal multicast tree assignment f^{OPT}_H[new] such that all the fans of H, $F(p_1)$, $F(p_2)$,$\cdots$, $F(p_j)$ have optimal assignments.

Proof: From Lemma 5, we know that there exists one fan $F(p_s)$ with an optimal assignment, where $s \in \{1, 2, \cdots, j\}$. Without loss of generality, let $F(p_q)$ be a fan which does not have an optimal assignment where $q \neq s$ and $q \in \{1, 2, \cdots, j\}$. Let $PD_{\text{OPT}}(H)$ be the optimal multicasting switching delay for H. We know that $PD_{\text{OPT}}(H)$ is of the form $u_s + t_s$ where u_s is the optimal value for $F(p_s)$. Hence $u_s + t_s > u_i + t_i$, $i \neq s$, $1 \leq i \leq j$. In particular, $u_s + t_s > u_q + t_q$ where u_q is suboptimal for $F(p_q)$. Let v_q be optimal for $F(p_q)$. Then, $v_q < u_q$ and hence by substituting an optimal assignment for $F(p_q)$, we get a new assignment for H. The only change in the path delay is the value of the qth path where $u_q + t_q$ changes to $v_q + p_q$. Since $u_s + t_s > u_q + t_q$, we have $u_s + t_s > v_q + p_q$. This implies that the new optimal assignment preserves the value of optimal delay $u_s + t_s$. Hence, all suboptimal assignments for the fans can be replaced by optimal assignments without a change to the optimal value $u_s + t_s$. □

Lemmas 5 and 6 lead to the following important theorem.

Theorem 7: Given a hook-up fan H as in Lemma 4 with multicast tree assignment f_H, f_H is an optimal multicast tree assignment.

Proof: We know from Lemma 4, f_H is a feasible assignment for H made up of f_r and $f_{F(p_i)}$, $1 \leq i \leq j$. We also know that f_H is an optimal solution to the system of switching delay vectors (of the fan obtained from H) $< t^i_1, m_i, t^i_2 + m_i, \cdots, t^i_j + m_i >$, where $m_i = PD_{\text{OPT}}(F(p_i))$, and $< t^i_1 + t^i_2, \cdots, t^i_j > = D(r_i)$ of H, $1 \leq i \leq j$. From Lemma 6, we know that there exists an optimal solution for hook-up fan H, whose fans also have optimal multicast tree assignments. This is shown in Fig. 7.

In Fig. 7, m_i is the maximum delay for fan $F(p_i)$ and m_i is optimal for $F(p_i)$, $1 \leq i \leq j$. The optimal switching delay for H is $\max\{u_i + t_i : 1 \leq i \leq j\}$. Secondly, delays $\{u_i + t_i : 1 \leq i \leq j\}$ are an optimal solution to the same set of switching delay vectors $< t^i_1 + m_i, t^i_2 + m_i, \cdots, t^i_j + m_i >$, $1 \leq i \leq j$. Hence the theorem. □

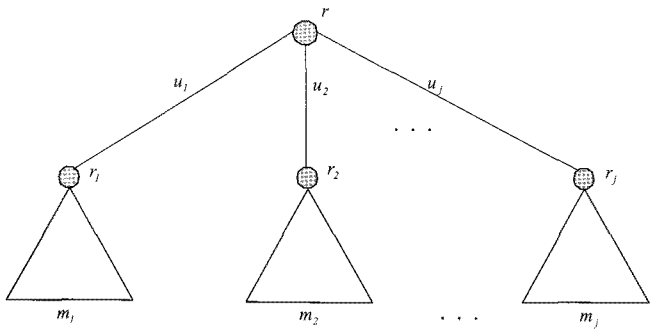

Fig. 7. Hook-up fan H with optimal multicasting.

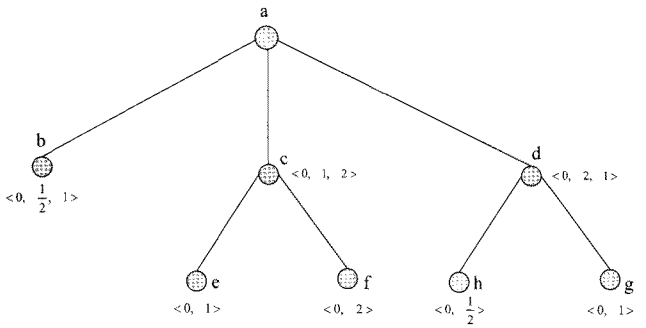

Fig. 8. Multicast tree with duplicating delay vectors at nodes.

Theorem 7 tells us that we can obtain an optimal solution to a hook-up fan by a bottom-up approach. Any multicast tree can be obtained by a series of hook-up operations starting from the base fans.

