Multiple Tree Multicast in a Dynamic Environment

David A. Johnston, David R. McIntyre[1], Francis G. Wolff, Christos A. Papachristou

EECS Department, Case Western Reserve University
Cleveland, Ohio, USA 44106

[1] CS Department, Cleveland State University
Cleveland, Ohio, USA 44115

Abstract

Multiple multicast trees have been shown to increase the performance of data distribution when compared with single tree multicast. Node loss and congestion changes the performance characteristics of the multicast trees. Multicast tree performance feedback can be used to determine the optimal tree to use based on the feedback. We further examine an optimizing methodology, Probabilistic Multicast Trees (PMT), for multiple multicast trees which makes use of the performance feedback, generates a probability of usage for each multicast tree based on that feedback and then makes intelligent choices about which multicast tree to use for a given packet in the presence of node loss and congestion.

Keywords: *Dynamic Multicast, Application-Level Multicast, ALM, Probabilistic Multicast Trees, PMT, Adaptive tree selection, Content distribution.*

1. Introduction

Smart phones, movies on demand, regulated industrial process information; the thirst for data access has never been greater and will only continue to grow. Network infrastructure must continue to evolve to meet the ever increasing demand for data. This is especially true when many devices demand the same data at the same time. Since improvement in pure network bandwidth capabilities is only part of the solution; many researchers have investigated efficient transfer of information and a variety of solutions have been proposed. One popular solution is hardware-based methods to distribute this data which resulted in IP multicast. Unfortunately, IP multicast has several limitations that prevent it from being globally used across multiple service provider domains on the Internet. Application level multicast (ALM) overcomes is weakness of being tied down to particular hardware solution [8]. ALM is a multicast overlay network which can be described as a tree. Saltzer [16] argued that the network should be kept as simple as possible and for any multicasting that the intelligence resides at the application layer. ALM is the fundamental principle of the “end to end” argument that Saltzer proposed.

The dynamic behavior of multicast networks presents unique challenges for data distribution in a network environment. Some multicast methodologies repair the multicast trees as needed in the presence of failures. Other multicast systems improve the performance of the multicast tree through probing methods. Both methods aim to address the issue of node loss and congestion. Other multicast research has attempted to address the long delays and performance issues of node loss and congestion by using redundant paths, additional replication of data or using wholly redundant trees. However, in all cases multicast node failures and network congestion still cause long delays, performance issues or missing data as the data is delivered.

In general, *multiple* multicast trees have been shown to benefit multicasting applications in that they increase throughput and reliability [3][10][17].

Many single multicast tree solutions and multiple multicast tree solutions have been developed; however, we still need to make these solutions more efficient. Several approaches to multiple tree multicasting have been implemented [3][4][7][8][15]. These approaches were designed with two goals. The first is to improve performance over the single multicast tree approach and the second is to manage node loss which is a fundamental problem of single multicast trees. Other techniques to manage node loss that were built upon multiple multicasting methods include replication of packets besides just the expected distribution through the tree, forward error correction [14] and multiple description coding (MDC) [11]. All of these schemes, which use additional network bandwidth, address the inherent lossy nature of wireless networks.

We explore Probabilistic Multicast Trees (PMT) [12][13] as applied in a dynamic network environment. PMT is an optimizing mechanism that is intended to improve the capabilities of any multiple multicast tree methodology with respect to the management of node loss and network congestion. PMT is designed to provide two main advantages over other multiple multicast tree schemes. It improves both data delivery latency, and data delivery efficiency.

Data delivery latency, ML_t, is an important performance measure for multimedia streaming. It is the total summation of all the source-to-destination packet delivery times for multicast tree t. The time difference can be calculated from a timestamp, T_s, that the source puts into each packet and the receive time, T_r, of the same packet by the receiving destination client. The goal of PMT is to reduce this latency on average over all the receiving clients. Data delivery latency can be expressed by the following equation where the summation is taken over all packets received.

$$ML_t = \sum_r T_r - T_s \qquad (1)$$

Data delivery efficiency (ME_t) refers to the percentage of the total number of multicast packets delivered (P_d) to all client destinations compared to the total number of packets sent (P_s) by the source as expressed by the following equation.

$$ME_t = P_d / P_s \qquad (2)$$

In this paper, we extend our previous work on PMT by taking into account node loss. PMT increases data delivery efficiency by delivering a higher percentage of the packets based on improved multicast tree selection. PMT achieves this by more severely punishing trees with drop out nodes by adding increased feedback delays to these trees resulting in the overall latency feedback for any tree containing such nodes being significantly increased. This results in PMT tending to chose alternate lower latency trees with fewer lost nodes for future transmissions which ultimately is reflected in reduced data delivery latency.

