# MESH-PULL BACKUP PARENT POOLS FOR VIDEO-ON-DEMAND MULTICAST TREES

Müge Sayıt[1] and E. Turhan Tunalı[2]

[1]International Computer Institute, Ege University, Izmir, Turkey
[2] Department of Computer Engineering, Izmir University of Economics, Izmir, Turkey

***ABSTRACT***

*Resilient multicast is a challenging issue for overlay trees particularly because of high churn. In this work, we propose a mechanism that allows scalable video multicast. While the regular operation involves tree-push of the video, any node that looses its parent on the tree solicits video from a predetermined backup set of nodes in a mesh-pull fashion. The main idea is to allocate less bandwidth for backup to improve bandwidth utilization while maintaining the best possible video quality. The choice of essential design parameters are studied together with seamlessness of the streaming under variety of fault scenarios. Simulation results indicate the optimality of the proposed approach as far as resiliency, bandwidth utilization, delay and video quality are concerned.*

***KEYWORDS***

*multicast trees, resiliency technique, mesh-pull backup, scalable codec*

## 1. INTRODUCTION

In peer to peer (P2P) communications, it is well known that tree-push type of approaches provide smaller overhead in utilization of network resources when there is no failure, but fail to provide resilience unless a considerable bandwidth is reserved idle for back-up [1]. Mesh-pull type of approaches [2], [3], [4], [5], [6], [7], [8], [9] introduced efficient solutions for resilience while introducing overhead network traffic due to excessive control messaging and/or redundant data packets. Since churn is high, node failure rate is considerably high in peer to peer networks. Hence tree based approach needs to consider this fact.

To provide a solution to the problem, backup parent concept was proposed together with degree constrained optimal tree [10], [11]. In case of parent failure, a tree node connects to a pre-determined backup parent that has adequate idle bandwidth. By this way, the tree is repaired and multicast can resume. Nomination of backup parent in advance prevents waste of time for backup search. This is classified as a "proactive strategy" in the literature [10], [11]. The problem with this approach is that the nodes should reserve idle bandwidth that can stream video with a predetermined rate. This, in turn, reduces bandwidth utilization. In order to increase bandwidth utilization, backup slots are reserved in accordance with video layers where scalable video is streamed in [12].

On the other hand, although mesh based systems are resilient to peer churn but they suffer playback lag among peers. The nodes having large playback lag with other nodes in the system may not use their upload bandwidth [13] and this reduces bandwidth utilization of the system.

DOI : 10.5121/ijcnc.2014.6203 

In this paper, we propose a new hybrid strategy in which regular streaming involves multicast through the nodes of a pre-constructed tree whereas streaming from backup involves a mesh-pull type of mechanism that does not require reservation of hefty idle bandwidth for each connection. Best effort is spent to provide high quality video when failure occurs.

The contributions of our work are as follows: First, we present a novel backup strategy that involves a hybrid mechanism suitable for scalable video. Second, we propose a new adaptive hybrid push-pull P2P video-on-demand streaming system. After multicast tree is constructed, this system having the tree push based structure is converted into the mesh based structure if node failure rate increases. By this adaptive approach, we may keep advantages of both tree push and mesh pull systems. Our proposed system is resilient to peer churn while the playback lag between peers is small. We present a simulation-based comparative evaluation against tree-push only strategy in which full bandwidth channel reservation is made for possible disconnected nodes.
The rest of the paper is organized as follows. In section two, related work on multicast trees together with error resilience problem is addressed. In section three, the proposed hybrid method for resilience together with its algorithms is given. In section four, simulation results that compare the proposed method with non-hybrid case are given. Finally, in section five, concluding remarks are made.

## 2. RELATED WORK

The error resilience problem in multicast trees has been addressed in many of the studies including [5], [12], [14], [15], [16]. [12] addresses the real-time issues in resilience, [14] and [15] discuss the optimal tree formation problem, [5], [16] discuss scalability and bandwidth utilization issues associated to resilience. [17] addresses recovery time and associated buffer requirement problem.