E. Algorithm and Its Time Complexity

1) Find optimal solutions to base fans $F(p_i)$. Let $PD_{\text{OPT}}(F(p_i))$ be the delays.
2) Hook them up and add $PD_{\text{OPT}}(F(p_i))$ to switching delay vectors.
3) Find optimal solutions to hook-up fans with such modified switching delay vectors.

Repeat steps 1-3 until the root of the tree is reached. After the root is reached, re-work the obtained solutions top-down to get the complete tree assignment.

To derive the complexity of our algorithm, we consider the result of Lemma 3. For each fan $F(p_i)$, using Lemma 3, we can compute an optimal solution in $O(p_i^{\frac{5}{2}})$ time where p_i is the number of leaves in fan $F(p_i)$. During the bottom-up approach, let us say, we have a sequence $l_1, l_2, \cdots, l_j$ leaves when we get to the root where $l_1 + l_2 + \cdots + l_j = O(n)$. Hence, the running time is bounded by $\sum_{i=1}^{j} O(l_i^{\frac{5}{2}}) \leq (\sum_{i=1}^{j} l_i)^{\frac{5}{2}} = O(n^{\frac{5}{2}})$.

Theorem 8: The optimal multicast tree assignment problem can be solved in $O(n^{\frac{5}{2}})$ time.

F. Illustration of the Algorithm

Bottom-up approach to computing the optimal multicasting tree assignment using hook-up fan decomposition is shown in Fig. 8. For simplicity, we assume that the link delays are the same on all links. The steps for computing the optimal solutions for fans from Fig. 8 are shown in Figs. 9 and 10. Re-working the optimal solutions, we get the optimal multicast tree shown in Fig. 11 with $PD_{\text{OPT}}(T) = 1\frac{1}{2}$ unit and the ordering at node 'a' is 'c', 'b', and 'd'. The ordering at node 'c' is 'f' and then 'e.' The ordering at node 'd' is 'h' and then 'g.'

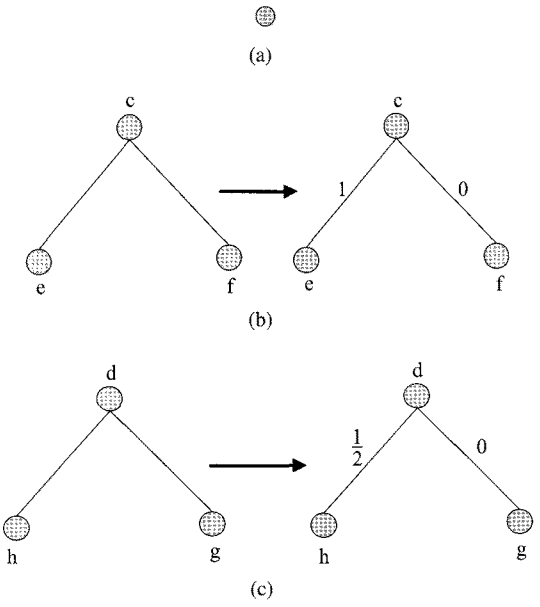

Fig. 9. Optimal solutions for fan: (a) b, (b) c, and (c) d.

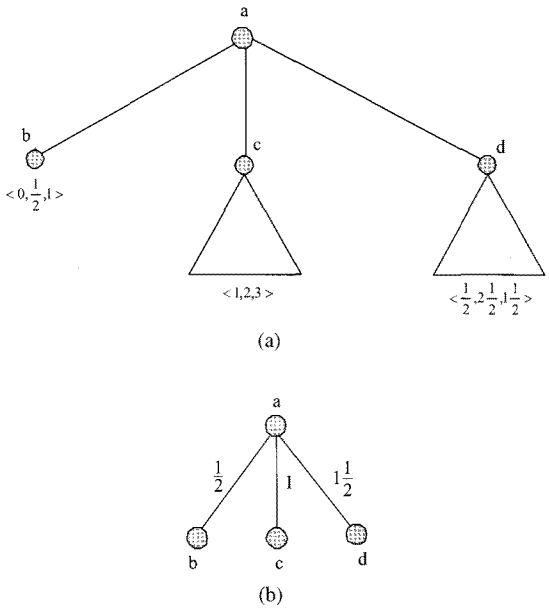

Fig. 10. (a) The hook-up fan and (b) its optimal solution.

