# Virtual Direction Multicast for Overlay Networks

Suat Mercan
University of Nevada - Reno
Computer Science and Engineering
Reno, NV 89557
smercan@cse.unr.edu

Murat Yuksel
University of Nevada - Reno
Computer Science and Engineering
Reno, NV 89557
yuksem@cse.unr.edu

***Abstract*—In this paper we propose Virtual Direction Multicast (VDM) which aims to minimize network usage and disconnection time for video multicast applications on peer-to-peer overlay networks. It locates the end hosts relative to each other based on a virtualized orientation scheme. It builds multicast tree by connecting the nodes which are estimated to be in the same virtual direction. By using the concept of directionality, we target to use minimal resources in the underlying network. We compare VDM against a Closest Node Multicast (CNM) protocol that connects nearby nodes to construct the multicast tree. Simulation results show that our proposed technique VDM consistently outperforms CNM under different churn rates.**

***Index Terms*—overlay multicast; peer-to-peer; peer-to-peer TV; path stretch**

## I. Introduction

With the improvement of bandwidth and expansion of Internet usage, content providers are aiming to deliver multimedia content over the Internet. Live video streaming applications have become highly popular both as backbone (e.g. IPTV [1]–[3]) and overlay (e.g. P2PTV [4], [5]) distribution applications. High demand for such applications which constitute a large amount of today's Internet traffic increases server load and network bandwidth consumption. This recent trend in moving video distribution on to the Internet is calling for mechanisms to efficiently and scalably transfer video content to many receivers from a single source. Such content delivery to many receivers is desired to be seamless to the multi-provider operation of the Internet and capable of handling a lot of churn from the network as well as the receiver population.

IP multicast [6] was proposed to provide efficient group communication, and can be implemented in network layer by integrating additional algorithms and multicast tables to routers. Even though IP multicast provides bandwidth efficiency, ISPs are reluctant to support it since it introduces extra workload and complicates network management. Thus, IP multicast has limited application on the Internet due to scalability, deployment and limited support for high level functionality. Since IP multicast does not get much support from network operators, application layer multicast (ALM) [7], [8], [10], [11], [13], [14], [17]–[20], has emerged as an alternative to achieve the multicast functionality. The idea is to establish a virtual network among end-hosts, each of which not only receives the stream but also forwards to other end-hosts. ALM does not require support from network layer routers. End-hosts constitute multicast group which move functionality from the network layer to the application layer. This makes ALMs easy to deploy. Thus, though backbone-level multicast streaming applications such as IPTV will serve a particular need, overlay multicast streaming solutions will likely to continue an ubiquitous solution to the multimedia delivery to the end points.

In this paper, we propose a new P2P multicast streaming technique which is called Virtual Direction Multicast (VDM). VDM focuses on tree construction method to reduce redundant data transmission and failure recovery to decrease data reception outage under churn. We aim to find the most appropriate parent for a peer so that data travels the minimum possible path. To converge on a tree with a minimal source-receiver delay, we exploit round-trip times (RTTs) to measure the "virtual distances" between peers. VDM uses an iterative approach by selecting a child which is in the same "virtual direction". The iterative process continues until the best potential parent is found. The key idea is to connect the nodes which are in the same virtual direction so that we try to minimize the source-destination path length for the overall structure. We also discuss how the concept of virtual direction can be generalized to metrics other than delay, such as loss or bandwidth.

We organize our paper as follows: We start with a comprehensive discussion of key issues in designing overlay multicast schemes in Section II. We survey related work in Section III. Section IV gives a detailed description of our VDM protocol. Simulation setup and results of a comparative performance evaluation of VDM are presented in Section V. Finally, in Section VI, we summarize our work with conclusions and possible future work.

## II. Overlay Multicast Issues

Application layer multicast is flexible and easy to deploy, but its performance heavily depends on some of its design techniques such as how the overlay multicast tree is constructed. The common goal of all ALM methods is to obtain an efficient and robust overlay multicast tree. However, the criteria for effectiveness of the overlay multicast tree can be various depending on the application goals.

