World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:3, No:3, 2009

An Efficient Algorithm for Delay Delay-variation Bounded Least Cost Multicast Routing

Manas Ranjan Kabat, Manoj Kumar Patel, and Chita Ranjan Tripathy

Abstract—Many multimedia communication applications require a source to transmit messages to multiple destinations subject to quality of service (QoS) delay constraint. To support delay constrained multicast communications, computer networks need to guarantee an upper bound end-to-end delay from the source node to each of the destination nodes. This is known as multicast delay problem. On the other hand, if the same message fails to arrive at each destination node at the same time, there may arise inconsistency and unfairness problem among users. This is related to multicast delay-variation problem. The problem to find a minimum cost multicast tree with delay and delay-variation constraints has been proven to be NP-Complete. In this paper, we propose an efficient heuristic algorithm, namely, Economic Delay and Delay-Variation Bounded Multicast (EDVBM) algorithm, based on a novel heuristic function, to construct an economic delay and delay-variation bounded multicast tree. A noteworthy feature of this algorithm is that it has very high probability of finding the optimal solution in polynomial time with low computational complexity.

Keywords—EDVBM, Heuristic algorithm, Multicast tree, QoS routing, Shortest path.

I. Introduction

NOW-A-DAYS Internet applications involve one-to-many or many-to-many communications, where one or more sources send data to multiple receivers. It is possible to provide transmissions to multiple receivers in three different ways: unicast: where a separate copy of the data is delivered to each recipient; broadcast: where a data packet is forwarded to all portions of the network even if only a few of the destinations are the intended recipients; and multicast: a single packet is addressed to all intended recipients and the network replicates packets only when it is needed. Potential applications of multicast are the business world, to help to increase the ability of organizations to communicate and collaborate, leveraging more value from network investment. Examples of multicasting are the transmission of corporate message to employees, video and audio conferencing for remote meeting and teleconferencing, transmission over networks of live TV or radio news and entertainment programs and many more. These new applications are compelling the need for advances in traffic handling to overcome bottlenecks [1].

Before transmitting multicast message, most multicast routing protocols establish the packet delivery path from the source to all the destinations called a multicast tree for efficiently transmitting the message to individual destination nodes. The overall performance of a multicast routing protocol largely depends on the efficiency of the multicast tree. Therefore, the task of building an efficient multicast tree has appeared as an important research topic in multicast communications. There has been plenty of literature dwelling on ways to establish competent multicast tree [1]. One of the most often considered multicast trees is referred to as the Minimum Steiner tree [2], [3].

Minimum Steiner tree (MST) [2], [3] algorithms attempt to minimize the total cost of the multicast tree. The total cost of a multicast tree is generally defined as the sum of costs of all edges in the multicast tree. The cost is usually measured as the bandwidth consumed by the tree. The minimum Steiner tree problem is known to be NP-Complete [5].

In real-time communications, messages must be transmitted to their destination nodes within a certain amount of time failing which the message will be considered as lost. To support real-time multicast communications, computer networks have to guarantee an upper bound on the end-to-end delay from the source node to each of the destination nodes. This is referred to as multicast end-to-end delay problem [4]. Besides, the multicast tree must also guarantee a bound on the variation among the delays along the individual source destination paths, which can avoid causing inconsistency or unfairness problem among users. Such a bound provides synchronization among the various receivers and ensures that no receiver is left behind and that none is far ahead during the lifetime of the multicast session. The Steiner tree under the delay and/or delay-variation constraint is called a Constrained Steiner tree. The problem of finding a Constrained Steiner tree is also NP-Complete [5].

In this paper, we propose an efficient heuristic algorithm, namely, EDVBM, for economic delay and delay-variation bounded multicast routing. The proposed algorithm makes use of two vectors, the least delay path (LDP) vector and the least cost path (LCP) vector. Our algorithm uses a novel heuristic function for finding a sub optimal path closer to the optimal one. This algorithm can easily find a delay constrained multicast tree in polynomial time.

The rest of the paper is organized as follows. Section II gives a formal definition of the delay and delay-variation bounded Steiner tree (DVBST) problem. Some recently proposed related heuristics methods are described in Section III. The Section IV describes the proposed heuristic algorithm EDVBM and its operations. Next we illustrate the applicability of the proposed algorithm with a simple example. The complexity analysis is presented in Section V. The simulation results of

M.R.Kabat is with the Department of Computer Science and Engineering, Veer Surendra Sai University of Technology, Odisha, 768018 INDIA e-mail: (manas_kabat@yahoo.com).

M.K.Patel is with the Department of Computer Science and Engineering, Veer Surendra Sai University of Technology, Odisha, 768018 INDIA e-mail: (patel.mkp@gmail.com).

C.R.Tripathy is with the Department of Computer Science and Engineering, Veer Surendra Sai University of Technology, Odisha, 768018 INDIA e-mail: (cse_uce@yahoo.co.in)

the proposed algorithm are presented in Section VI. Finally, we conclude in Section VII.

II. DVBST Problem

The communication network is modeled as a connected, directed graph $G = (V, E)$, where V is the set of vertices and E is the set of edges representing physical connections between nodes. Let R^+ denote the set of positive real numbers. Two positive functions are defined associated with each link $e(e \in E)$: the delay function $delay(e) : E \rightarrow R^+$ and the cost function $cost(e) : E \rightarrow R^+$. Each link may be asymmetric, that is, the costs and the delays of the link $e = (v_i, v_j)$ and the link $e = (v_j, v_i)$ may have different values.

