Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 10 (2012) 161 – 169

Procedia Computer Science

The 3rd International Conference on Ambient Systems, Networks and Technologies (ANT)

New strategies and approaches for efficient overlay multicast routing

Adel Ben Mnaouer[a], Abdelfettah Belghith[b], Mohamed Aissa[c]

[a] *Dar Al-Uloom University,Ryadh, Saudi Arabia*
[b] *University of Manouba, Tunis, Tunisia*
[c] *Universty of Nizwa, Nizwa, Sultanate of Oman*

Abstract

In many group-oriented media streaming applications, overlay multicast (also called Application-layer multicast) is utilized as an alternative technology for IP multicast, which suffers from various issues such as group management, congestion and flow control, and security. Recent researches on overlay networks have revealed that user-perceived network performance, such as end-to-end delay performance, could be improved by an overlay routing mechanism. However, these studies only consider end-to-end delay, or only bandwidth, and there are few works focusing on delay variation constraint. We proposed an algorithm, which defines an optimal balance tree to optimize a trade-off between delay and bandwidth consumption with constraints on both delay and delay variation. Furthermore, we introduce a new Delay–Variation Estimation Scheme and core selection strategies for multicast routing in leased overlay networks.

Keywords: multicast overlay routing algorithms, end-to-end quality of service, delay and delay-variation constrained routing, core selection

1. Introduction

IP multicast service has been widely accepted as the way to implement multicast for the past decade. However, despite the conceptual simplicity of IP multicast and its obvious benefits, it is yet to take off. As an alternative to IP multicast, *Overlay Multicast* has recently received more attention [1]. Providing the required quality of service for Peer-to-peer and many multimedia applications over a packet switching network has been a critical task for a long time. A recent approach to providing QoS without changing the

* Corresponding author: Abdelfettah Belghith
E-mail address: abdelfattah.belghith@ensi.rnu.tn or abdelfettah.belghith@ieee.org

doi:10.1016/j.procs.2012.06.024

network architecture is based on the overlay network, an application-layer logical network created on top of the physical network. It is formed by all, or a subset, of the underlying physical nodes. The connection between each pair of overlay nodes is provided by overlay links, which consist of paths composed of underlying physical links. In an overlay network, each individual logical link that connects two nodes can contain several routers and hosts in the underlying physical network. Overlay networks provide a flexible approach for applications.

Certain collaborative applications, such as video streaming applications, real-time multiplayer games, require the overlay architectures to provide certain quality of services (bandwidth, delay bound, stability ...). To construct such an overlay network with quality of service guarantees, service providers usually distribute a set of MSNs (Multicast Network Node) over the Internet and lease links (virtual path or virtual leased link) between two MSNs. We refer this kind of overlay architectures as virtual private (personal) leased overlay networks (VPON). *Delay* is a significant factor in collaborative applications in a leased overlay network and is taken as a constrained metric of the presented algorithm. In addition, *end-to-end delay* is definitely used rather than average delay or total of the whole tree, because each user is mostly concerned to receive information from the source as soon as possible. Besides, *inter-destination delay variation* is paid attention in this paper as well. It is necessary that every participant to receive information from the source at the same time so that the fairness is guaranteed. There are several situations in which we need to limit the variation among the path delays by a certain given maximum bound. During a teleconference, it is important that a speaker is heard by all participants at the same time; otherwise, the communication may lack the feeling of an interactive face-to-face discussion [2].

In [3], the authors have considered the problem of determining a multicasting sub-network with an end-to-end delay bound and with a tight delay variation for multimedia applications on overlay network. Then they have presented an algorithm Chain, which (as they declared) achieves the tightest delay variation for a given delay bound. However, again in their heuristic, the delay variation was only minimized and not constrained.