Live multimedia streaming is a real-time application that requires *minimal delay*, where the delay is defined as time needed for a packet to reach its receiver(s). The data packets should ideally traverse the minimum path while being transferred from source to destination; however, the end-to-end delay might be longer due to a high number of intermediate nodes as a result of an inappropriate overlay structure. Overlay tree design should provide possible minimum delay for each multicast receiver.

Another challenge to be addressed in a P2P network is the ad-hoc behavior of the members of the overlay tree. Since most of the P2P systems do not have membership requirements, peers might join and leave at any time. This behavior, known as churn, makes tree maintenance harder. Ungraceful exit made by a peer may cause interruption of data reception at its descendants. When such ungraceful exits happen, the orphan peers need to be quickly reconnected to another parent. Long and frequent data outage is not acceptable for real-time applications, and thus *robustness against churn* is of crucial importance for overlay multicasting.

Moreover, large volume of data is transmitted in multimedia applications, which requires *avoiding redundant transmission* of the multicast traffic. The reason for client/server model not being feasible for these applications is that data has to be sent to each receiver separately which consumes bandwidth and server power. IP multicast is the best solution from this perspective if we disregard its shortcomings to be deployed on the Internet. It prevents duplicate transmissions. Thus, ALM can not solve this problem as optimum as IP multicast. Because of ALM's nature, this redundant transmission will occur in some links. But, it is crucial to minimized the amount of such redundant transmissions by efficiently constructing and maintaining the overlay multicast tree.

One of the drawbacks of ALMs is being deprived of underlying network structure knowledge. This makes it hard to construct efficient multicast data paths. This could be solved by doing some measurements. Most used technique is to measure distance between peers as round trip times (RTTs). Some geo-location techniques which estimate geographical location of an IP and topology maps also can be used to overcome this problem. In general, measuring and effectively using the information about the underlying network is a key issue in ALM design.

In an overlay multicast protocol, there needs to be some message exchanges between peers. The overlay protocol uses these messages for initialization and extension of the tree, to accommodate underlying network changes and for recovery from peer failures. *Minimizing the control messaging overhead* is crucial to the scalability of such a system. Uncontrolled increase of control messages constrains the scalability of the overlay system.

A peer in overlay serves its children while being served by its parent. A peer can serve only a limited number of children based on its outgoing bandwidth, which we call *degree limit*. Because of this limitation, the optimal overlay tree may not be built. Even there are some nodes that does not contribute to tree at all which we call free riders. Some enforcement mechanisms are needed against free riders.

In an overlay tree, it is advantageous for a node to be close to source since it experiences lower delay. Also, reliability and consistency of nodes which are in higher levels in tree are more important. An incentive mechanism can be used to move upward reliable nodes in terms of consistency and contribution to the tree. So, we get a more stable tree while rewarding trustable peers.

All matters we mentioned in this section can be used to improve performance of an ALM. But, we focus on tree construction mechanism and quick recovery method that we detail later in Section IV.

## III. Related Work

Numerous algorithms have been proposed using different techniques to achieve a successful overlay structure for live video streaming. Overlay network construction techniques can be classified into two main categories according to their structure [9]; mesh-based and tree-based.

In mesh-based approach, either nodes join to multiple disjoint trees (e.g., SplitStream [10] and ChunkySpread [11]) or choose a set of neighbors to create a mesh topology (e.g., CoolStreaming [12] and Narada [17]). This approach is known as a pull-based mechanism. Important point for the mesh-based approach is its robustness to churn; but it is more expensive to maintain due to its higher control overhead, which also limits scalability. This approach is not satisfactory in terms of network resource usage and is wasteful in leveraging the underlying network bandwidth.

In second, tree-based approach BTP [13], HMTP [14], Yoid [18], nodes are organized in a tree structure rooted at the source. Nodes have parent-child relationship. When a node receives a packet from its parent, it forwards to children; which is also named as a push-based mechanism. The tree is extended when a new node joins the group. Tree-based approach is efficient in terms of avoiding redundant data transmission; but, when a node leaves its offspring, peers suffer from data outage. The tree must be repaired quickly in this case. So, it is not robust to churn from this point. Another disadvantage is that while interior nodes are busy with forwarding data, leaf nodes stay idle; which is an unfair manner for members. Tree-based approach is scalable and can used for large groups.

We give short explanations of two ALM methods.