Resilience technique on multicast tree concentrates on repairing the tree in case failure happens. In [5] and [18], nodes are clustered according to the delay values between the nodes and cluster leaders selected from each cluster are responsible to convey video data to their clusters. If these relay nodes leave the system, a new cluster leader is selected and hence tree is repaired. Since selecting a new cluster leader after the failure of the cluster leader may cause interrupt of display, selecting backup nodes before failure is proposed in the literature. For this purpose, some backup mechanism is utilized by considering the capacity of each node. In [10], [19] and [20], each node reserves a number of backup slots, each slot being able to stream baseline video and nodes in the tree select their backup parents according to available backup capacity and RTT value between candidate backup parent and themselves. Unfortunately, a portion of the backup capacity reserved stays idle and bandwidth utilization reduces in the overall scheme. In simulation studies, it has been observed that half of the available bandwidth should be reserved for backup to provide acceptable resilience [21].

In order to make use of advantages of both tree based and mesh based systems, many hybrid based systems are proposed in the literature [22], [23], [24]. Most hybrid systems construct more than one multicast tree [22] and send different parts of the video data over those trees [25]. In figure 1, an example overlay architecture used in such systems is illustrated. In the figure, source node splits the video up to two streams called stream 1 and stream 2. The packets belonging to different streams follow different paths on the tree. A node can be an interior node in one tree and can be a leaf node in another tree as it can be seen from the figure. This structure provides robustness since the nodes still receive the video packets from another tree after experiencing parent failure in one tree.

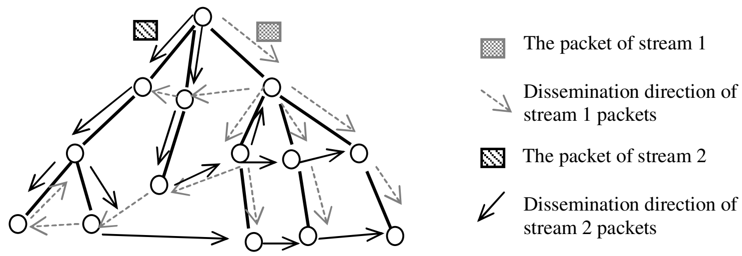



Figure 1. An example of a multitree based hybrid overlay network.

These video partitions can be obtained by using Multiple Description Codec (MDC) [26] and each received description increases the received video quality [27], [28], [29], [30]. However such systems have longer playback delay when compared to tree based systems. In [23], stable nodes construct a tree backbone and unstable nodes connect to the system in a mesh structure. Some hybrid systems utilize cloud or Content Distribution Network (CDN) servers [31], [32], [33], [34]. The main purpose of implementing a cloud based or CDN based service is to recover lost packets by retransmission so that higher QoE is obtained. In this work, we propose a hybrid system that drastically differs from the existing work. The novelty of our system is in its adaptive communication model implemented. Under normal operation, the system operates under tree-push model. As the node failure rate goes up, the tree structure adapts into a mesh based system. The proposed model utilizes the advantages of both tree based and mesh based systems thereby providing low playback delay and robustness.

Our work here targets a new strategy in which less bandwidth is kept idle while maintaining the same resilience level. We introduce new heuristics to modify backup parent pool approach introduced in [12] to maximize resilience of the method of [21]. While [12] repairs the tree with full capacity channel, the proposed approach of this work constructs a mesh of nodes that is formed from members of designated backup parent pool of a particular layer of the tree plus the disconnected node. Therefore, for those sub-trees in which churn has not occurred, the overlay preserves its tree structure with root being the original source whereas roots of disconnected sub-trees pull video from respective mesh and those sub-trees are effectively disconnected from the root. This strategy allows reservation of smaller ratio of bandwidth as backup yet providing higher resilience.

## 3. TREE-PUSH BROADCAST WITH MESH-PULL BACKUP

A variation of algorithm of [12] is used to construct the multicast tree. The algorithm considers variety of factors in attempting to optimize performance. We briefly summarize properties of the algorithm details of which are beyond the scope of this work. The heuristics of the algorithm of [12] spend best effort to

i. Maximize average bandwidth from root to individual nodes
ii. Minimize average delay from root to individual nodes
iii. Maximize bandwidth utilization
iv. Maximize resilience
v. Minimize average node depth

The algorithm of [12] constructs backup pools at each layer of the tree in which one or more nodes are expected reserve some pre-determined portion of its total upload bandwidth for backup.