In [4], Sheu and Chen studied the problem of minimizing multicast delay variation under the multicast end-to-end delay constraint. As a result, they have proposed the Delay and Delay Variation Constraint Algorithm (DDVCA) that was derived from the Core Based Tree (CBT) [5] and the minimum path algorithm [6]. It has a complexity of $O(|E||V|^2)$ (where V is the set of vertices and *E represent* the set of edges in a graph). When several nodes are possible candidates for a core node, the DDVCA chooses one of them randomly. However, in DDVCA, the delay variation was only minimized and not constrained.

In [7], an algorithm with a complexity of $O(|E||V|^2)$ and based on CBT [5] was proposed. It produces multicast trees with low multicast delay variation. The algorithm consists of two parts. In the first part, a core node is selected. In the second part, a multicast tree is constructed. The simulation results show that the proposed scheme obtains a better minimum multicast delay variation than what the DDVCA achieved.

In [17], authors improved greedy based multi-constrained multicast solutions and proposed the ICRA algorithm that improves the well known Mamcra algorithm. In the quest to enhance the execution time, they further proposed a taboo search algorithm coined the Taboo-MQR algorithm. In [16, 18], the authors proposed the mQMA algorithm; a QoS multicast aggregation algorithm which handles multiple additive QoS constraints. mQMA deals with two important problems of traditional IP multicast, i.e., multicast forwarding state scalability and multi-constrained QoS routing. It builds few trees and maintains few forwarding states for the groups thanks to the technique of multicast tree aggregation, which allows several groups to share the same delivery tree. Moreover, the algorithm mQMA builds trees satisfying multiple additive QoS constraints.

The main contribution of our work is the discovery of a simple yet effective heuristic that exhibits very good performance and that can be easily implemented in a wide range of leased overlay networks. Furthermore, we extend the DDVCA, Kim's and Chain algorithms by *(a)* adding the delay-variation constraint (rather than minimizing it), *(b)* introducing a combination of the residual bandwidth and the delay as a supplementary QoS metric, *(c)* proposing an algorithm with lower time complexity and *(c)* introducing a new delay-variation estimation method, which, to the best of our knowledge, is the first of its kind to be used with overlay networks.

In the remainder of this paper, Section 2 will present the multi-constrained overlay network problem definition and we propose a new delay-variation estimation scheme. In Section 3, we propose the core node selection strategies. In Section 4, we propose our Bandwidth Delay and Delay-Variation constrained (*BDDVC*) algorithm. This section also includes some theorems and comparison of our proposed algorithm with some well-known algorithms from the literature. In Section 5, we state the correctness and time complexity analysis of BDDVC algorithm. Finally, Section 6 concludes the paper.

2. The overlay network problem

2.1. The Overlay network model

The routing problem can be considered as QoS multicast problem on the leased overlay network [8]. Given an undirected graph *G=(V, E)* to denote the leased overlay network, with *V* as the set of vertices (end-users, i.e., MSNs in case of proxy based overlay) and *E* as the set of edges (*links*) between the end-users (MSNs). We define the following weight functions on the edges, for any edge $B(e): E \to \mathbb{R}^+$, a positive real edge residual bandwidth function, and $D(e): E \to \mathbb{R}^+$, a positive real delay edge function. The delay of a link is the sum of the perceived queuing, transmission, and propagation delays over that link. The nodes represent routers or switches and edges represent the communication links between them. An edge $e \in E$ from $u \in V$ to $v \in V$ is represented by $e(u,v)$. The bandwidth we are interested in here, is the residual bandwidth (some percentage of bandwidth) that is available (reserved) for a new traffic flow (i.e., QoS flows). We define the bandwidth of a path as the minimum of the residual bandwidth of all links on the path or the bottleneck bandwidth. The delay and the bandwidth of the unique path $P(s,v)$ in T considered from s to v are defined as follows:

$$Delay[s,v] = \sum_{e \in P(s,v)} D(e) \tag{1}$$

$$BW(e) = min_{e \in P(s,v)} B(e), \quad \text{where } P(s,v) \text{ is the unique path considered from } s \text{ to } v. \tag{2}$$

On this graph, we designate a source node $s \in V$ and a set of destination nodes *Z*, called the multicast group members such that $Z \subseteq V - \{s\}$.