The remainder of this paper is laid out as follows: Section 2 discusses node failure and congestion, Section 3 describes the design of PMT, Section 4 describes data metrics, Section 5 shows the results, and Section 6 discusses conclusions.

2. Node Failure and Congestion Simulation

Past multicasting research has focused on three main areas: building trees efficiently, reducing maintenance overhead, and using other forms besides trees to deliver the data. Multicast overlay network failures causing long delays and performance issues as the data is delivered have been only moderately addressed. Most approaches have either supported repairing the tree as failures occurred or improving the performance of the tree through probing methods. The performance improvement methods, by design, also repaired the tree. Other research focused on addressing the long delays and performance issues by using redundant paths, replicating data or using wholly redundant trees. These methodologies repair the multicast trees as needed in the presence of failures. One form of multicasting uses several multicast trees where data is sent equally on each tree. This methodology is called multiple tree multicasting. Multiple multicast trees have been shown to benefit multicasting applications in that they increase throughput and reliability and several approaches to multiple tree multicasting have been implemented. Multiple multicast trees are built at the application layer to support the data distribution. In a given multicast tree, a subset of the client nodes assists with the data delivery. With multiple multicast trees, more client nodes assist with data delivery. Whether streaming video or sharing files such as with Napster or BitTorrent, using multiple multicast trees is more efficient than using a single multicast tree. These approaches were designed to manage node loss which is a fundamental problem of single multicast trees specifically targeting wireless networks. These mechanisms to manage node loss include additional replication of packets besides just the expected distribution through the tree, forward error correction [14] and multiple descriptions coding [11]. No matter which methodology is examined, repairs of the multicast trees take a long time with respect to the time frame for data delivery where data delivery is on the order of tens of milliseconds and tree repair is on the order of tens of seconds.

Unfortunately, no one multicast solution has addressed all of the problems of multicast data delivery. The ideal solution would provide the following properties so that the application layer would be only minimally affected due to changes to the structure of the multicast tree [1][2][5][7][9][13][18]:

1. *Minimize time to deliver the data.*
2. *Maximize the number of packets delivered.*
3. *Minimize bandwidth utilization.*
4. *Minimize network maintenance overhead.*
5. *Quick detection of failures.*
6. *Quick response to those detected failures.*
7. *Seamless repair mechanisms.*

When there are no disruptions to the multicast overlay network, the data is transmitted effectively and received appropriately by all client nodes via the multicast trees. However, in the presence of network congestion and node turnover, problems arise. Early multicast overlay network research investigated many of the properties of an ideal solution but fundamentally failures still cause too much delay in data delivery. Since people are the ultimate client of the data any perceived quality of experience degradation causes frustration. Although an ideal solution is not possible, any new solution should have most of the following attributes:

1. *It must be efficient in that it performs comparably to other solutions in a static environment and does not cause undue stress on the multicast overlay network by reducing bandwidth utilization and multicast tree maintenance overhead.*
2. *The solution must be resilient in that failures of the multicast overlay network are transparent to the application or minimally disruptive.*
3. *It must be quick to detect network failures and just as quickly respond to the failures to maintain the transparency.*
4. *The solution must provide excellent performance under a variety of conditions that will rival or surpass other similar solutions.*
5. *The solution must provide for quick recovery in the face of network dynamic behavior seamlessly repairing the multicast tree network.*

The main contribution of this work is the development of Probabilistic Multicast Trees as an optimizing mechanism to improve the data delivery latency and data delivery efficiency of *any* multiple multicast tree methodology. PMT was designed to be inserted into any multiple tree multicasting model. The advantage gained by using PMT is that it improves upon the management of the dynamic behavior of the clients where the target connectivity is constantly changing because of its feedback mechanisms and probabilistic tree selection. PMT will be compared against SplitStream, a multiple multicast tree model, under a variety of conditions to show the advantages that PMT provides. SplitStream [6] is a multicasting model that relies on a structured peer-to-peer overlay network called Pastry [10][21], and on Scribe [5], an application-level multicast system built upon this overlay to construct and maintain multicast trees.