It has been discussed that the higher (lower) the portion of the bandwidth reserved for backup, the higher the resilience (the higher the bandwidth utilization).

In this work, we aimed at reducing overhead of this reserving backup approach while maintaining a comparable resilience. For this purpose, rather than allocating a complete video bandwidth to a disconnected node, we propose to construct a mesh-pull that is formed from the disconnected node and the backup set of a particular tree layer. By this way, for a particular node, we avoid allocating an integral amount of video bandwidth for backup. As a tradeoff the streaming quality to the disconnected sub-tree may decrease due to delays caused by the mesh-pull.

We will assume that we stream scalable video that consists of a *base* and an *enhancement* layer that are going to be indicated by $b$ and $e$ respectively. Each tree layer will be indicated by $l$ with $l$=0 being the layer of the root. All nodes of each tree layer $l$ will reserve some portion of their total upload bandwidth less than $b$ for backup and they all will take part in backup mesh of tree layer $l$. When a node is disconnected from its parent, the algorithm of figure 2 is run in all nodes to request backup from respective tree layers.

```
given    l, the layer of the node on the broadcast tree
if       (reply for request backup from layer l-1)
         then    join the designated backup mesh;

else if  (reply for request backup from layer l-2)
         then    join the designated backup mesh;

else if  (reply for request backup from layer l)

         then    join the designated backup mesh;
else     abort streaming application;
```

Figure 2. Backup request heuristics

We note that the backup request heuristics solicits backup from backup meshes of three layers in the vicinity of the disconnected node. The following algorithm runs in backup mesh leaders of all of the tree layers when a backup request message is received.

```
if       (current backup capacity > b + e )
         then    grant b+e to the requesting node;

else if  (current backup capacity > b )
         then    grant b to the requesting node;

else if  (e being served to any node)
         then    run "victim selection for e" and grant b
                          to requesting node;

else     run "victim selection for b"
         if      victim selected
                 then    grant b to requesting node;
         else    reject request;
```

Figure 3. Backup allocation heuristics

We observe that when there is sufficient bandwidth, the backup allocation algorithm of a particular tree layer attempts to allocate best video quality. As bandwidth shrinks, the allocation algorithm attempts to allocate a base video quality with best effort. The following two algorithms called "victim selection for *e*" and "victim selection for *b*" are given in figure 4 and figure 5 respectively. The algorithms determine the criteria on dropping sending enhancement and base quality videos respectively.

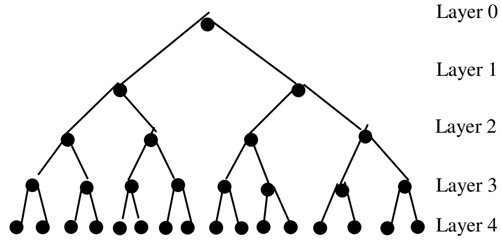
```
given    l, the layer of mesh on the broadcast tree
if       (e being served to any orphan at l-2)
         then    choose an orphan as victim from layer l-2;

else if  (e being served to any orphan at l-1)
         then    choose an orphan as victim from layer l-1;

else if  (e being served to any orphan at l)
         then    choose an orphan as victim from layer l;

else     choose the lowest bandwidth-highest delay child as
                                a victim from layer l-1;
```