IV. LOWER BOUND RESULT

From Lemma 3, we know that given a multicast fan T, a special case of a tree, an optimal multicast tree assignment for T and the corresponding optimal multicasting switching delay can be found in $O(n^{\frac{5}{2}})$ time, where n is the number of nodes in T. Conversely, we can also show in a straightforward fashion that solving the multicast tree problem is at least as hard as the min-max matching problem. Hence, it is unlikely that the above

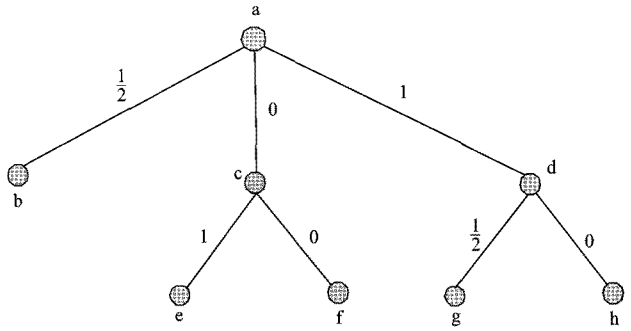

Fig. 11. The optimal multicasting tree.

time-complexity can be improved easily. To see this, let there be a weighted, complete bipartite graph $G = (X, Y, E)$ where $X = \{v_1, v_2, \cdots, v_k\}$, $Y = \{1, 2, \cdots, k\}$, and edge set $E = \{(v_i, j)$: $1 \leq j \leq k, 1 \leq i \leq k\}$. The weight of an edge $e = (v_i, j) \in E$ is given by $w((v_i, j)) = t_j^i$, $1 \leq i, j \leq k$.

We transform this graph into a fan $T = (V, E)$ which is a multicast tree with $k + 1$ nodes, where k of the $(k + 1)$ nodes are leaves attached directly to the root node. Let the leaf nodes be $v_1, v_2, \cdots, v_k$ attached to the root node that we call v. Let $D_i = w((v_i,j)) = t_j^i$ where $1 \leq j \leq k$ for each i, $1 \leq i \leq k$. Indeed, D_i can be taken to be the switching delay vector for node v_i.

Furthermore, it is easy to see that computing the min-max matching on G can be achieved by computing the optimal multicast tree assignment of T. Noting that solving the optimal multicast tree assignment for an arbitrary tree is as hard as for a special case of fan, we have proved that the optimal multicast tree assignment problem has a lower bound of $O(n^{\frac{5}{2}})$ time.

V. CONCLUSION

In this paper, we have considered a more generalized form of switching delay vectors where all the elements of a vector may not be equal. Given a multicast tree with link delays and generalized switching delay vectors at each non-leaf node, we provide an algorithm which schedules the message delivery at each non-leaf node in order to minimize the delay of the multicast tree. Our algorithm, which has a complexity of $O(n^{\frac{5}{2}})$, uses the concept of min-max matching problem on bipartite graphs. We also show an important lower bound result that optimal multicast switching delay problem is as hard as min-max matching problem on bipartite graphs. As part of our future work, we will develop an algorithm for finding the order in a multicast tree such that the end-to-end delay variation from the root to any two leaf nodes is minimum. Another logical extension to our work would be to consider the link delays and switching delay vectors as probabilistic functions.

REFERENCES

[1] L. Kou, G. Markowsky, and L. Berman, "A fast algorithm for steiner trees," *Proc. Acta Informatica*, vol. 14, pp. 145–151, 1981.

[2] S. Ramanathan, "Multicast tree generation in networks with asymmetric links," *IEEE/ACM Trans. Netw.*, vol. 4, no. 4, pp. 573–568, 1996.

[3] H. Takahashi and A. Matsuyama, "An approximate solution for the Steiner problem in graphs," *Mathematica Japonica*, vol. 24, no. 6, pp. 573–577, 1980.

[4] S. Y. Shi and J. S. Turner, "Multicast routing and bandwidth dimensioning in overlay networks," *IEEE J. Sel. Areas Commun.*, vol. 20, no. 8, pp. 1444–1455, Oct. 2002.

[5] S. Y. Shi and J. S. Turner, "Routing in overlay networks," in *Proc. IEEE INFOCOM*, June 2002, pp. 1200–1208.

[6] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, and S. Khuller, "Construction of an efficient overlay multicast infrastructure for real-time applications," in *Proc. IEEE INFOCOM*, Mar. 2003.

[7] A. Riabov, Z. Liu, and L. Zhang, "Overlay multicast trees of minimal delay," in *Proc. IEEE ICDCS*, 2004.

[8] Y. H. Chu, S. G. Rao, S. Seshan, and H. Zhang, "Enabling conferencing applications on the Internet using an overlay multicast architecture," in *Proc. ACM SIGCOMM*, Aug. 2001, pp. 55–67.

[9] G. N. Rouskas and I. Baldine, "Multicasting routing with end-to-end delay and delay variations constraints," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 3, pp. 346–356, 1997.