- *Link cost function:* The link cost is a function of the amount of traffic traversing the link e and the expected buffer space needed for that traffic [2], [11].
- Link delay function: The link delay is the sum of the queuing delay, transmission delay and propagation delay of the link. Let D be the set of destination nodes, then, for each path from the source s to any destination d, $d \in D$, $D \subseteq V$, the delay of the $s-d$ path is defined as the sum of the link delays along the path, i.e.

$$delay(\prod_{s,d}) = \sum_{e \in \prod_{s,d}} delay(e) d \in D$$

  Where $\prod_{s,d}$ is the set of edges constituting the path from source node s to destination node d in the tree T. The maximum end-to-end delay of a multicast tree is the maximum delay from the source to any multicast group member, i.e.

$$Max\_delay(s, D) = \max_{d \in D} \sum_{e \in \prod_{s,d}} delay(e) d \in D$$

- *Delay bound function:* An upper bound U is assigned to the delay along every path from a source to any destination $d \in D$. The delay bound for each destination node can be different since each communication link in the network can have different delay constraint as specified by the multicast application.
- *The multicast delay-variation function δ :* The parameter δ means that the difference of the maximum end-to-end delay and minimum end-to-end delay among the paths from the source node to all the destination nodes has to be kept within δ.

The DVBST problem is to determine a multicast tree connecting the source node s to every destination node $d \in D$ such that the cost of this tree is minimal, while the total delay from the source node to any destination node does not exceed U and the delay-variation is not greater than δ. The problem is to find a tree $T = (V_T, E_T)$; where $V_T \subseteq V$ and $E_T \subseteq E$ such that the total cost of the tree $\sum_{e \in E_T} cost(e)$ is minimized subject to the following three constraints:

- $\{s\} \cup D \subseteq V_T$
- $\sum_{e \in \prod_{s,d}} delay(e) \leq U$
- $$\max_{d_1, d_2 \in D} \left| \sum_{e \in \prod_{s,d_1}} delay(e) - \sum_{e \in \prod_{s,d_2}} delay(e) \right| \leq \delta$$

III. Related Work

In recent years, many research efforts have been directed towards the development of multicast routing algorithms for construction of delay constrained Steiner tree. The first heuristic for delay constrained minimum Steiner tree problem was given by Kompella et.al. [5] and is referred to as KPP. This heuristic computes a constrained closure graph which takes time $O(|U|V^3)$. When the link delays and the user specified delay bound (U) takes non-integer values, KPP multiplies out fractional values to get integers. The KPP approach guarantees construction of a constrained tree if one exists.

The bounded shortest multicast algorithm (BSMA) is another delay-constrained minimum Steiner tree algorithm that is considered the best in terms of tree cost [6]. The BSMA iteratively replaces the edges in the tree until the tree cost cannot be further reduced. The BSMA uses k-shortest path algorithm to find lower cost edges. The time complexity for BSMA is $O(K|V|^3 \log |V|)$, where K may be very large in case of large, densely connected networks and it may be difficult to achieve acceptable running times.

The CAO is another heuristic proposed by Widyono [7]. In CAO, the constrained Bellman-Ford algorithm is used to connect one group member at a time to the source. After each run of the constrained Bellman-Ford algorithm, the unconnected member with the minimum cost constrained path to the source is chosen and is added to the existing subtree. The cost of links in the existing subtree is set to zero. The CAO is always capable of constructing a constrained multicast tree, because of the nature of the breadth-first search conducted. if one exists.

Youssef et.al. [8] proposed a Tabu Search (TS) algorithm to construct a delay-constrained Steiner tree. The TS starts with an initial feasible solution built by using Dijkstras shortest path algorithm, and this tree is called sink tree of source s. For generating neighbors, the sink tree for each destination is constructed. Then one superpath is deleted randomly from the current solution, and neighbors are generated by adding superpaths from one of the destinations sink trees. Kabat et.al. proposed a heuristic algorithm, called Cost-Sensitive Delay-Constrained Multicast (CSDCM) algorithm [3] that finds a cost-sensitive multicast tree by using a novel heuristic function. It first finds the Least Cost Path (LCP) and Least Delay Path (LDP) vectors and then selects the best possible candidate node for the multicast tree by using the heuristic function.

The delay and delay-variation bounded multicast tree (DVBMT) problem has also been proven NP-Complete and a heuristic algorithm called DVMA (Delay Variation Multicast Algorithm) has been proposed in [12]. It is also proved in [13] that unless NP equals P, the DVBMT problem does not have any ϵ-approximation algorithm, where ϵ is a constant. The computer simulations demonstrated in [12], [14] shows that DVMA performs good in terms of multicast delay-variation.

However, the time complexity of DVMA algorithm is very high i.e. $O(klmn^4)$. In the worst case, the maximum value that parameters k and l can take depends on the number of paths satisfying the delay bound between any two nodes. The m and n represents the number of destination nodes and number of nodes in the computer network respectively.

Another efficient heuristic algorithm called DDVCA (Delay and Delay Variation Constraint Algorithm) for delay and delay-variation bounded tree problem has been proposed in [15]. The said algorithm comes from Core Based Tree (CBT) [15] and minimum delay path problem [10]. The CBT is established by choosing some core routers, which compose the core backbone. To determine a proper core router or central node, the DVMA algorithm calculates delay-variation from each core node to all the destination nodes. The core node from which minimum delay-variation is achieved is selected as the central node. Then, the source and destinations are connected to the central node through the minimum delay paths. In comparison with DVMA, this algorithm possesses a significant lower time complexity i.e $O(mn^2)$. The DVMA and DDVCA generate a multicast tree considering delay and delay-variation only, but not the cost.

The DDVBM is the first heuristic distributed algorithm for DVBST problem proposed by Low and Lee in [16]. This algorithm has two phases. In the first phase, the authors attempt to find a minimum cost tree that satisfies the delay constraint. If the tree violates the delay-variation constraint, then it enters into the second phase of the algorithm. In the second phase, the algorithm searches for minimum cost tree that satisfies both delay and delay-variation constraints using a procedure called Path Search. The time complexity for DDVBM is $O(n^3)$. However, here the authors neglect the destination nodes that can be taken as the relay nodes to transmit packets. Hence, while getting the accessorial Steiner node tree, DDVBM fails to construct the multicast tree by connecting the source to all of Steiner nodes.

A distributed multicast routing algorithm has been proposed by Kun et. al. to produce a multicast tree having minimum cost under the delay and delay-variation constraints [2] . This algorithm starts with finding k-least delay paths using distributed k-Bellman Ford (kBF) algorithm [4]. Then, the Simulated Annealing (SA) procedure is called to iteratively reduce the total cost of the multicast tree without violating the delay and delay-variation constraints. The worst case time complexity of the algorithm is $O(k^2m^2nlogk)$. Since this algorithm constructs the multicast tree from k-least delay paths, it may so happen that the multicast tree may not be cost sensitive if k is not sufficiently large. If k is considered to be sufficiently large then the time complexity becomes more. This limits the utility of the k-Bellman Ford algorithm.