In Banana Tree Protocol (BTP) [13], a newcomer joins to the root first (i.e. becomes a child of the root), then switches its parent if it finds closer peer among the siblings. So, it attempts to lower tree cost. This method, even though startup time is short, causes too many oscillations for peers until they find the appropriate parent. Also, the fact that each node connects to root first put too much load for root and upper level nodes.

HMTP [14] interconnects IP-enabled islands. If IP multicast is available in any subnet, one node is selected as head to join the overlay tree and IP multicast is used in subnet. The key idea in HMTP is connecting nearby peers. When a new peer wants to join, it contacts the source, and gets list of the children. By probing each child, it finds closest child to itself in terms of delay. It repeats the same process with the closest child. This iterative process is repeated until best potential parent is found. HMTP also applies a tree refinement process. Each node randomly select a peer in its root path and look for if any closer peer than its parent connected in meantime. This refinement process is repeated periodically. HMTP aims to reduce routing inefficiency. It also proposes a foster child concept to shorten startup time. A node connects root at the beginning to start stream immediately. Then, it jumps to ideal parent when it is found.

## IV. Virtual Direction Multicast

VDM is an overlay multicast protocol that builds its tree by *establishing parent-child relationships between nodes that are determined to be on the same virtual direction* based on virtual distances between them. The virtual distances are calculated according to the performance (i.e., delay, loss, or bandwidth) of the connections between the nodes. The ultimate overlay tree can be customized by using performance metric of choice while calculating the virtual distances.

### *A. Key Design Considerations and Components*

A key design component of VDM is *directionality*. We position a newly joining peer relative to the existing peers with an iterative process using this concept of directionality. We take the peers three by three, and we estimate the location of the new peer relative to the existing peers by comparing inter-peer distances.

Another design component of VDM is the capability of virtualizing the underlying network in different ways. Though we only consider inter-peer delay (i.e. RTT) in this paper, it is possible to establish "virtual directions" based on other performance metrics such as loss or bandwidth.

In environments like the P2P networks, churn is a major issue. When peers are leaving or joining frequently, the performance of the protocol depends heavily on being able to swiftly switch to a new tree. Reevaluation of the overall multicast tree requires a centralized approach and is typically not possible within the very short period of time available for switchover. Our directionality-based procedure is completely distributed and can quickly establish a new and good performing tree with local repair.

*1) Virtual Directionality on a Line:* VDM exploits virtual directions when peers are joining or leaving the system. Suppose that that there is a source (S) and an existing node (E) which are already in overlay network. There is a new node (N) which is going to join to overlay tree. We measure the distances among these three nodes. In Figure 1, S and E which are already in overlay network keep their position while N could be in three different locations. Based on position of N, three nodes can form three combinations in linear representation, i.e., a line. It is possible to establish the virtual directions on a 2-D space, but we limit the complexity to a 1-D space in this paper.

Distances d1, d2 and d3 are round-trip times (RTTs) measured by probing. Longer distance is generally not equal to the sum of shorter distances which seems equal in linear representation. We look at the longest one to determine into what case the combination falls. There are three cases.

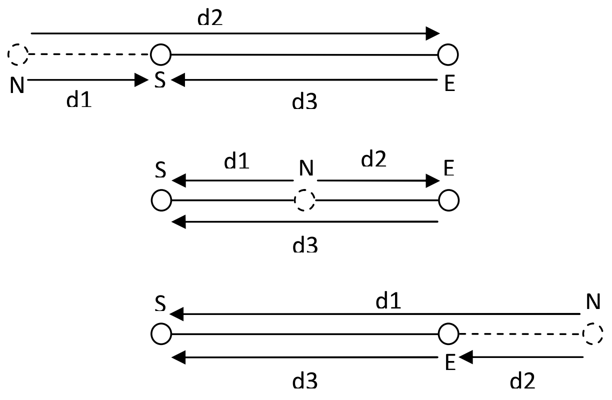

Fig. 1. Directionality Concept

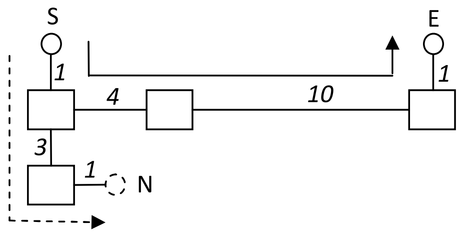

Fig. 2. Case I

Case I – *The new node is on a new and separate virtual direction than the existing node:* The source node is in the middle of the new and the existing nodes, i.e., S is in the middle. In this case, N should be connected to S. Figure 2 represents this case in router level. Numbers are representing relative link delays. Dashed arrow shows new stream.