3. Probabilistic Multicast Trees

PMT is based on latency feedback. In order to provide latency feedback a separate periodic thread was created that executes at a fixed time period of one second. This thread sends feedback data to its parent for each multicast tree. The feedback packet consists of the averaged feedback from all the parent's children and the parent's average latency delay value. Of course, missing feedback from children causes the averaged delay value to be larger thereby penalizing the multicast tree. New feedback values overwrite older feedback values. It is these feedback values that are used to generate the probability of usage table that the source will use to make a decision about which multicast tree to use for each packet. The Scribe [5] “anycast” functionality was added to enable this feedback from child to parent. The latency feedback mechanism is the key to PMT.

PMT is built upon the following premise: since each multicast tree does not have the same performance characteristics, PMT relies on the latency feedback mechanism from each multicast tree to generate a probability percentage of usage for each multicast tree. The probability percentage of usage for a given multicast tree is a value indicating how frequently a particular multicast tree may be chosen. For each packet sent, one multicast tree is chosen randomly based on its probability percentage of usage. The higher a value for a particular multicast tree, the higher its probability is for being chosen for the next packet to be sent. As a result, the tree with the best performance will be used most often and poorer performance trees will be used less frequently. However, less frequently poorer performance trees will nonetheless occasionally be used possibly yielding improvements in latency feedback possibly due to decreased network congestion for these trees.

There are two reasons for using multiple trees. The first is to maintain the benefits of multiple multicast in that more nodes are actively multicasting the data. The second is to account for changing bandwidth patterns as the underlying networks exhibit their dynamic behavior. The decision to select a multicast tree for a packet about to be sent is based on the generation of a random number and this number is applied against the trees' probability percentage of usage to make the selection. As the performance of the multicast trees change due to node loss, network congestion, tree performance improvement or other changes due to mobile nodes, the latency feedback mechanism continually provides updated latency values to the source so that as the multicast trees' probability percentage of usage is recalculated tree selection chooses the best tree most often at any given time. Recalculation is performed at regular intervals once per second.

PMT improves upon the management of the dynamic behavior of the clients when the target connectivity is constantly changing because of its feedback mechanisms and probabilistic tree selection. This improvement manifests itself in data delivery latency, a metric measured as an output of the process. An improvement in the metric is an indication that using PMT is advantageous.

Figure 1 illustrates three multicast spanning trees. To the source node each tree is a wholly separate multicast tree. In SplitStream each tree is used in a round robin fashion to send each individual packet. For example, the first packet is sent on the blue tree, second packet is sent on the red tree, the third packet is sent on the black tree. The fourth packet will be sent on the blue tree as the process repeats until all the data is transmitted. Figure 2 shows the three non-overlapping trees.

PMT does not follow this round robin process for tree selection. For this example, Tree *2* has been determined to be a more efficient tree for transmission than Tree *1*. Tree *1* has been determined to be a more efficient tree for transmission than Tree *3*. Tree *2* is assigned a probability of usage of 0.67 based on its relative efficiency as compared to the other two trees. Tree *1* is assigned a probability of usage of 0.31 based on the same criteria. Tree *3* is assigned a probability of usage of 0.02. The efficiency of each tree was measured via feedback over a period of time with the network in a steady state mode which resulted in the assigned probabilities.

The calculation of the probabilities will be described below. To choose a tree for transmission a random number is generated. If the random number is less than 0.67 then Tree *2* is chosen. If the random number is between 0.67 and 0.98 then Tree *1* is chosen. If the random number is greater than 0.98 then Tree *3* is chosen. This process is repeated for each packet transmitted. As long as no significant changes occur in the performance of the trees, then the probability of usage for each tree will remain the same. When the efficiency of the trees changes then the probability of usage will change based on the relative performance of each tree.

PMT data delivery latency includes the estimated missing packet latency delay times. Also, congestion has much less impact on the latency feedback when compared to lost node impact. This is to be expected since the missing node penalty value is so large compared to congestion.