IV. THE PROPOSED ALGORITHM: EDVBM

In this section, we introduce a fast and efficient heuristic algorithm EDVBM to solve the DVBST problem. In what follows, we discuss the multicast routing information stored at each node, the working principle of our proposed algorithm followed by the illustrations and complexity analysis.

A. Multicast Routing Information - the vectors

The traditional distance vector routing algorithms require each router to maintain a table (i.e., a vector), which gives the best known distance to each destination and determines outgoing links to be used to reach there. In our proposed algorithm, each node maintains two vectors, the least delay vector and the least cost vector, which provide the best known values based on two different metrics; delay and cost respectively. Each entry in the least-delay vector at a node (e.g., node v_i) contains the following information:

- v_j : the destination node identity;
- $delay(P_{ld}(v_i, v_j))$: the delay of the least-delay path $P_{ld}(v_i, v_j)$;
- $cost(P_{ld}(v_i, v_j))$: the cost of the least-delay path $P_{ld}(v_i, v_j)$;

The least-delay path $P_{ld}(s, d)$ is the path from s to d, which satisfies $delay(P_{ld}(s, d)) = min\{delay(p), p \in P(s, d)\}$, where $P(s, d)$ is the set of all possible paths from s to d. The least-delay path can be obtained by using Dijkstras shortest path algorithm considering delay as the only weight of the link.

Similarly, the entry in the least-cost vector contains the following information:

- v_j : the destination node identity;
- $delay(P_{lc}(v_i, v_j))$: the delay of the least-cost path $P_{lc}(v_i, v_j)$;
- $cost(P_{lc}(v_i, v_j))$: the cost of the least-cost path $P_{lc}(v_i, v_j)$;

The least-cost path $P_{lc}(s, d)$ is the path from s to d, which satisfies $cost(P_{lc}(s, d)) = min\{cost(p), p \in P(s, d)\}$, where $P(s, d)$ is the set of all possible paths from s to d. The least-cost path can be obtained by using Dijkstras shortest path algorithm considering cost as the only weight of the link.

B. Operation of EDVBM Algorithm

The proposed EDVBM algorithm constructs the multicast tree node by node from the source to all the destinations of a multicast group. The nodes are added to the multicast tree by evaluating the heuristic function $weight()$ on all the nodes that are already generated.

The operation of the proposed algorithm is summarized in Fig.1. The algorithm begins with computing the least-delay path from the source to all the destinations of the multicast group by using Dijkstras shortest path algorithm. Then, there the source node checks whether the delay of the least-delay paths from source to all the destinations satisfies the delay constraint. If it is not satisfied for all destinations of the multicast group, there exists no delay constraint multicast tree. If the delay constraint is satisfied, then there exists at least one or more delay constraint multicast tree. The multicast tree starts with the source node and includes the nodes of G one by one till all the destination nodes of the multicast group are completed. The algorithm selects the nodes by evaluating the heuristic function $weight()$. To calculate the $weight()$ at each neighbor node of the current node each node should keep account of the least-cost vector and least-delay

Algorithm EDVBM($\mathbf{G}, \mathbf{cost}, \mathbf{delay}$)

{
Initially at source node s

if ∃ d ∈ D, $delay(P_{ld}(s,d)) > U$ then
  write("No delay-constrained multicast tree exists");
  exit();
else{
  delaysofar(s)=0;
  costsofar(s)=0;
  $V_T = \{s\}$, $E_T = \Phi$, $S_{pq} = \Phi$, $B_{pq} = \Phi$
  $d\_min = +\infty, d\_max = 0$;
  for (i=1 to n) do
    dvcw[i] = dvdw[i] = 0;
}
$\forall v \in V$ and $\forall d \in D$,
    calculate $delay(P_{ld}(v,d))andcost(P_{ld}(vj,d))$;
repeat
{
  Upon receiving PATH_CONSTRUCTION message or at
  for source node s each neighbouring node w
    calculate $weight(newpred(w), w)$
if $(w \in S_{pq})$
{
 / compare the new weight with the earlier weight /
 if $(weight(c\_pred(w), w) > weight(newpred(w), w))$
 {
  $add(B_{pq}, w)$;
  $b\_pred(w) = c\_pred(w)$;
  $c\_pred(w) = newpred(w)$;
 }
 else
  $add(B_{pq}, w)$;
 $b\_pred(w) = newpred(w)$;
}
else if$(w \in V_T)$
{
 if$(weight(t_pred(w), w) > weight(newpred(w), w))$
 {
  $V_T = V_T - t\_pred(w)$
  $E_T = E_T - t\_pred(w), w)$;
  $t_pred(w) = newpred(w)$
  $V_T = V_T \cup t\_pred(w)$
  $E_T = E_T \cup t\_pred(w), w)$;
 }
}
else
{
 $add(S_{pq}, w)$
 $c\_pred(w) = newpred(w)$;
}
 $v = extract(S_{pq})$;
 $V_T = V_T \cup v$
 $t\_pred(v) = c\_pred(v)$;
 $E_T = E_T \cup t\_pred(v), v)$;
if $(S_{pq} == \emptyset)$ then
{
 $v = extract(B_{pq})$;
 $E_T = E_T - \{t\_pred(v), v\}$
 $E_T = E_T \cup \{b\_pred(v), v\}$
 $t\_pred(w) = b\_pred(w)$;
}
$delaysofar(v) = delaysofar\ (t\_pred(v)) +$
                $delay(t\_pred(v), v)$;
if $(v \neq d)$then
 $costsofar(v) = 0.5 \star (costsofar(pred(v)) +$;
                $cost(t\_pred(v), v))$
else
{
 $costsofar(v) = 0$;
 if $(delaysofar(v) > d\_max)$
  $d\_max = delaysofar(v)$;
 if $(delaysofar(v) < d\_min)$
  $d\_min = delaysofar(v)$;
} send path_construction message to v.
}until$(D \subseteq V_T)$

Fig. 1. Pseudo Code for EDVBM algorithm

vector as discussed in section IV. These vectors are calculated from all the nodes to all the destinations by using Dijkstras algorithm [10]. The $delaysofar$ and the $costsofar$ value at the source node is initialized to zero. The priority queues S_{pq} and B_{pq} are initialized to null. The $addpq()$ procedure adds the node to the priority queue. Then it extracts the node that has minimum heuristic weight. Assuming the current node as v_i, the procedure to calculate heuristic function $weight()$ for each neighbor node v_j, is shown in Fig.2.