Figure 4. Enhancement victim selection (victim selection for *e*) algorithm

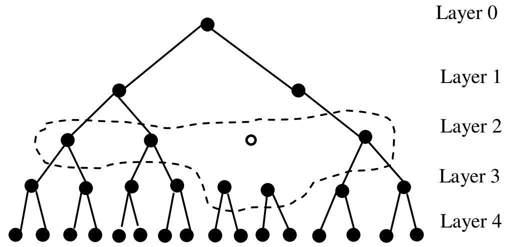
We note that enhancement victim selection algorithm attempts to drop enhancement from the lowest layer first. The layers are then scanned to find an enhancement being served. Since the algorithm is called when *e* is being served to some node, there will always be a victim selected. Victim selection for *b* algorithm also scans the layers starting from the lowest layer served. If the request node is in a higher layer of tree as compared to a node being served, then request node is given higher priority. Ultimately, the request node will always find a backup from its own layer due to its priority.

```
given    l=the layer of mesh on the broadcast tree,
         rl=layer of the requesting node

if       (rl=l+2)
         then    reject request;

else if (rl = l+1)
     then
         if (there exists an orphan in l+2 with b served)
             then select orphan as victim;
         else  reject request;

 else if (there exists an orphan in l+2 with b served)
     then select orphan as victim;

else if (there exists an orphan in l+1 with b served)
     then select orphan as victim;

else if (there exists a child in l+1 with b served)
     then choose the lowest bandwidth-highest delay child as
                                              a victim;
else reject request;
```

Figure 5. Base victim selection (victim selection for b) algorithm




(a) Original multicast tree




(b) One of the tree layer 2 nodes fail and its children join the upper layer backup mesh

Figure 6. Outlook of overlay topology

When capacity is allocated in order to supply backup to the requesting node, the node sends the buffer maps to the set of selected backup parents. Since selected backup parents are always at the same tree layer according to given heuristics, the playback lag between them is expected to be small. Thus, for backup requesting node, it is not difficult to schedule chunk requests. Figure 6 demonstrates the outlook of the overlay topology. In figure 6.a, original multicast tree is given when no failure has occurred yet. We note that every non-leaf node in a particular tree layer acts as the root of a subtree in regular broadcast operation and acts as a member of the respective backup mesh simultaneously. Figure 6.b demonstrates the situation when a layer 2 node fails and two of its children take part in the upper layer mesh and receive packets in a mesh-pull fashion.

## 4. Simulation Results

### 4.1. Simulation Testbed

In order to measure performance of the proposed algorithm, a network of 500 nodes is constructed by using GT-ITM module [35]. After creating the network topology, distances, i.e. round trip time values between nodes are calculated in NS2 environment [36] by using the shortest physical path between nodes. The tree construction algorithm is then implemented by considering two different strategies: In the first strategy, tree-push mechanism that involves repairing tree by re-allocating a full bandwidth channel to each disconnected node is utilized. In the second strategy, the hybrid backup policy of the proposed method is considered. Both of the strategies are tested in local simulator which enables running simulations with large amount of nodes. Comparisons are carried out by examining received bit-rate, network utilization and total delay from source to destination.

Video file used in the experiments are encoded using layered codec and has base layer with 500 Kbps and base plus enhancement layer with 1000 Kbps. The range of the upload bandwidth of the nodes is between 0 and 2200 Kbps with average being 1200 kbps. The single source node that is in the root of the tree is assumed to be in the system throughout the streaming period. In tree-push strategy, each node having surplus upload bandwidth reserves backup capacity that is adequate for at least on child.

## 4.2. Comparisons

Figure 7 examines the distribution of total delay from the source to individual nodes. We observe that delay has considerably reduced in the proposed method due to fact that tree-push reserves a hefty backup capacity that reduces the number of children connected to a high upload bandwidth node thereby increasing the average depth of the nodes. The delay from source to destination in the hybrid system is limited to 3 seconds while this value reaches up to 30 seconds in the tree-push system.

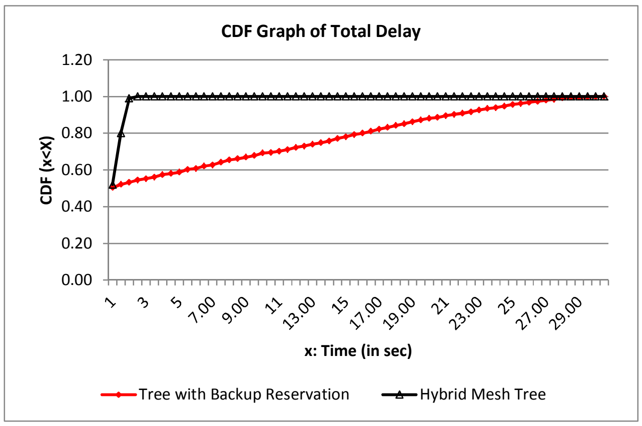



Figure 7. CDF Graph of Total Delay: From Source to Destination

Table 1. Comparative hop count values.