2.2. Our analytical model

One of the principal resources that an overlay network must manage is its residual bandwidth. This residual bandwidth represents a major cost, and is typically the resource that constrains the number of simultaneous multicast sessions that an overlay network can support. Hence, the routing algorithms used by an overlay multicast network, should seek to optimize its use. Additionally, a multicast routing algorithm should ensure that the routes selected for multicast sessions do not contain excessively long paths; as such paths can lead to excessively long packet delays. However, the objective of limiting delay in a multicast network can conflict with the objective of optimizing the interface bandwidth usage, so multicast routing algorithms must strike an appropriate balance between these two objectives. To maintain this balance, and based on the observations and suggestions proposed in [9], we propose a link cost function capturing the tradeoff between residual bandwidth minimization and the risk level due to the delay constraint. That is, it takes into account *simultaneously* the residual bandwidth and the delay. For that purpose, we define the cost of a link $e = (v_i, v_j)$ as follows:

$$C(e) = \frac{D(e)}{B(e)} \tag{3}$$

where *D(e)* and *B(e)* (as defined before), the delay and the residual bandwidth of a link respectively. The total cost of the shortest path $P(s, v_i)$ connecting the source node with a node v_i is calculated as follows:

$$Cost\big(P(s, v_i)\big) = Cost[s, v_i] = \sum_{e \in P_T(s,v_i)} C(e) = \sum_{e \in P_T(s,v_i)} \frac{D(e)}{B(e)} \qquad (4)$$

In all the following, we mean by *LDBT*, the *Least Delay Bandwidth Tree,* obtained at the beginning of our proposed algorithm (Algorithm 1) and using Dijkstra's algorithm [10].

2.3. Delay and delay-variation-constrained overlay model

Given an overlay network *G=(V,E)*, a source node $s \in V$, a multicast group $Z \subseteq V - \{s\}$, a link delay function *D*, a delay constraint (delay bound) *Δ*, a delay variation tolerance *δ*, and an overall *Cost,* the residual bandwidth, delay, and delay variation-bounded overlay routing problem can be stated as follows: Find a multicast sub-network $T = (V_T, V_E)$ $(T \subseteq G)$ rooted at *s* and spanning all nodes in *Z*, such that for each node v_i in *Z:*

$$Delay[s, v_i] = \sum_{e \in P_T(s,v_i)} D(e) \le \Delta, \quad \forall v_i \in Z \qquad (5)$$

$$\left| \sum_{e \in P_T(s,v_i)} D(e) - \sum_{e \in P_T(s,v_j)} D(e) \right| \le \delta, \quad \forall v_i, v_j \in Z \qquad (6)$$

$$Cost(T) = \sum_{e \in P_T} C(e) \ is\ minimized \qquad (7)$$

Here $P_T(s,u)$ is the unique path in the tree *T* considered from the source *s* to a destination node (v). We replace (6) by a simpler formula as proposed in [11]:

$$\left| Avdelay(T) - \textstyle\sum_{e \in P_T(s,v_i)} D(e) \right| \le \delta, \quad \forall v_i \in Z \qquad (8)$$

Where $Avdelay(T)$ – the tree average delay calculated as follows:

$$AvDelay(T) = \frac{\sum_{v_i \in Z} Delay[s, v_i]}{|Z|}, \quad \forall v_i \in Z \qquad (9)$$

The DVBMT (Delay and Delay Variation-Bounded Multicast Tree) problem based on (5) and (6) was proved to be NP-Complete [2] whenever the size of the multicast group $m = |Z|$ is greater than 2. Consequently, our problem is also NP-Complete. In the remainder, we will call a tree (path) that satisfies both constraints (5) and ((6) or (8)) *a feasible tree (path).*