For each neighbor node w of the current node, if the node is neither in S_{pq} nor in V_T then the generated nodes are stored in S_{pq} and the predecessor is recorded as current predecessor ($c\_pred(w)$). If the node is already in the priority queue S_{pq} then the weight is calculated via the new predecessor. If the earlier weight is less, then the node along with its new predecessor is stored in a backtrack queue B_{pq} and the predecessor is recorded as $b\_pred(w)$. If the new weight is less, then the node is placed in B_{pq} and $c\_pred(w)$ becomes $b\_pred(w)$, and the $c\_pred(w)$ is updated to the new predecessor. The B_{pq} is referred, only when S_{pq} is empty and all the destinations are not yet included in V_T.

If the node w is already in V_T then the weight via its new predecessor is compared with the weight via its old predecessor $t\_pred(w)$. If the new weight is less, then the node is connected with its new predecessor and $t\_pred(w)$ is updated to the new predecessor. The function extract (S_{pq}) is used to choose the node, say w, whose value of the heuristic function $weight(v_i, w)$ is the minimum one among all the live nodes in the priority queue S_{pq}. If more than one node have the same value, it chooses the node that

$dcw(v) = dsfar(u) + delay(u,v) + \min_{d \in D \& d \notin V_T} \{delay(P_{lc}(v,d))\}$

$ddw(v) = dsfar(u) + delay(u,v) + \min_{d \in D \& d \notin V_T} \{delay(P_{ld}(v,d))\}$

$dvcw(v) = dvdw(v) = \delta$
if $(d\_min \neq \infty)\{$
$dvcw(v) = max\{abs(d\_max - dcw(v)), abs(d\_min - dcw(v))\}$
$dvdw(v) = max\{abs(d\_max - ddw(v)), abs(d\_min - ddw(v))\}$
$\}$
if $(dcw(v) \leq U$ and $dvcw(v) \leq \delta)$ then
$weight(u,v) =$

$$costsofar(u) + cost(u,v) + \min_{d \in D \& d \notin V_T} \{cost(P_{lc}(v,d))\}$$

else if $(ddw(v) \leq U$ and $dvdw(v) \leq \delta)$ then
$weight(u,v) =$

$$costsofar(u) + cost(u,v) + \min_{d \in D \& d \notin V_T} \{cost(P_{ld}(v,d))\}$$

else
$weight(u,v) = \infty$

Fig. 2. Pseudo Code to calculate weight of a link (u,v)

comes first in the priority queue. After selecting a node w generated from v_i, the new $delaysofar(w)$ is updated to $delaysofar(v_i) + delay(v_i, w)$. Similarly $costsofar(w)$ is updated to zero if w is a destination node, otherwise updated to half of the $costsofar(v_i) + cost(v_i, w)$. The node currently added to V_T is chosen as the extended node and can be extended further.

The proposed algorithm generates the multicast tree by learning from experience. When a destination is found, the maximum delay, d_max, and minimum delay, d_min, from source to the destinations are updated. The d_max and d_min is used to calculate the delay-variation weight, dvw, at each node. The weights of all the nodes in S_{pq} and B_{pq} are also updated. The destinations are selected one by one and the process continues till all the nodes of the multicast group are included in to the multicast tree.

C. EDVBM Algorithm: An illustration.

For ease of understanding, we consider the example network in Fig. 3. The multicast trees of the network are calculated by Zhang's Simulated Annealing method [2] and our proposed method for the purpose of comparison.

In the example network example network shown in Fig. 3, source={F} and the set of destinations = {B, D, E, H}, where the numbers in the parentheses along each edge represent the cost and delay for that edge, given the delay bound $U = 10$ and the delay-variation bound $\delta = 3$.

Table I shows the Delay of the Least-Delay (DLD) and Cost of the Least-Delay (CLD) paths from all the nodes to the destinations and Table II shows the Delay of the Least Cost (DLC) and Cost of the Least-Cost (CLC) paths from all the nodes to the destinations. First, the weight for the entire neighbor nodes of the source is calculated and the node with minimum weight is selected for the multicast tree. Then the weights for each neighboring nodes of the recently selected

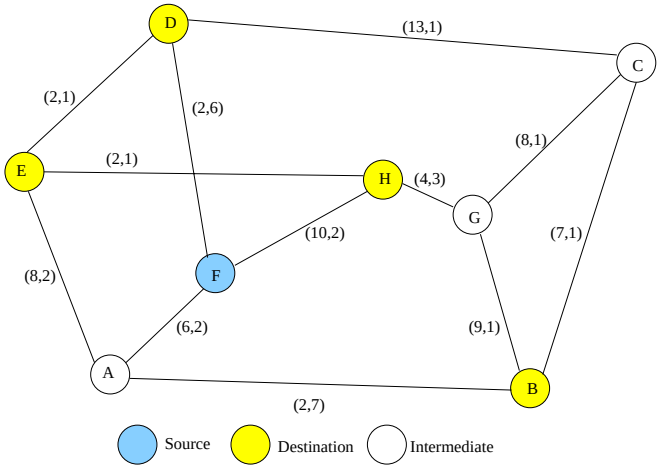

Fig. 3. The Example network

TABLE I
THE DELAY AND COST OF THE LEAST-DELAY PATHS FROM ALL THE NODES TO THE DESTINATIONS.