| | ***Tree with Backup Reservation*** | ***Hybrid Mesh Tree*** |
|---|---|---|
| *Average number of hop counts* | 124.75 | 6.99 |
| *Maximum hop count* | 249 | 8 |

In Table 1, average and maximum hop count values measured in both systems are given. Hop count of a node equals to number of nodes on the path between the source and that node. For the same number of nodes and same distribution of upload capacities of the nodes, tree based system suffers from the reservation slots and maximum hop count value reaches to 249 while that value equals to 8 for hybrid mesh tree based system. Hop count values directly affect the delay values from source to destination. These values clarify the reason of obtained delay values given in figure 7.

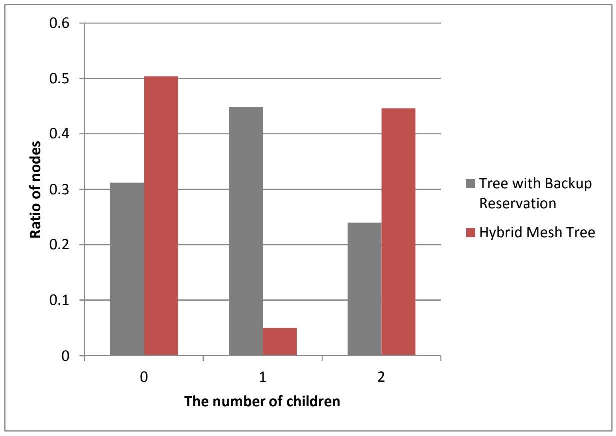



Figure 8. Classification of the nodes according to the number of children.

We also classify the nodes in the system according to their number of children. The ratio of the number of nodes having no child, one child and two children to the total number of nodes in the system is given in figure 8. Nearly half of the nodes have one child in tree based system while less than 5% of the nodes have one child in hybrid mesh tree based system. The main reason is that the nodes reserve only small part of their upload capacity in hybrid tree based system; hence most of the nodes have two children if upload capacity is enough to send video at the highest bitrate. The graph also shows that ratio of the nodes having no child is 30 % in tree based and 50 % in hybrid mesh based systems. Note that considerable number of nodes having no child is free-riders, i.e. having zero upload capacity for both systems.

To evaluate the performance of the system under different node failure (churn) scenarios, the nodes are forced to fail with a probability, $f$, that ranges between 0.1 and 0.6. The failure rates are classified as low, medium and high according to respective slots of failure probabilities. Figure 9 examines bandwidth utilization at low failure rate. We observe that the proposed method yields utilization above 70% while tree-push utilization is below 50%.

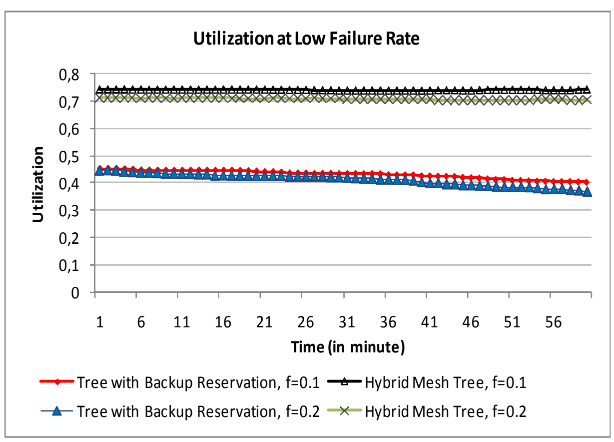



Figure 9. CDF Graph of bandwidth utilization for low failure

Utilization at medium and high failure rates are given in figure 10 and figure 11, respectively. We observe that the proposed method is still superior to tree-push strategy. Utilization decreases while failure rate increases since the number of nodes having no child also increases. In Table 2, the received bit-rate for nodes in the proposed system is given according to node failure rate.