2.4. Delay-variation estimation method and feasible path analysis

The equations (5) and (6) have two conflicting objectives [2]. The delay constraint (5) dictates that short paths must be used. However, choosing short paths may lead to a violation of the delay variation constraint (6) among nodes which are close to the source and nodes which are far away from it. Consequently, it may be necessary to select longer paths for some nodes in order to satisfy the latter constraint. A balance must be struck between the two constraints [2]. In [2, 3, 8], to solve this problem, multiple paths between the source and each destination are pre-computed and then one of those that satisfy the QoS requirements is selected. This exhibits higher computation cost and execution time. In [12], we solved this problem by proposing instead of (9) the following new tree average delay formula:

$$AvDelay(T) = \begin{cases} \Delta, & if\ Delay[v_i] \le \delta, \ \forall v_i \in Z \\ \frac{\sum_{v_i \in Z} Delay[s, v_i]}{|Z|}, & otherwise \end{cases} \qquad (10)$$

In [12], we stated and proved the following formulas observed in a feasible tree:

$$0 < \delta < \Delta; \qquad Delay[s, v_i] > \delta, \ \forall \ v_i \in Z; \qquad \delta < AvDelay(T) \leq \Delta \tag{11}$$

Our contribution is *to assign our results (11) to overlay networks* and *to extend them* with the following two theorems and new tree's delay-variation estimator. As it is seen from (11), a destination node v_i having $Delay[s, v_i] < \delta$ cannot participate in the construction process of a feasible tree.

Theorem 1

In a feasible tree, the tree average delay proposed in [11, 12] is restricted by the following interval:

$$\Delta_{min} \leq AvDelay(T) \leq \Delta_{max} \tag{12}$$

Where Δ_{min} and Δ_{max} are the minimum and maximum delay bound respectively in the *LDBT*.

Theorem 2

After the construction of the *LDBT*, the calculated maximum tree delay variation tolerance δ_T (δ_{max}) can be deduced by the following formula:

$$\delta_T = \delta_{max} = \Delta_{max} - \Delta_{min} \tag{13}$$

Table 1. Core node selection strategies

Strategy	Description	Formula
The core node should be a multicast node	*Motivated by the simulation results provided in* [14], we adopt a strategy dictating that core candidates are restricted to be *multicast group members*.	$Core\ node\ v_i \in Z$ (14)
The core node should be a QoS parameter center	Rather than adopting a strategy similar to "*Topological Center of Z in Z"* [14], which is difficult to maintain in networks where their topology change dynamically, we adopt a different strategy based on *QoS aware member center*. That is, we select a core node among destination nodes having the least $Cost$. That is, by this value, it is situated the nearest to all the remaining destination nodes in the *LBDT*. It is as if it is situated at the *center of the remaining destination nodes.*	$Cmp(v_i) = \frac{\sum_{j=1, j\neq i}^{\lvert Z\rvert-1} C(v_i, v_j)}{\lvert Z\rvert}, \ \forall \ v_i, v_j \in Z, \ v_i \neq v_j$ (15)
The core node should be a favourite destination node	To reduce computational overhead, the center member should have the maximum degree composed of destination nodes. We call such a destination node "*a favourite destination node*". The *destination degree* of a destination node v_i, denoted by $degd(v_i)$, in an undirected graph *G*, is the number of edges *containing destination nodes* incident with it. Consequently, the degree of a destination node v_i, denoted by $deg(v_i)$, in an undirected graph *G* is the sum of number of relay edges and edges *containing destination nodes* incident with it.	$Core\ node\ v_i \in max\{degd(v_j)\}, \ \forall v_i, v_j \in Z$ (16)

Our contribution is to propose a new delay-variation estimator for overlay networks. Based on theorem 1, we introduce in overlay networks a simple new delay variation estimator based only on the minimum and maximum delay bounds calculated in *LDBT* and deduced as follows:

$$AvDelay(T) = \begin{cases} \Delta_{max}, & if\ Delay[v_i] \leq \delta, \ \forall v_i \in Z \\ \Delta_{aver} = \frac{\Delta_{max} + \Delta_{min}}{2}, & otherwise \end{cases} \tag{17}$$

Consequently $AvDelay(T)$ in (8) will be replaced by (17). Comparing (17) with (10), we observe that (17) is simpler and attractive as it requires only two parameters Δ_{max} and Δ_{min}, which reduces computational overhead.