Case II – *The new node is on the same virtual direction as the existing node, but closer to the source:* The new node is in between the source and the existing nodes, i.e., N comes in between S and E. N becomes child of S and parent of E. When we look at Figure 3, dotted line shows old stream, and it is removed when new connection, dashed line, is established.

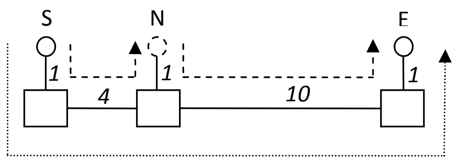

Fig. 3. Case II

Case III – *The new node is on the same virtual direction as the existing node, but farther to the source:* The existing node is in between the source and the new nodes, i.e., E is in between S and N. N is connected to E. Dashed line in Figure 4 is new connection.

With this technique, we aim to minimize multiple packets on the same link and resource usage in the network. If there has to be multiple packets on a link, we try to find possible shortest one to minimize network usage. Figure 5 helps more

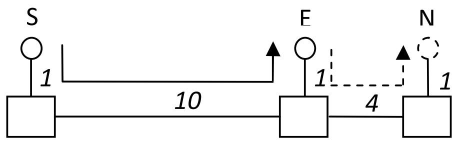

Fig. 4. Case III

to understand the last case. N is joining to network. It will either connect to S or E, or E will connect to N. The best solution is to connect N to E which gives us minimum stress and stretch.

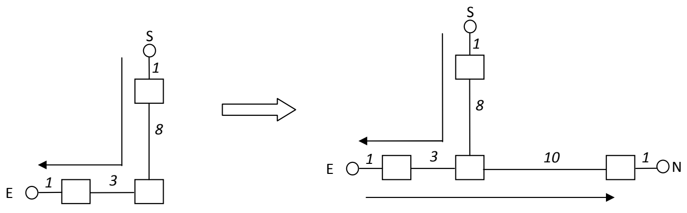

Fig. 5. Another illustration for Case III

*2) Generalizing Virtual Directions for Various Metrics:* We use virtual distance for constructing the overlay multicast tree in the prior sections. Distance when determining virtual directions could be calculated using various metrics such as delay, loss or bandwidth. Even though we focus on delay based directionality in this paper, the concept can be extended to other metrics. Overlay tree should be formed based on the this distance. For example, delay and loss rate between two nodes may be uncorrelated because of background and cross traffic on routers. This may produce different virtual distances and thus different overlay tree in our protocol. By generalizing and customizing virtual direction, we can establish target specific overlay trees to improve some specific performance metrics desirable by applications.

We illustrate a topology in Figure 6. Again S is source, E is existing child and N is a newcomer. Relative distances among these three nodes might be different as shown in Figure 7 when we do distance measurement in terms of delay and loss. As a result, overlay tree will be formed in different ways as in Figure 8. For this specific topology, this difference is caused by the traffic characteristic on router R4.

## *B. Join*

When a node wants to join to the overlay tree, it sends a query to the source. It gets list of its children and learns RTT by probing. Every node has children list and distances to these children. For instance, N in Figure 9 compares these values and decides the way to continue. N determines E1 among four children of S. It repeats same procedure for E1. This process continues until N finds its appropriate parent which is E1 in this case. When N1 wants to join, it will go through the same process, and it will find N as its potential parent.