[10] S. Kapoor and S. Raghavan, "Improved multicast routing with delay and delay variation constraint," in *Proc. IEEE GLOBECOM*, 2000, pp. 476–480.

[11] P. Sheu and S. Chen, "A fast and efficient heuristic algorithm for the delay and delay variation bound multicast tree problem," in *Proc. Int. Conf. Inf. Netw.*, Feb. 2001, pp. 611–618.

[12] S. M Banik, S. Radhakrishnan, and C. N. Sekharan, "Multicast routing with delay and delay variation constraints for collaborative applications on overlay networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 3, pp. 421–431, Mar. 2007.

[13] Q. Zhu, M. Parsa, and J. J. Garcia-Luna-Aceves, "A source-based algorithm for delay-constrained minimum-cost multicasting," in *Proc. IEEE INFOCOM*, Apr. 1995, pp. 377–385.

[14] H. Lee and C. Youn, "Scalable multicast routing algorithm for delay-variation constrained minimum-cost tree," in *Proc. IEEE ICC*, 2000, pp. 1343–1347.

[15] Y. Bang, S. Radhakrishnan, N. S. V. Rao, and S. G. Batsell, "On multicasting with minimum end-to-end delay," in *Proc. Int. Conf. Comput. Commun. Netw.*, Oct. 1999, pp. 604–609.

[16] F. Bauer and A. Verma, "Degree-constrained multicasting in point-to-point networks," in *Proc. Joint Conf. IEEE Comput. Commun. Societies*, Apr. 1995.

[17] A. Bar-Noy, S. Guha, J. Naor, and B. Schieber, "Message multicasting in heterogeneous networks," *SIAM J. Comput.*, vol. 30, no. 2, pp. 347–358, 2000.

[18] E. Brosh, A. Levin, and Y. Shavitt, "Approximation and heuristic algorithms for minimum-delay application-layer multicast trees," *IEEE/ACM Trans. Netw.*, vol. 15, no. 2, pp. 473–484, Apr. 2007.

[19] E. Brosh, "Approximation and heuristic algorithms for minimum delay application-layer multicast trees," *Mater's Thesis*, Department of EE-Systems, Tel-Aviv University, Israel, 2003.

[20] O. Gross, "The bottleneck assignment problem: An algorithm," in *Proc. Rand Symposium on Mathematical Programming*, Rand Publications R-351, 1960, pp. 87-88.

[21] E. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt, Rheinhart and Wilson Publishing Company, 1976.

[22] J. E. Hopcroft, and R. M. Karp, "An $n^{\frac{5}{2}}$ algorithm for maximum matchings in bipartite graphs," *SIAM J. Comput.*, vol. 2, pp. 225–231, 1973.

[23] T. Feder and R. Motwani, "Clique partitions, graph compression, and speeding-up algorithms," *J. Comput. Syst. Sci.*, vol. 51, pp. 261–272, 1995.

Chandra N. Sekharan is currently Professor and Chair of the Department of Computer Science and Assistant to the Provost at Loyola University of Chicago. He was a Fellow of the Fellows Program of the American Council on Education during 2009-2010. He received a B.S. in Electrical Technology and Electronics degree, an M.S. in Computer Science from the Indian Institute of Science, Bangalore, India and Ph.D. in Mathematical Sciences from Clemson University. His research interests are in the general areas of parallel and distributed computing, graph algorithms, and wireless networking in which he has published extensively and received extramural funding from government agencies and private companies.

Shankar M. Banik received the BTech (Hons) degree in Computer Science and Engineering from the Indian Institute of Technology, Kharagpur in 1997 and the M.S. and Ph.D. degrees in Computer Science from the University of Oklahoma in 2001 and 2006, respectively. In 2006, he joined the Department of Mathematics and Computer Science, The Citadel, where he is currently an Assistant Professor. His research interests include collaborative computing, overlay networks, multicasting, mobile and distributed systems, and network security. He is a Member of the IEEE.

Sridhar Radhakrishnan received his B.Sc. (Physics) degree in 1983 from Vivekananda College, Chennai, India, B.S. (Computer Science) degree in 1985 from University of South Alabama, Mobile, Alabama, and M.L.I.S degree in 1986, M.S. (System Science) degree in 1987, and Ph.D. (Computer Science) degree in 1990 all from Louisiana State. He joined the Faculty at the University of Oklahoma in 1990 and is currently a Professor and Director at the School of Computer Science. His research interests are in the areas of network algorithms, parallel algorithms, protocol design for wireless and mobile computing, power aware protocols in mobile networks, algorithms for quality of service routing in broadband networks, and resource allocation problems in wireless networks. He has published over 100 research articles in journals, conference proceedings, and book chapters. He is a Senior Member of the IEEE.