3. Core node selection strategies

We denote *a core-selection algorithm* as *delay-bounded,* if the algorithm considers a given delay-bound for the group during the selection process, and the resulting core is such that a path exists between each source-receiver pair in the group *which passes through this core* without violating the delay-bound

[13]. Furthermore, we denote *a core-selection algorithm* as *delay-variation bounded,* if the algorithm considers a given delay variation tolerance for the group during the selection process, and the resulting core is such that the difference between the end-to-end delays along the paths from the source s to any two- destination nodes, which passes through this core, satisfies the delay variation tolerance [13]. Our contribution is to propose core node selection strategies and criteria in an overlay network (Table 1).

To improve the center member parameter (15), we integrate (16) in (15). The idea behind this integration is to give priority to a favourite destination node to participate in the core selection process. The advantage of such selection is to reduce the computational overhead and to alleviate the traffic through relay nodes and orient it to *a favourite destination center member node*. If two destination nodes have the same $Cmp(v_i)'$ value, then the priority is given to the destination node with the biggest $degd(v_j)$. *We denote a core node verifying (18) a favourite destination center member node.*

$$Cmp(v_i)' = \frac{1}{degd(v_i)} \times \frac{\sum_{\substack{j=1 \\ j \neq i}}^{|z|-1} C(v_i, v_j)}{|Z|}, where\ \ v_i, v_j \in Z\ \ and\ \ v_i \neq v_j \tag{18}$$

To avoid message retransmission and alleviate the network traffic, we adopt a strategy based on the hypothesis that if a message passes through a destination node first, then it is received immediately by this node.

Algorithm 1. Our BDDVC Algorithm

Input: *a computer network G=(V,E), a set of destination nodes M, a source node s, an upper bound Δ of end-to-end delay.* δ *- Delay variation tolerance*

Output: *a delay and delay-variation bounded sub-network.*

Begin

Cost[s]=0, Delay[s]=0; $T = \emptyset, Q \leftarrow \emptyset, v_c \leftarrow \emptyset$

For *each vertex* $u \in V - \{s\}$ **do**

$\{Cost[u] = \infty, Delay[u] = \infty, Centerp(v_i)' = \infty\}$

Call Dijkstra's algorithm to compute the least Bandwidth Delay Tree (*LDBT*) . Find out $\Delta_{\max}$ and $\delta_{\max}$.

If $\Delta > \Delta_{max}$ and .$\delta > \delta_{max}$, then relax one or both input data

/ For this LDBT, compute $C(v_i, v_j)$ of all LDBT paths connecting each destination node v_i with another destination v_j. Then calculate for each destination node v_i its corresponding $Cmp(v_i)'$ */*

For each $v_i \in Z$ **do**

{

For each $v_j \in Z$ **do** { $Cost[v_i] \leftarrow Cost[v_i] + C(v_i, v_j)$}

$Cmp(v_i)' = \frac{1}{degd(v_i)} \times (Cost[v_i]/|Z|)$

$Q \leftarrow Cmp[(v_i])'$;

}

Sort *Q* in *an increasing order*

/ Center member selection process */*

For each $v_i \in Q$ **do**

{ **For** each $v_j \in Z$ **do**

. { $Delay[s, v_j] \leftarrow Delay[s, v_i] + Delay[v_i, v_j]$;

if $Delay[s, v_j] < \Delta$ *and* $|AvDelay(T) - Delay[s, v_j]| < \delta$

Flag =1; else Flag=0; endif; next v_j*;* }

If flag =1 { $v_c = v_i$; *Exit;* } *else next* v_i

*/*if the first destination node verifying both constraints is considered core node and no need to consider the remaining dest. nodes in Q */*