If the potential parent found by N doesn't have any children. N will connect directly to this node without further query. This

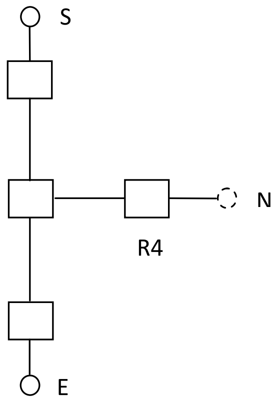

Fig. 6. A sample topology

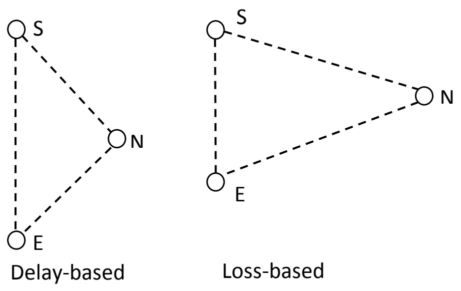

Fig. 7. Relative virtual distances among nodes

case is valid when N is the first node connecting to the overlay tree.

When N finds the correct node, it connects and becomes a child. A node can accept connections up to its maximum degree, which we call ”degree_limit”. Each node has a pre-defined degree_limit. We assume that degree_limit of each node is at least one. If the node (E1) that N decided to connect reached its degree_limit, N connects to its closest child which can accept connection without breaking its degree_limit.

A pseudo-code for the Join procedure is given in Figure 10. Nodes store some state information. Each node has children list and distances to them. They also know their parent and grandparent. For a join process, when N gets ping responses from parent and all children, we first look for if Case II or Case III exists among parent, an existing child and N. We check for each child. If we find Case II which means the new node is between two existing nodes (parent and currently checked child), then proper connections are made, and join process is done. If we encounter Case III, we proceed to next iteration from that child, and repeat the same procedure. If Case II or III is not found, it means that the new node is not in the same direction with none of existing children in this iteration. Then Case I is executed. In Case I, if the potential parent doesn't have free degree slot to accommodate a new connection, new node connects to the closest free child of the potential parent.

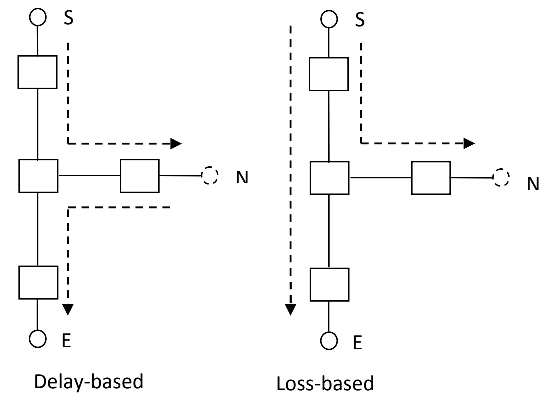

Fig. 8. Differently formed overlay trees based on different virtual distances: delay vs. loss

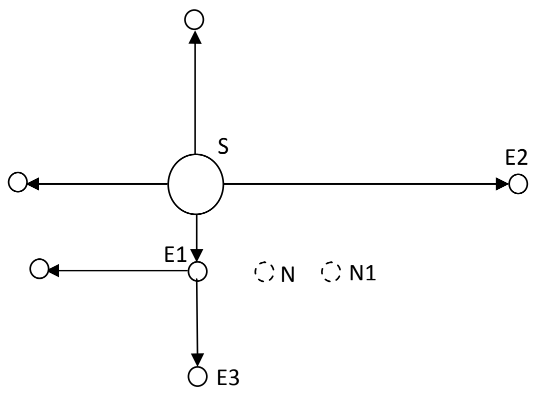

Fig. 9. VDM: A join example

```
S = source
N = new node
Join (N)
  while
    N pings to S and all children of S
    E = find the directional node
      if E is NOT NULL            if Case II or Case III exist
        if N is between S and E                  Case II
          S becomes parent of N
          N becomes parent of E
          exit while
        elseif E is between S and N              Case III
          S = E                  same process is repeated
        end if
      else                                       Case I
        if S has free degree
          N connects to S
        else
          N connects to closest free child
      end if
  end while
```

Fig. 10. Pseudo-code for VDM's join procedure

### C. Reconnection

In VDM, a peer is required to inform its children when it is leaving. When an orphan child gets this leave message, it sends a reconnection request to its grandparent. We start reconnection process at grandparent instead of source to expedite reconnection. Since all orphan nodes are close to each other (their parent were common), it is expected that parent-child relationships will be established among them after rejoin. But, when all nodes attempt to rejoin simultaneously, they will miss each other because all of them are disconnected and they are not in children list of any node. To overcome this problem, grandparent makes a queue for orphan nodes. First coming node will start rejoin process immediately. When the node rejoined the tree, it alerts grandparent to release next orphan node for rejoin process.