There is no disconnected node even if the node failure rate is high. The received bit-rate for tree-push system is 1000 Kbps for almost all nodes since nodes experiencing parent failure can receive video at the highest bit-rate when it is connected to a backup parent.

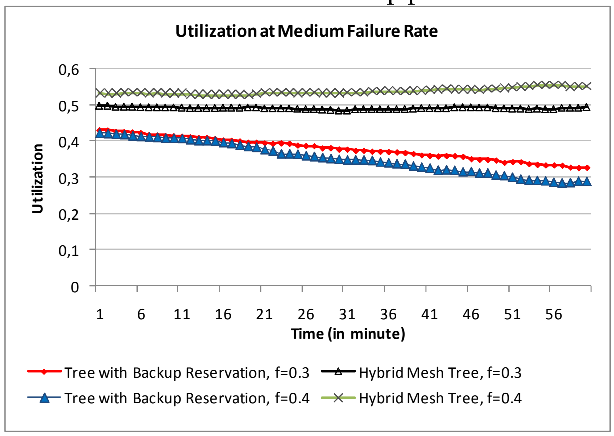



Figure 10. CDF Graph of bandwidth utilization for medium failure

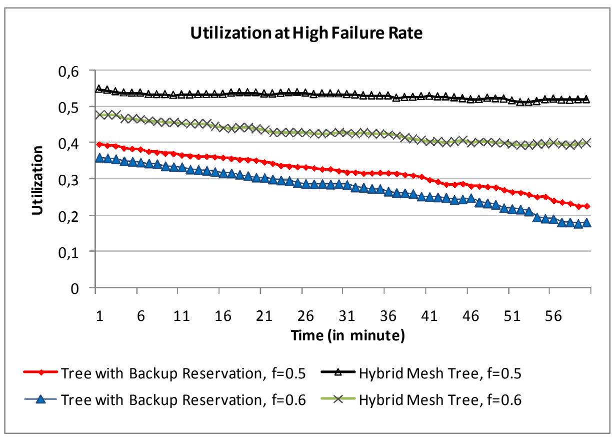



Figure 11. CDF Graph of bandwidth utilization for high failure

Table 2. Percentage of nodes according to the received bitrate.

| **Failure Probability** | **Received Bitrate** | | |
|---|---|---|---|
| | ***<500 Kbps*** | ***500 Kbps*** | ***1000 Kbps*** |
| f=0.1 | - | 28% | 72% |
| f=0.2 | - | 30% | 70% |
| f=0.3 | - | 68% | 32% |
| f=0.4 | - | 57% | 43% |
| f=0.5 | - | 39% | 61% |
| f=0.6 | - | 25% | 75% |

## 4. CONCLUSIONS

In this work, we proposed a hybrid mechanism to exploit the efficiency of both tree-push and mesh-pull methods. If churn is low, then tree-push dominates the multicast and obtains high utilization of bandwidth due to a fraction of a channel bandwidth allocated for backup. If churn is high, then mesh-pull dominates the overall mechanism. Simulations are carried out to compare bandwidth utilization in tree-push only and proposed hybrid method and it has been observed that the proposed method yields much better utilization of bandwidth.

This system can be improved and re-designed in order to stream MDC or Multiview coded video. In the future, we plan to stream Multiview coded video over our hybrid network. Multiview coded video may also have scalable layers; hence by implementing similar backup pool strategy, this system can be used to stream 3D data.

## REFERENCES

[1] Birrer, S. & Bustamante, F.E., (2007) “A Comparison of Resilient Overlay Multicast Approaches”, IEEE J. on Sel. Areas in Communications, Vol. 25, pp1695-1705.

[2] Zhang, X., Liu, J., Li, B. and Yum, Y., (2005) “CoolStreaming/DONet: A Data-driven Overlay Network for Peer-to-peer Live Media Streaming, IEEE INFOCOM.

[3] PPLive. http://www.pplive.com

[4] PPstream http://www.pps.tv/en/

[5] Banerjee, S., Bhattacharjee B. and Kommareddy, C., (2002) “Scalable Application Layer Multicast”, ACM SIGCOMM, USA.[6] Liao, X., Jin, H., Liu, Y., Ni, L.M. and Deng, D., (2006) “AnySee: Peer-to- Peer Live Streaming”, IEEE INFOCOM.