} */* External For loop end */*

if $v_c = \emptyset$ v_c *=s /* The source node is selected as a center member node*/*

If $v_c \neq s$ $T = T \cup \{L / L \in$ minimum *Cost* path from s to v_c

For each $v_j \in Z$, $v_j \notin path(s, v_c)$ **do**

$\{T = T \cup \{L / L \in$ minimum *Cost* path from v_j to v_c

else Call Dijkstra's algorithm to compute the delay bounded and delay-variation constrained multicast tree spanning M $\cup$ $\{s\}$ and rooted at *s*.

Return *T*; **End** (of the Algorithm)

4. The BDDVC algorithm

Similar to DDVCA [4] and Kim's algorithms [7], our BDDVC algorithm (Algorithm 1) basically comes from CBT [5], and the Dijkstra's shortest path algorithm [10]. CBT establish a multicast tree by choosing some Core Routers, which compose the Core Backbone. Afterwards, all node operations relating to join and leave the multicast group are based on issuing a request toward an appropriate Core Router. In our BDDVC algorithm (as in DDVCA), we select a Core Router addressed as a central node.

4.1. BDDVC algorithm description

Our BDDVC algorithm contains five stages. The first stage (lines 2-4) is the initialization. The second one (line 5), during which the Least Bandwidth Delay Tree ($LBDT$) is computed by using Dijkstra's algorithm [10]. Subsequently, the user input data are verified. If these data are too tight, then they are relaxed (line 6). The third stage (lines 7-12) is the computation of Center Member Parameter in order to form an ordered set of candidate center members. In this phase, $Cmp(v_i)'$ is calculated for every destination node v_i and then introduced in a priority queue Q (line 11). Subsequently, Q is sorted in an increasing order (line 13) such that the first node in Q has the least $Cmp(v_i)'$ and therefore, has the highest priority. The center member selection constitutes the fourth stage algorithm execution (lines 14-20). At this stage, we test whether for the picked node from Q, the shortest path from the source to any destination node passing through this picked node satisfies the delay bound Δ and the delay variation tolerance δ. If it does, then it is selected as a center member ($v_c = v_i$) (line 19). Therefore, there is no need to treat the other remaining nodes in Q. Otherwise, we pick from Q the next candidate (line 14) and the same constraints (5) and (8) are tested. If all nodes in Q are treated and no one verifies (5) and (8) then the source s is considered as the only candidate ($v_c = s$) (line 21). The fifth algorithm phase (lines 22-25) represents the multicast tree construction process. We first connect the source node with the center member v_c (line 22), and then we connect to this center member all the remaining destination nodes (lines 23-24). If the source is selected as a center member, we apply Dijkstra's Bandwidth-Delay shortest path algorithm to compute the delay and delay-variation bounded multicast tree rooted at the source s and spanning all destination nodes (line 25).

4.2. BDDVC algorithm operations

A detailed example in Fig. 1 is provided to show how our BDDVC algorithm works on the original graph depicted in Fig. 1(a). [3, 4] used the same graph with the same settings but without residual bandwidths and delay variation constraint. We applied Kim's Algorithm [7] on this graph. Our resulting multicast tree is shown in Fig. 1(c). In the original graph, each pair *b/d* of numbers along any edge, represent the residual bandwidth *(b)* and delay *(d)* for that edge. s is set to be the source node. The delay bound Δ is set to **60** (as in [4]), the delay-variation tolerance δ is set to **26** (our input data), and the set of destination nodes Z is set to: Z=*{B,E,H}*. The resulting tree is shown in Fig.2 (c).