	DLD				CLD			
Dest	B	D	E	H	B	D	E	H
A	5	3	2	3	30	10	8	10
B	0	2	3	4	0	20	22	24
C	1	1	2	3	7	13	15	17
D	2	0	1	2	20	0	2	4
E	3	1	0	1	22	2	0	2
F	6	4	3	2	23	14	12	10
G	1	2	3	3	9	21	23	4

TABLE II
THE DELAY AND COST OF THE LEAST-COST PATHS FROM ALL THE NODES TO THE DESTINATIONS.

	CLC				DLC			
Dest	B	D	E	H	B	D	E	H
A	2	8	8	10	7	8	2	3
B	0	10	10	12	0	15	9	10
C	7	13	14	12	1	1	5	4
D	10	0	2	4	15	0	1	2
E	10	2	0	2	9	1	0	1
F	8	2	4	6	9	6	7	8
G	9	8	6	4	1	5	4	3
H	12	4	2	0	10	2	1	0

node are calculated. At each step, the BESTNODE (the node with minimum heuristic weight) is selected for the multicast tree V_T and this process is repeated till all the destination nodes are included into the multicast tree. The weights at the nodes including the source node are calculated as follows.

At node F	$V_T = \{F\}$
weight(F,A)=6+2=8	$S_{pq} = \{A, D, H\}$
weight(F,D)= 2	$B_{pq} = \infty, V_T = \{F, D\}$
weight(F,H)=10	$d\_max = 6, d\_min = 6$
At node D	
weight (D,C)=13+7=20	$S_{pq} = \{A, H, C, E\}$
weight (D,E)=2	$V_T = \{F, D, E\}$
	$d\_max = 7, d_min = 6$

At node E
weight (E,H)=2 $S_{pq} = \{AH, C\}$
Weight (E,A)= $+\infty$ $B_{pq} = \{H\}$
$V_T = \{F, D, E, H\}$
$d\_max = 8, d\_min = 6$
At node H
weight (H,G)= $+\infty$ $S_{pq} = \{A, C\}$
$V_T = \{F, D, E, H, A\}$
At node A
weight (A,B)=6+2=8 $S_{pq} = \{B, C\}$
Weight (A,E)= $+\infty$ $V_T = \{F, D, E, H, A, B\}$
$d\_max = 9, d\_min = 6$

To make a comparative study the performance of our EDVBM algorithm with the Zhang's algorithm, the candidate-paths-set for the example network is generated by using distributed kBF algorithm. Table III shows the candidate-paths-set for the example network, where $k = 5$. Then the multicast tree is generated step by step using Zhang's Simulated Annealing procedure. The final multicast tree generated by Zhang's algorithm is shown in Fig. 5. The cost of the multicast tree generated by Zhang's algorithm is 27, the maximum delay is 8 and the delay-variation is 2.

The multicast tree generated by the proposed EDVBM algorithm is shown in Fig.4. The cost of the multicast tree generated by EDVBM algorithm is 14, the maximum delay is 9 and the delay-variation is 3.

TABLE III
THE CANDIDATE-PATHS-SET OF EXAMPLE NETWORK, K=5.

F,H,G,B	6	F,H,E,D	4	F,H,E	3	F,H	2
F,H,E,DC,G,B	7	F,A,E,D	5	F,A,E	4	F,A,E,H	5
F,A,E,D,C,B	7	F,D	6	F,D,E	7	F,D,E,H	8
F,A,E,D,C,G,B	8	F,H,G,C,D	7	F,H,G,C,D,E	8	F,A,E,D,C,G,H	10
F,D,C,B	8	F,A,E,H,G,C,D	10	F,D,C,G,H,E	12	F,D,C,G,H	11

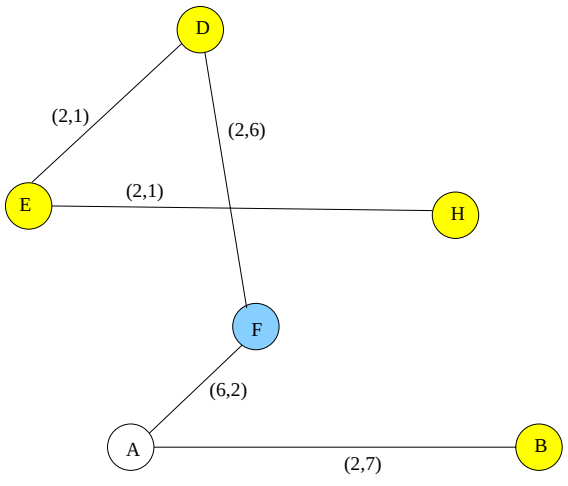

Fig. 4. The multicast tree generated by EDVBM algorithm

D. Correctness of the EDVBM algorithm.

When the contents in the vector at all nodes are up-to date and do not change during the path construction period, our EDVBM algorithm can always find a feasible multicast tree if such a multicast tree exists. To prove the correctness of the proposed EDVBM algorithm, we propose the following theorems.

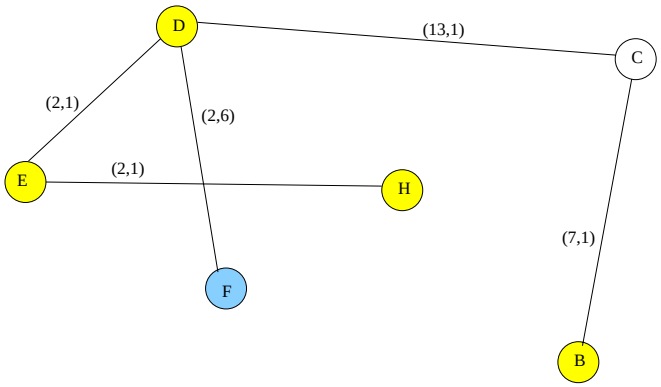

Fig. 5. The multicast tree generated by EDVBM algorithm

Theorem 4.1: The EDVBM algorithm always finds multicast tree from the source to a set of destinations D satisfying the given delay bound U and delay-variation bound δ if such multicast tree exists.

Proof: If for any destination d in D, the condition $delay(P_{ld}(s, d)) \leq U$ does not satisfy, then the EDVBM algorithm would terminates immediately with failure notification at the source tree. If for any destination d in D, the least-delay path can not satisfy the delay constraint then there exists no feasible multicast tree. If it is satisfied for all destinations in D, delay-constrained multicast tree exists and EDVBM algorithm starts.