[7] Magharei N. & Rejaie, R. (2007) “PRIME: Peer-to-Peer Receiver-Driven Mesh-Based Streaming,” IEEE INFOCOM.

[8] Kumar, R., Liu, Y. and Ross, K., (2007) “Stochastic Fluid Theory for P2P Streaming Systems,” IEEE INFOCOM.

[9] Sianati, A., Abbaspour, M. and Norouzi, M. (2011) “Head-tail Video Streaming Over Peer to Peer Systems”, International Journal of Computer Networks & Communications (IJCNC), Vol. 3, No. 2, pp.68-81.

[10] Jeon, H., Son, S.C. and Nam, J.S., (2008) “Overlay Multicast Tree Recovery Scheme Using a Proactive Approach”, Elsevier Computer Commun., Vol. 31, pp3163-3168.

[11] Fei, Z. & Yang, M., (2007) “A Proactive Tree Recovery Mechanism for Resilient Overlay Multicast”, IEEE/ACM Trans Networking, Vol. 15, 2007.

[12] Sayıt, M.F., Tunalı, E.T. and Tekalp, A.M., (2012) “Resilient Peer-to-peer Streaming of Scalable Video Over Hierarchical Multicast Trees with Backup Parent Pools”, Elsevier Signal Processing: Image Communication, Vol. 27, pp113-125.

[13] Hei, X. Liang, J. Liang, J. Liu, Y., Ross, K.W. (2007) “A Measurement Study of a Large-Scale P2P IPTV System”, IEEE Trans. On Multimedia, Vol. 9, No. 8, pp1672-1687.

[14] Jin, X., Yiu, W.P.K., Chan, S.H.G. and Wang, Y., (2007) “On Maximizing Tree Bandwidth For Topology-Aware Peer-To-Peer Streaming”, IEEE Trans. on Multimedia, Vol. 9, pp1580-1592.

[15] Kim, E., Jang, J., Park, S., Sussman, A. and Yoo, J.S., (2006) “Improving Resiliency Using Capacity-Aware Multicast Tree In P2P Streaming Environments”, High Perf. Computing and Communications, pp. 925-934.

[16] Birrer S. & Bustamante, F.E., (2005) "Resilient Peer-To-Peer Multicast Without The Cost", Multimeda Computing and Netw. Conference.

[17] Kusumoto, T., Kunichika, Y., Katto, J. and Okubo, S., (2005) "Tree-based Application Layer Multicast Using Proactive Route Maintenance And Its Implementation", ACM workshop on Adv in Peer-To-Peer Multimed Str., pp49-58.

[18] Tran, D.A., Hua, K.A., and Do, T.T., (2004) "A Peer-to-Peer Architecture for Media Streaming", IEEE Journal on Selected Areas in Communications, Vol. 22, No. 1, pp121-133.

[19] Banerjee, S., Lee, S., Bhattacharjee, B. and Srinivasan, A., (2006) "Resilient Multicast Using Overlay", IEEE/ACM Trans. Networking, Vol. 14, pp237-248.

[20] Okada, Y., Oguro, M., Katto, J. and Okubo, S., (2005) "A New Approach For The Construction Of ALM Trees Using Layered Video Coding", ACM workshop on Adv. in Peer-To-Peer Multimed Str., pp59 - 68.

[21] Sayit, M. F.. Tunali, E. T and Tekalp, A. M. (2009) "Bandwidth-aware Multiple Multicast Tree Formation for P2P Scalable Video Streaming Using Hierarchical Clusters", Proc. of IEEE ICIP, Egypt, pp945-948

[22] Li, B., Xie, S., Qu, Y., Keung, G.Y., Lin, C., Liu, J., Zhang, X., (2008) "Inside the New Coolstreaming: principles, measurements and performance ımplications", Proceedings of 27th IEEE Conference on Computer Communications, pp1031 - 1039.

[23] Wang, F., Xiong, Y. and Liu, J., (2010) "mTreebone: A Collaborative Tree-Mesh Overlay Network for Multicast Video Streaming," IEEE Trans. Parallel and Distributed Systems, Vol. 21, No. 3, pp379-392.