4.3. Comparison with other algorithms

In Table 2, we compare the execution of the mentioned algorithms on the original graph depicted in Fig. 1 (a). The delay bound Δ is set to 60, the delay-variation tolerance δ is set to 26, and the set of destination nodes Z is set to: *Z={B,H,E}*. In this table, we calculate the delay variation between every pair of destination nodes using (6). Then the maximum delay variation tolerance δ_T is fixed and calculated for destination nodes Z is set to: *Z={B,H,E}*. In this table, we calculate the delay variation between every pair of destination nodes using (6). Then the maximum delay variation tolerance δ_T is fixed and calculated for every tree as follows: $\delta_T = max\{\delta_{BE}, \delta_{BH}, \delta_{EH}\}$. It is to be noticed that the tree constructed by our BDDVC algorithm is similar to that constructed by the Kim's algorithm.

5. Ccorrectness proof and time complexity analysis of the BDDVC algorithm

The correctness and time complexity of the algorithm BDDVC results from the following theorems. The proofs are omitted for lack of space.

Theorem 1: The algorithm BDDVC always constructs a delay and delay variation-bounded multicast tree if such a tree exists.

Theorem 2: The time complexity of BDDVC is $O(|E||V|)$.

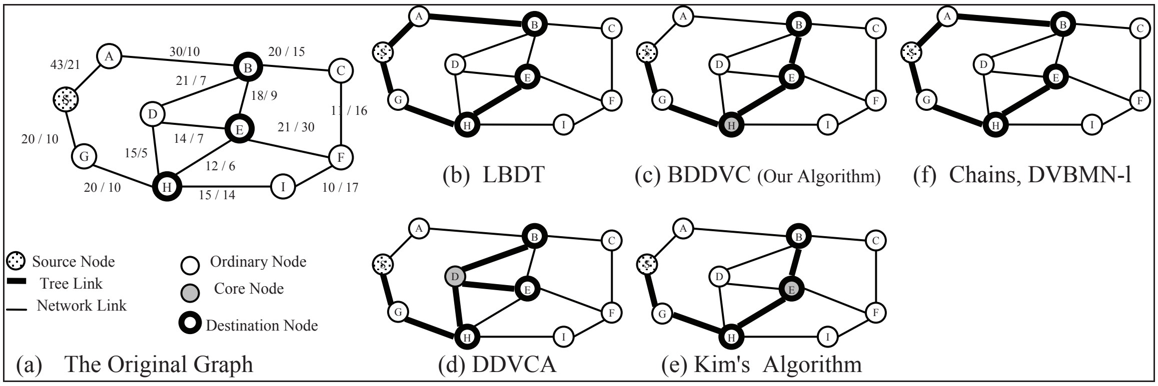

Fig. 1. Comparison between BDDVC and other algorithms

Table 2. Algorithm complexities and Comparison between BDDVC and other algorithms

Algorithm	Total Delay	δ_{BH}	δ_{BE}	δ_{EH}	δ_T	Total Delay	Time complexities
DDVCA [4]	84	12	0	12	12	84	$O(\|E\|\|V\|^2)$
Kim's Algorithm [7]	81	15	9	6	15	81	$O(\|E\|\|V\|^2)$
BDDVC	84	12	0	12	84	12	$O(\|E\|\|V\|)$
Chains [3]	77	11	5	6	11	77	$O(\|E\|^2k)$
DVBMN-l [15]	77	11	5	6	11	77	$O(\|E\| + \|V\|klog(\|E\|/\|V\|) + \|Z\|klog\|Z\|)$

Table 2, proves that our BDDVC algorithm has better complexity than others well-known algorithms to which it is compared. In this Table, *E, V* and *Z* are as mentioned before, *k*-number of shortest paths.

6. Conclusion

In this paper, we considered the problem of generating minimum Bandwidth-Delay multicast trees that satisfy certain bounds on the end-to-end delay from the source to the destination nodes and the inter-destination delay variations between paths from the source to the destination nodes. These constraints are imposed by the user process. Therefore, based on the combination of CBT and the Dijkstra's shortest path algorithm, we proposed BDDVC with much lower time complexity ($O(|E||V|)$ than DDVCA and Chains. Furthermore, we extend DDVCA and Kim's algorithms by adding the residual bandwidth metric and the delay-variation constraint. Besides, we introduced in overlay networks a new delay-variation estimator, and new strategies in core selection method and in shortest path minimization relaying on using simultaneously both residual bandwidth and delay in these two processes. Thus, our algorithm produces paths and trees, which are stable, less risky and suitable for various overlay network conditions.