Then, at each following node v_i, this condition is implicitly guaranteed by the definition of $weight()$ function or inherited from the preceding node. However, in a different format, i.e., $delaysofar(v_i) + delay(v_i, v_j) + \min_{d \in D \& d \notin V_T}\{delay(P_{ld}(v_i, d))\} \leq U$, where the left side of the inequality is the summation of the delay accumulated from s to v_i, delay of $link(v_i, v_j)$ and minimum of the minimum possible delays from the node v_j to all nodes in D that are not yet selected into the multicast tree. We now complete the proof by the induction on node v_i.

At first, s is the only member of the multicast tree V_T, $delaysofar(s)$ is initialized to 0 and $delaysofar(s) + delay(P_{ld}(s, d)) \leq U$. The maximum delay d_max and minimum delay d_min calculated so far is initialized to 0 and ∞ respectively. The source node in the basis of our induction, we assume a sub multicast tree connecting an intermediate node v_i to a set of destinations $D_1, D_1 \in D$. This sub multicast tree is rooted at v_i and we assume that at node $v_i, delaysofar(v_i)$ is already repeated and $\forall d \in D_1, delay(P_{ld}, (v_j, d)) \leq U$ still holds, where v_j is the neighbour node of v_i.

Then the EDVBM algorithm checks the condition, $delaysofar(v_i) + delay(v_i, v_j) + delay(P_{1c}(v_j, d) \leq U$ for all d that are not yet selected in multicast tree V_T and calculates the $weight(v_i, v_j)$ as $costsofar(v_i) + cost(v_i, v_j) + \{cost(P_{lc}(v_j, d))\}$. If that condition fails then the condition $delaysofar(v_i) + delay(v_i, v_j) + delay(P_{ld}(v_j, d) \leq U$ is

checked and the weight is calculated as $costsofar(v_i) + cost(v_i, v_j) + \{cost(P_{ld}(v_j, d))\}$, otherwise the weight is taken as $+\infty$. Once a destination d is found, the d_max and d_min are updated to $delaysofar(d)$, then the root part of the multicast tree is generated by generating the paths to all remaining destinations that satisfy the delay-variation bound.

Our proposed algorithm keeps the records of all the nodes in a priority queue (S_{pq}) that are generated but not yet included into the multicast tree. If a node is already in the priority queue then the weight via new parent is compared with the earlier weight. If the earlier weight is less than the node, its new parent and the corresponding weight is stored in a backtrack priority queue (B_{pq}). Otherwise the node with its new parent is placed in S_{pq} and the old parent is placed in B_{pq}. If the node that is recently generated is already in V_T than the new weight is compared with the earlier weight. If the new weight is cheaper, then the node is connected via its new parent removing the connectivity from its old parent. This is clear from the algorithm that whenever a node v is selected with updated $delaysofar(v)$, for all destination leaf nodes of the sub multicast tree rooted at v, delay and delay-variation bounds are satisfied.

Thus EDVBM algorithm finds a multicast tree satisfying the delay and delay-variation bound. Hence EDVBM algorithm can always find a multicast from the source to set of destination D satisfying the given delay bound U and delay-variation bound δ, if such a multicast tree exists. ■

Theorem 4.2: The multicast tree found by EDVBM algorithm contains no loop.

Proof: After receiving a PATH CONSTRUCTION message each node calculates the heuristic weight for all its neighbors. If the neighbor node is already in the priority queue S_{pq}, then the weight is calculated via its new predecessor and compared with the earlier weight. Then, the node along with its predecessors through which the weight is minimum, is placed in the priority queue and the node with other predecessor and weight is placed in B_{pq}.

If the node is already in V_T and a new predecessor is found through which the weight is minimum then the node is connected with the new predecessor removing the connectivity through its old predecessor. So there is no chance of getting loop in the multicast tree. ■

V. Complexity Analysys.

The proposed algorithm EDVBM maintains two vectors: the least-cost path vector and least-delay path vector. For a network $G = (V, E)$, using the distance vector routing protocols [17], the worst case time complexity for each node to compute the vectors is found to $O(|V|^3)$. Since we assume the EDVBM algorithm to be based on the existing distance vector routing protocols, the overhead for routing vector maintenance is considered using the same parameters as that of distance vector routing. Therefore, the extra overhead introduced by EDVBM algorithm is only computed.

As discussed earlier, in EDVBM algorithm each multicast tree is constructed in an on-demand manner. For each multicast tree generation, a node should evaluate the link selection function $weight()$ atmost l times, where l is the number of neighbors of that node. At each node the neighbor nodes are inserted into a priority queue and then the node with minimum $weight()$ is extracted and the priority queue (or heap) is adjusted. Thus, in the worst case, the extra computational complexity for a node to select the next node is $O(l\ log_2|V|)$, this is because the worst case computational complexity of inserting l neighbor nodes to the heap is $O(l\ log_2|V|)$ and the worst case computational complexity of deleting the node with minimum weight from the heap and adjusting the heap is $O(log_2|V|)$. In the worst case, the multicast tree can have $|V|$ nodes. So the computational complexity for finding an economic delay delay-variation constrained multicast tree is $O(l_{max}|V|log_2|V|)$, where l_{max} is the maximum number of neighbors of any node in the graph.

VI. Simulation Results and Analysis

The proposed algorithm has been implemented in Visual C^{++}. The experiments are performed on a Pentium IV, @ 2.8 G.Hz. and 1 GB RAM based PC platform. A random generator developed by Salama is used to create a random topology [9].

The positions of the nodes are fixed randomly in a rectangle of size 4000 km x 2400 km. The Euclidean metric is then used to determine the distance between each pair of nodes. Edges are introduced between the pairs of nodes u, v with a probability that depends on the distance between them. The edge probability is given by $P(u, v) = \beta exp(-l(u, v)/\alpha L)$, where $l(u, v)$ is the distance from node u to v, L is the maximum distance between two nodes. The α and β parameters are set to 0.15 and 2.2 respectively.