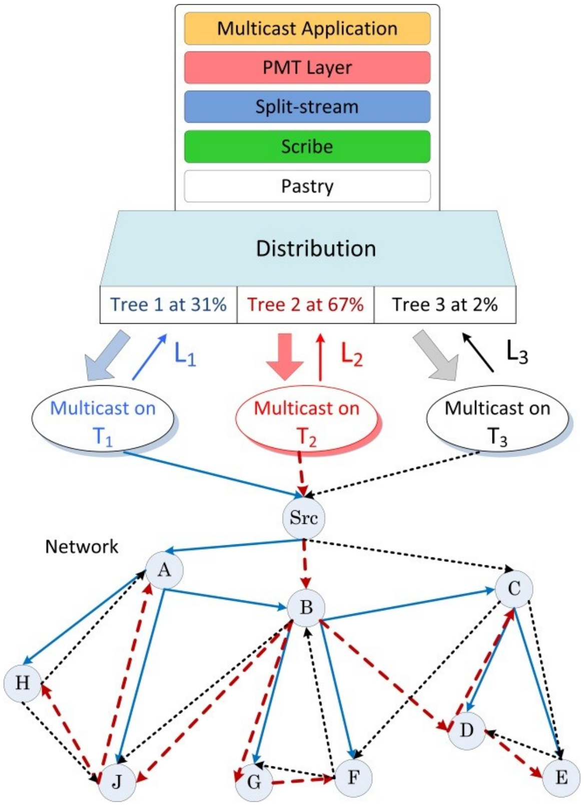

Fig. 1 PMT Multicast Tree Selection

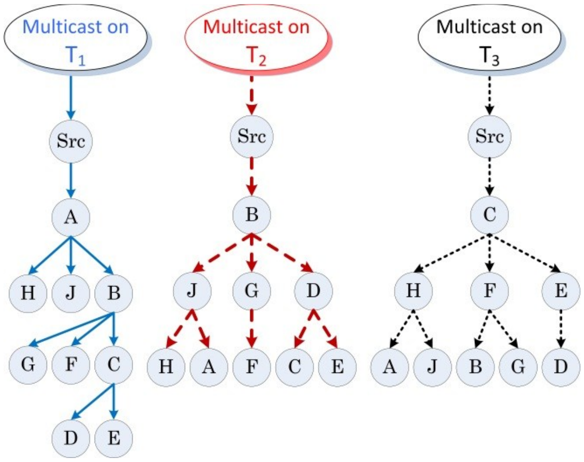

Fig. 2 Three Multicast Trees

4. Data Metrics

Simulation data is collected and passed through a series of calculations that will be used for analysis and comparison.

One metric is *total delay latency time*. This is the summation of all the time differences from all the packets received. During dynamic testing some nodes are lost which means that the remaining nodes will not receive all of the packets. The *non-normalized data delivery latency* is the total delay latency time for the actual number of packets received and does not include the missing packet latency delays.

The *normalized data delivery latency* metric, NL, is based on the non-normalized data delivery latency metric. NL is calculated as if the receiving node had received every packet. For each lost node a sufficiently large penalty value is substituted for the feedback value from the node.

A *Better/Worse percentage* metric, BW, is calculated by dividing the PMT NL value by the SplitStream NL value.

$$BW = \frac{NL_{PMT}}{NL_{SplitStream}} \times 100\% \qquad (3)$$

A value less than 100% means that the PMT method performed better than SplitStream. The BW percentage is an indication of how well the PMT method performed in a simulation test. The results of these calculations are analyzed statistically and presented in the next section.

5. Results

Simulations were run as follows. The source puts the tree number into the packet and a time stamp into each packet. The receiver uses the tree number to drive metric collections for each tree and it performs a difference calculation to generate the delay time from the source. This delay time is added to the data delivery latency total and is used for worst case delay comparison. The following enumeration describes the raw data collected by the nodes of the multiple multicast networks.

1. *The source node tracks the number of packets sent on each tree. For SplitStream this will always be the same number; however, for PMT, the number will vary for each tree.*
2. *The client node tracks the total number of packets received on each tree.*
3. *The client node tracks the worst case data delivery latency per tree.*
4. *The client node tracks the total data delivery latency for all packets received.*

PMT is compared against SplitStream using the total delay latency metric. Each set of tests is averaged and the mean of the total delay latency is compared directly.

Each dynamic test simulation run had an initial node count and a specified number of nodes to be removed - approximately 10% of the total. After all the nodes were created a subset of nodes were randomly chosen for removal beginning 8 seconds after data transmission started. The chosen nodes were removed from the "active" list and placed on the "to-be- removed" list. The "to-be-removed" node list is iterated to remove the nodes at the appropriate point in the simulation so that the nodes can be removed from the trees. This process provided sufficient dynamic behavior for comparison.