References

[1] Zhang W, et al. An overlay multicast protocol for live streaming and delay-guaranteed interactive media. *Journal of Network and Computer Applications (2011)*, doi:10.1016/j.jnca.2011.02.013

[2] G. Rouskas and I. Baldine. Multicast routing with End-To-End Delay and Delay Variation Constraints. *IEEE Journal on Selected Areas in Communications*, 15(3), April 1997.

[3] Shankar M. Banik, Sridhar Radhakrishnan, Chandra N. Sekharan. Multicast Routing with Delay and Delay Variation Constraints for Collaborative Applications on Overlay Networks. *IEEE Transactions on Parallel and Distributed Systems, Volume 18 , Issue 3* (March 2007) Pages: 421-431

[4] Pi-Rong Sheu, Shan-Tai Chen**.** A fast and efficient heuristic algorithm for the delay- and delay variation-bounded multicast tree problem. *Computer Communications*, Volume 25, Issue 8, 15 May 2002, pp. 825-833.

[5] A. Ballardie. Core Based Tree (CBT) multicast routing architecture. *Internet RFC 2201*, Sept. 1996.

[6] R. Prim. Shortest ConnectionNetworks and Some Generalizations. *Bell Systems Tech. J., vol. 36*, pp.1389-1401, 1957.

[7] M. Kim, Y.C. Bang and H. Choo. On core selection algorithm for reducing delay variation of many-to-many multicasts with delay-bounds. *NETWORKING 2004*: pp. 200-210.

[8] X. Zhang, G. Zhang. Load balance Multicast routing Algorithms on Leased Overlay Network. *IEEE* 2006.

[9] S.-J. Lee, S. Banerjee, P. Sharma, P. Yalagandula, and S. Basu. Bandwidth-aware routing in overlay networks. *INFOCOM2008*, 2008.

[10] E. W. Dijkstra. Note on Two Problems in Connection With Graphs. *Numerische Mathematik,* 1 (1959), pp. 269-271.

[11] Hun-Young Lee, Chan-Hyun Youn. Scalable Multicast Routing Algorithm for Delay-variation Constrained Minimum cost Tree. *Proc. of the IEEE ICC2000 conference*, June 2000, New Orleans, USA.

[12] Mohamed Aissa et Al. A new scalable multicast routing algorithm for interactive real-time applications. *The International Journal of Personnal and Ubiquitous Computing, Springer,* Volume 15, Issue 8 (2011), Page 833-844.

[13] A. Karaman, H. Hassanein. Core-selection algorithms in multicast routing – comparative and complexity analysis. *Computer Communications* (2005), pp.1-17.

[14] E. Fleury, Y. Huang, P. K. McKinley. On the Performance and Feasibility of Multicast Core Selection Heuristics. *Networks*, vol. 35, no. 2, pp. 145-156, John Wiley, 2000.

[15] A. Bhattacharya, Z. Yang. DVBMN-l: Delay Variation Bounded Multicast Network with Multiple Paths. *Proc. of the ICME 2009 Conference*, June 2009, New York, USA.

[16] N. BenAli, J. Moulierac, A. Belghith and M. Molnar. mQMA: Multi-Constrained QoS Multicast Aggregation. IEEE Globecom '07, Washington D.C., USA, 26-30 November, 2007.

[17] N. BenAli, M. Molnar and A. Belghith. Multi-constrained QoS multicast routing optimization. INRIA technical report no. 1882, IRISA, February 2008.

[18] N. BenAli, A. Belghith and M. Molnar. QoS Multicast aggregation under multiple additive constraints. Computer Communications, vol. 31, no. 15, pp. 3564-3578, Elsevier, Septembre 2008.