The link delay function $d(e)$ is defined as the propagation delay of the link and queuing and transmission delays are assumed to be negligible. The link cost function $c(e)$ is defined as the current total bandwidth reserved on the link, which is a random variable uniformly distributed between 10 to 120 Mbps. The simulation was run for 200 times for each case and the output is considered as the average of that.

Cost performance

The Fig. 6(a) illustrates the tree cost Vrs network size. Our algorithm generates the multicast tree which satisfies the delay and delay-variation bounds. It also shows that our algorithm has the better performance than that of Zhang. The Fig. 6(b) shows the cost performance versus group size for a 50-node network with $m = 5, U = 35$ ms and $\delta = 20$ ms. From the Fig. 6, it is clearly that the proposed EDVBM algorithm has the best cost performance measures in comparison to that of Zhang. Also it reveals from the result that the tree cost increases as the group size increases. The increase in total cost with its group size is due to increase in group size. Irrespective of the network size or group size, the proposed algorithm EDVBM exhibits less cost in comparison to the method proposed by Zhang by [2].

Delay and delay-variation performance

Next, we compare the delay and delay-variation of the multicast trees in Fig. 7 and Fig. 8 respectively. The Fig. 7(a) shows the delay performance for different number of network

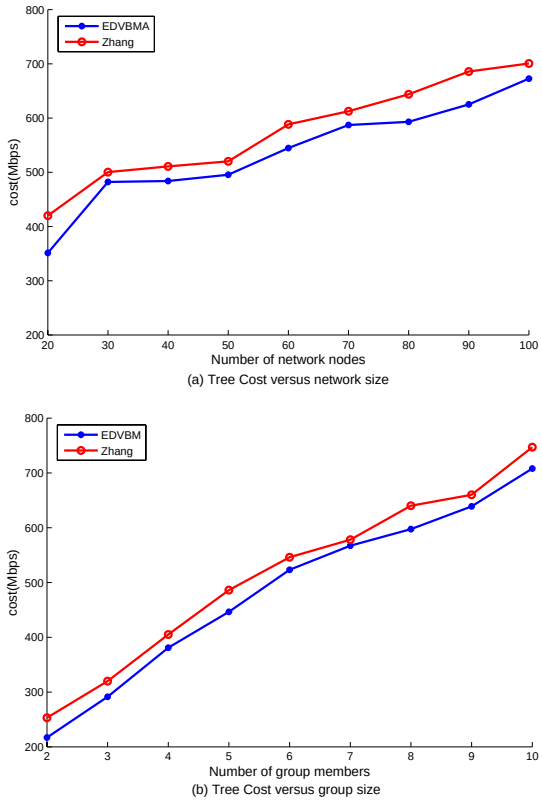

Fig. 6. A comparison on the tree cost of the EDVBM and Zang's algorithm.

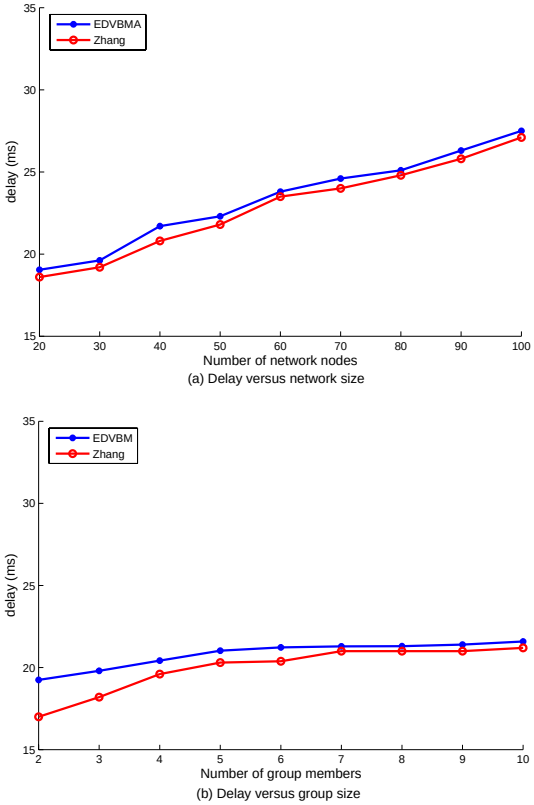

Fig. 7. A comparison on the end-to-end delay of the EDVBM and Zang's algorithm.

nodes ranging from 20 to 100. In fig. 7(b), delay vs. group size has been plotted.

Our algorithm shows the increase in delay than that of Zhang's algorithm. This is because our algorithm selects the appropriate nodes to construct an economic multicast tree keeping the delay within the bound.

In Fig. 8(a) and (b), we have shown the delay-variation of the multicast tree for network size and group size respectively. Our algorithm shows the increase in delay-variation than that of Zhang's algorithm because our algorithm selects the nodes to construct an economic multicast tree. The Fig. 8(b) shows the increase in delay-variation performance as the group size increases. This happens because, larger the size of the multicast group, the larger number of the destination nodes physically closer or farther to the source, which results in increase of the delay-variation between the destination nodes.

VII. CONCLUSION

In this paper, we presented a heuristic approach and proposed a new algorithm EDVBM to construct an economic multicast tree that satisfies the delay and delay-variation constraints. Our algorithm constructs the multicast tree node by node choosing the best node after each iteration. Since it keeps all the nodes generated in a priority queue and searches the graph heuristically, it definitely finds a multicast tree if one exists. Our algorithm calculates the weight() function considering cost, delay and delay-variation. So our tree is more cost-sensitive than the tree generated by Zhang's algorithm. The analysis of results presented in this paper reveal that the cost of the multicast tree generated by our algorithm is less than the tree generated by Zhang's algorithm. The delay and delay-variation is found to be less in Zhang's algorithm as compared to our algorithm. However, as the finding of a cost-sensitive multicast tree satisfying the delay and delay-variation bound is important, the proposed algorithm EDVBM takes the same into consideration.

The time complexity of our algorithm is found to be $O(l_{max}|V|log_2|V|)$ whereas the time complexity of Zhang's algorithm is $O(k^2m^2nlogk)$. The time complexity analysis establishes the proposed algorithm EDVBM to be more efficient than that of Zhang.