[24] Castro, M., Druschel, P., Kermarrec, A., Nandi, A., Rowstron, A. and Singh, A., (2003) "SplitStream: High-Bandwidth Multicast in Cooperative Environments," Proc. 19th ACM Symp. Operating Systems Principles (SOSP), pp298-313.

[25] Noh J. & Girod, B., (2012) "Time-Shifted Streaming in a Tree-Based Peer-to-Peer System," Journal of Communications, Vol. 7, No. 3, pp202-212.

[26] Goyal, V.K., (2001) "Multiple Description Coding: Compression Meets the Network", IEEE Signal Processing Magazine, Vol. 18, No. 5, pp74-94.

[27] Liu, Z. Shen, Y. Ross, K., Panwar, S., Wang, Y., (2007) "P2P Video Live Streaming with MDC: Providing Incentives for Redistribution", IEEE ICME.

[28] Magharei, N. & Rejaie, R., (2006) "Understanding Mesh-Based Peer-to-Peer Streaming", International Workshop on Network and Operating System Support for Digital Audio and Video.

[29] Padmanabhan, V.N., Wang, H.J. and Chou, P.A., (2003) "Resilient Peer to Peer Streaming", Proc. IEEE International Conference on Network Protocols, pp16-27.

[30] Favalli, L., Folli, M., Lombardo, A., Reforgiato, D., Schembra, G., (2011) "A P2P Platform for Real-Time Multicast Video Streaming Leveraging on Scalable Multiple Descriptions to Cope with Bandwidth Fluctuations", International Journal of Computer Networks & Communications (IJCNC) Vol.3, No.6, pp71-94.

[31] Yin, H., Liu, X., Zhan, T., Sekar, V., Qiu, F., Lin, C., Zhang, H., Li, B., (2010) "LiveSky: Enhancing CDN with P2P", ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP), Vol. 6, No. 3.

[32] Payberah, A. H., Kavalionak, H., Kumaresan, V., Montresor, A., Haridi, S., (2012) "CLive: cloud-assisted P2P live streaming", Proceedings of IEEE 12th International Conference Peer-to-Peer Computing (P2P), pp79 – 90.

[33] Lu, Z., Wang, Y. and Yang, Y.R., (2012) "An Analysis and Comparison of CDN-P2P hybrid Content Delivery System and Model, Journal Of Communications, Vol. 7, No. 3.

[34] Rocha, M.V.M. & Rodrigueson, C.K.S., (2013) "Client's Interactive Behaviour to Design Peer Selection Policies for Bittorrent-Like Protocols", International Journal of Computer Networks & Communications Vol.5, No.5, pp141-159.

[35] “GT-ITM”, http://www.cc.gatech.edu/projects/gtitm/

[36] “Network Simulator (NS2)”, http://www.isi.edu/nsnam/ns/

**Authors**

**Muge Sayit**

Muge Sayit is an Assistant Professor at International Computer Institute in Ege University in Turkey. She received a Master of Science degree in 2005 and PhD degree in 2011 in Information Technologies from the same institute and a degree in Mathematics from Ege University in 1999. Her research interests include P2P networks, live streaming, video-on-demand and distributed systems.

**Turhan Tunalı**

Turhan Tunalı earned a BSc degree in Electrical Engineering from Middle East Technical University and MSc Degree in Applied Statistics from Ege University, both in Turkey. He then earned a DSc Degree in Systems Science and Mathematics from Washington University in St. Louis, USA in 1985. After his doctorate study, he joined Computer Engineering Department of Ege University as an assistant professor where he became an associate professor in 1988. During the period of 1992–1994, he worked in Department of Computer Technology of Nanyang Technological University of Singapore as a Visiting Senior Fellow. He then joined International Computer Institute of Ege University as a Professor where he was the director. In the period of 2000–2001 he worked in Department of Computer Science of Loyola University of Chicago as a Visiting Professor.He is currently working as a Professor at Department of Computer Engineering of Izmir University of Economics. His current research interests include adaptive video streaming and Internet performance measurements. Dr. Tunali is a member of the IEEE.