GT-ITM (Georgia Tech Internet Topology Models) is an internet topology generator [20]. Since its release GT-ITM has been widely used in the scientific community for network simulations. We used the GT-ITM model with 8 trees and node counts of 550, 1100, and 2200. Figure 3 and Figure 4 show the average total data delivery latency for the two methods, PMT and SplitStream. As indicated from the means of the two charts, PMT shows a 16% improvement in average total data delivery latency. This improvement percentage is actually more impressive because the SplitStream data delivery latency was calculated using the non-normalized formula which does not include the missing packet latency delays whereas the PMT means included the estimated missing packet latency delay times. The forth Figure 5, shows the calculated comparisons between the actual data delivery latency for PMT and a calculated normalized data delivery latency for SplitStream within the bounds of the same network configuration. The comparison shows a 14% improvement by the PMT over the SplitStream code.

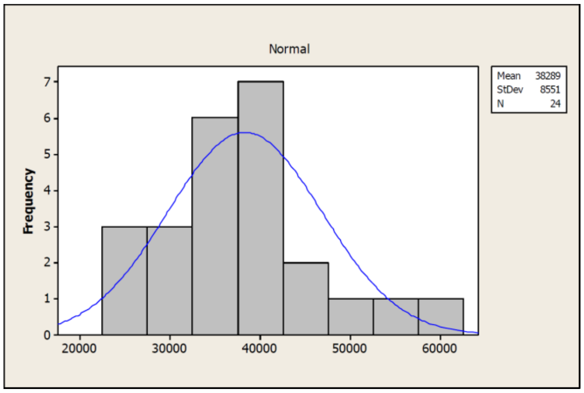

Fig. 3 SplitStream Data Delivery Latency

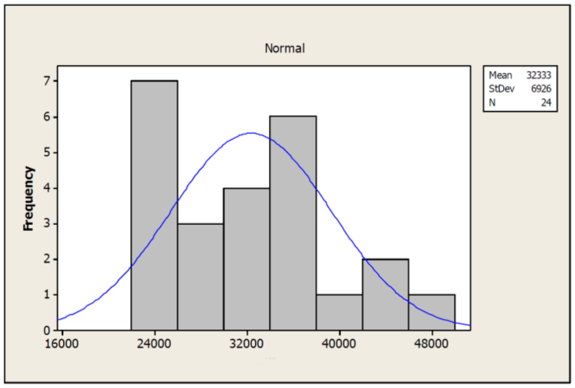

Fig. 4 PMT Data Delivery Latency

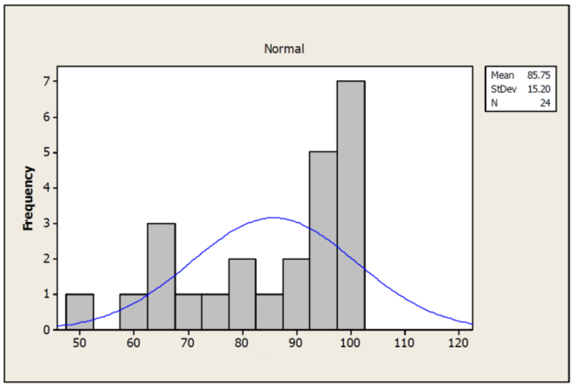

Fig. 5 PMT versus SplitStream Better/Worse

6. Conclusions

This paper has presented PMT, an optimizing mechanism that is intended to improve the capabilities of *any* multiple multicast tree methodology with respect management of node loss and network congestion. Simulations with PMT have shown to improve data delivery latency over the multiple multicast tree scheme SplitStream. As a byproduct data delivery efficiencies are improved by PMTs avoidance of trees with high node loss.

References

[1] D. Andersen, H. Balakrishnan, F. Kaashoek and R. Morris, “Resilient Overlay Networks,” Proc. 18th ACM SOSP, October 2001.

[2] T. Baduge, A. Hiromori, H. Yamaguchi and T. Higashino, “A distributed algorithm for constructing minimum delay spanning trees under bandwidth constraints on overlay networks,” Systems and Computers in Japan, Vol. 37, No. 14, pp. 15 – 24, 2006.