REFERENCES

[1] D. Kosiur, *IP Multicasting: The Complete Guide to Interactive Corporate Networks*, Wiley, New York,1998.
[2] Z. Kun, W. Heng, L. Feng-Yu, Distributed Multicast Routing for delay delay variation-bounded Steiner tree using Simulated Annealing, *Computer Communications*, 28, 2005, pp. 1356-1370.
[3] M. R .Kabat, S. K. Mohanty, R. Mall, C. R. Tripathy, An Efficient Heuristic Multicast QoS-Routing Algorithm for Real-Time Applications, *International Journal on Information Processing*, 1(1), 2007, pp. 22-29.
[4] Z.F. Jia, P. Varaiya, Heuristic methods for delay constrained least cost routing using k-shortest paths, *INFOCOM* 2001.
[5] V.P. Kompella, J.C.Pasquale, G.C.Polyzos, Multicast routing for multimedia communication. *IEEE/ACM Transaction on Networking*, 1(3), 1993, pp. 286-292.

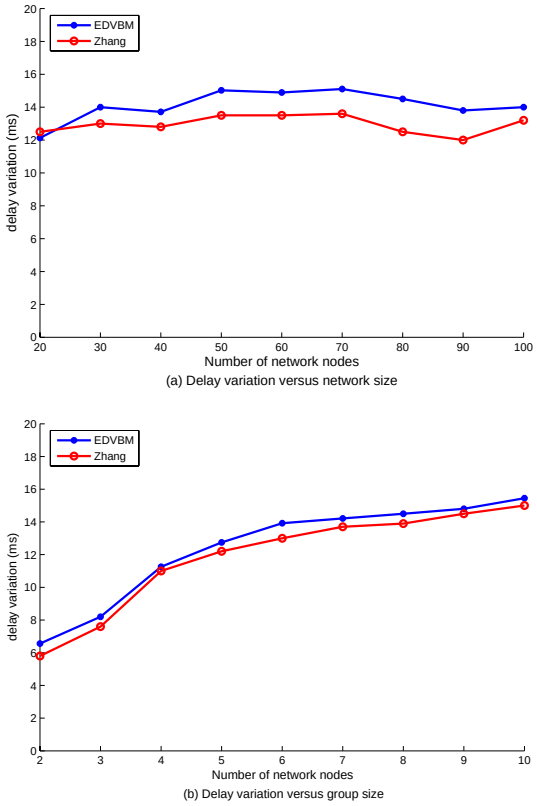

Fig. 8. A comparison on the delay-variation of the EDVBM and Zang's algorithm.

[6] Q. Zhu, M. Parsa, J. Garcia, A source-based algorithm for delay-constrained minimum cost multicasting, *Proceedings of IEEE INFOCOM*, 95, 1995, pp. 353-360.
[7] R. Widyono, The design and evaluation of routing algorithm for real-time channels. Tech report icsi tr-94-024, University of Calfornia at Berkeley, International Computer Science Institute, June1994.
[8] H.Youssef, A. Al-Mulhem, S.M.Sait, M.A. Tahir, QoS-driven multicast tree generation using Tabu search, *Computer Communication*, 25, 2002, pp. 1140-1149.
[9] H.F. Salama, D.S. Reeves, Y. Viniotis, Evaluation of multicast routing algorithms for real-time communication on high-speed networks, *IEEE Journal on Selected Areas in Communications*, 15(3), 1997, pp. 332-345.
[10] E.W. Dijkstra, A note on two problems in connection with graphs, *Numer. Math.*, I, 1959, 269-271.
[11] S.Rampal, D Reeves, An evaluation of routing and admission control algorithms for multimedia traffic. *Computer communication*, 18(10), 1995, pp. 755-768.
[12] G.N.Rouskas, I. Baldine, Multicast routing with end-to-end delay and delay variation constraints, *IEEE J. Selected Areas Communication*, 15(3), 1997, pp. 346-356.
[13] P.R.Sheu, S.T.Chen, On the hardness of approximating the delay variation constraint multicast trees, *Proceedings of the 1999 National Computer Symposium*, Taipci, Taiwan, ROC, pp. A351-A358.
[14] S.T.Chen, A Study on multicast routing with delay variation constraints, *Master Thesis, Department of Electrical Engg. National Yunlin University of Science & Technology*, Taiwan, ROC, June 1998.
[15] P.R.Sheu, S.T.Chen, A Fast and Efficient heuristic algorithm for the delay- and delay variation-bounded multicast tree problem, *Computer Communication*, 25, 2002, pp. 825-833.
[16] C. P. Low, Y.J. Lee, Distributed Multicast Routing with end-to-end delay and delay variation constraints, *Computer Communication*, 23 (9), 2000, pp. 848-862.
[17] T. Pusateri, Distance Vector Routing Protocol, draft-etf-idmr-dvmrp-v3-07, 1998.

Manas Ranjan Kabat has completed his B.E in Electrical Engineering and M.E in Info. Tech. and Comp. Engg. from Utkal university and Bengal Engineering College, India. He has submitted his Ph.D thesis to Sambalpur University, India. Currently, he is working as Senior Lecturer in Veer Surendra Sai University of Technology, India. He has published about 8 research papers in various international journals and conferences. His research area includes QoS routing, Multicasting and algorithms.

Manoj Kumar Patel has completed his M.Sc in Mathematics and MCA from Sambalpur University and IGNOU, India. He is currently working as a Technical Asst. Gr-I(Computer) in the department of Computer Science & Engineering of Veer Surendra Sai University of Technology, India. He is persuing Ph.D under Sambalpur University, India. His research area includes QoS routing and soft computing techniques.

Chita Ranjan Tripathy has coompleted his M.Tech and Ph.D from Indian Institute of Technology, Kharagpur, India. He is currently working as Professor in Computer Science and Engineering and Dean Academic Affairs, Veer Surendra Sai University of Technology, India. He has published more than 70 research papers in different International journals and conferences. His research area includes Computer networks, Parallel processing and reliability. He is a life member of IE, ISTE and OITS.