[3] S. Banerjee, S. Lee, B. Bhattacharjee and A. Srinivasan, “Resilient multicast using overlays,” Proc. of ACM SIGMETRICS, June 2003.

[4] S. Birrer and F. E. Bustamante, “Magellan: performance-based, cooperative multicast,” 10th International Workshop on Web Content Caching and Distribution (WCW 2005), pp. 133- 143, Sept. 2005.

[5] S. Birrer and F. E. Bustamante, “Resilient peer-to-peer multicast without the cost,” Proc. of MMCN, January 2005.

[6] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, “Splitstream: High-bandwidth multicast in cooperative environments,” Proc. of the 19th ACM SOSP, October 2003.

[7] M. Castro, P. Druschel, A.-M. Kermarrec and A. Rowstron, “SCRIBE: a large-scale and decentralized application-level multicast infrastructure,” IEEE Journal on Selected Areas in Communications, vol. 20, no. 8, pp. 1489–1499, 2002**.**

[8] Y. Chu, S. G. Rao, S. Seshan and H. Zhang, “A Case for End System Multicast,” IEEE Journal on Selected Areas in Communications, vol. 20, no. 8, Oct 2002

[9] N. Feamster, D. Andersen, H. Balakrishnan, and F. Kaashoek, “Measuring the Effects of Internet Path Faults on Reactive Routing,” Proceedings of ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS), 2003.

[10] Z. Fei, and M. Yang, “A Proactive Tree Recovery Mechanism for Resilient Overlay Multicast,” ACM Transactions on Networking, vol. 15, no. 1, February 2007.

[11] V. K. Goyal, “Multiple Description Coding: Compression meets the Network,” Signal Processing Magazine, vol. 18, no. 5, pp. 74-93, 2001.

[12] D. A. Johnston, D. R. McIntyre, F. G. Wolff and C. A. Papachristou, “Optimizing Application Level Multicast Trees over Wireless Networks,” IEEE NAECON 2011.

[13] D. A. Johnston, D. R. McIntyre, F. G. Wolff and C. A. Papachristou, “Probabilistic Multicast Trees,” Journal of Computer Science & Technology (JCS&T), vol. 12, no. 1, April 2012.

[14] D. Koutsonikolas, Y. C. Hu, “The Case for FEC-based Reliable Multicast in Wireless Mesh Networks,” Proc. of 37th Annual International Conference on Dependable Systems and Networks Washington DC, pp 491 – 501, 2007

[15] A. Rowstron and P. Druschel, “Pastry: Scalable, decentralized object location and routing for large-scale peer to peer systems,” Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware’01), 2001

[16] J. H. Saltzer, D. P. Reed and D. D. Clark, “End-to-End Arguments in System Design,” ACM Transactions on Computer Systems, vol. 2, no. 4, pp 277-288, Nov 1984

[17] K.K. To and Jack Y.B. Lee, "Parallel Overlays for high data-rate multicast data transfer Computer Networks," 51, pp. 31-42, 2007

[18] D. Tran, K. A. Hua and T. Do, "ZIGZAG: An Efficient Peer to Peer Scheme for Media Streaming," Twenty-Second Annual Joint Conference of the IEEE Computer and Communications (INFOCOM 2003), Vol. 2, pages 1283- 1292, 2003

[19] V. Venkataraman, K. Yoshida and P. Francis, "Chunkyspread: Heterogeneous Unstructured End System Multicast," Proceedings of 14th IEEE International Conference on Network Protocols, 2006.

[20] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to Model an Internetwork", Proceedings of IEEE INFOCOM '96, San Franscisco, pp. 594–602, March 1996.

[21] FreePastry Simulation http://www.freepastry.org/FreePastry

David A. Johnston received a Ph.D. in Engineering and Computer Science from Case Western Reserve University.

David R. McIntyre received a Ph.D. in Computer Science from the University of Waterloo. He is Associate Professor of Computer Science at Cleveland State University.

Francis G. Wolff received a Ph.D. from Case Western Reserve University. He is a Senior member of the IEEE and the Senior member of the ACM.

Christos A. Papachristou is Professor of Electrical Engineering and Computer Science at Case Western Reserve. He received the Ph.D. degree in Electrical Engineering and Computer Science from Johns Hopkins University. He is a Fellow of the IEEE and a member of the ACM and Sigma XI, and is listed in Who's Who in America.