If parent and grandparent of a child leave at the same time which is expected to happen very occasionally, an orphan node starts reconnection process at source since it will not be able to find grandparent.

## V. Performance Evaluation

In this section, we evaluate the performance of VDM which uses directionality concept to construct the overlay tree. In order to prove the efficiency of this concept, we compare against another method which connects nearby peers. In this method, which we call as Closest Node Multicast (CNM), a node tries to find the closest node to itself. The join procedure of CNM is similar to VDM's. It starts querying at the source, then finds the closest child. It requests the children list of this child and repeats the same process. This iterative process continues until the closest potential parent is found. CNM utilizes closeness instead of directionality. In the next subsections, we compare VDM and CNM for all metrics that we define for performance evaluation.

### A. Simulation Setup

We use NS-2 [15] to conduct simulation experiments for evaluating our protocol. We generated transit-stub model topology consisting of 792 nodes using GT-ITM [16]. One of the nodes is chosen as source for the multicast tree. The source is assumed to be alive during the entire simulation time, and is known by other peers. Randomly selected 200 of 792 nodes join to the overlay multicast tree. We run the simulation for 10,000s, and dedicate 2,000s for the join process at the beginning. We take 400s as a time interval and define the churn based on that interval. Based on the churn rate, a number of nodes join and leave the tree. For example, if the churn rate is 10%, then 20 new nodes join and 20 of the existing nodes leave in each time interval. The number of nodes in the overlay is retained at 200 by the end of the 400s time interval. At the end of every time interval, we give 100s for tree to come to steady state, then we do the measurements. We expose the tree to churn again in the next time interval after the measurement. This process is repeated until the end of the entire simulation time. For instance, the nodes are renewed almost twice over lifetime under 10% churn. Some nodes may join and leave several times while some never join. There is no super node in that all nodes are considered equal. Degree_limits of nodes are uniformly distributed within the range from 2 to 5.

We simulated the protocols under different churn rates from 1% to 20%. We repeated the simulation experiments 32 times for each churn rate, and we report 90% confidence intervals on our results.

### B. Performance Metrics

To evaluate the performance of our protocol, we focus on four performance metrics:

- *Stress:* Stress is defined as the number of identical packets transmitted on the same link. In IP multicast, stress is always one since a packet goes through a link only once.
- *Stretch:* Stretch is the ratio of path length a packet is traveling in the overlay multicast tree to that of in unicast. Unicast is assumed to have optimal stretch.
- *Messaging Overhead:* We define overhead as the ratio between maintenance messages and data messages.
- *Loss Rate:* Loss rate at a peer is the ratio of number of lost packets to the number of packets supposed to be received in the peer's lifetime.

### C. Simulation Results

We show results of previously defined four metrics with 90% confidence interval for VDM. We investigate the behavior of these metrics versus churn rate. VDM is compared to IP multicast with stress. It shows how much VDM converges to IP Multicast. Stretch is comparing VDM to unicast. Unicast provides smallest delay for peers. Overhead and loss cannot be avoided especially under ad-hoc behaviors of peers. But they should be kept minimal, and they shouldn't increase exponentially with churn rate.

In Figure 11, we show stress vs churn rate. Stress is one of the most important metric for resource usage efficiency. Average stress of VDM is below 1.5 while average stress of CNM is above 1.6. It doesn't change significantly while churn rate increasing.

Figure 12 shows stretch vs churn rate. Stretch is important for efficient content delivery and efficient resource usage. VDM outperforms CNM in terms of stretch. Average stretch is between 3.8 VDM while it is 5.4 for CNM. We have not implemented tree refinement process both for VDM and CNM. VDM aligns nodes in best manner using directionality. High churn rate prevents VDM to behave as it supposed to. So, stretch is increasing in small amount when churn rate is increasing. On the other hand, churn gives CNM the opportunity to adjust nodes in a better alignment. So, stretch is decreasing slightly for CNM with churn. But, still it is higher than VDM.

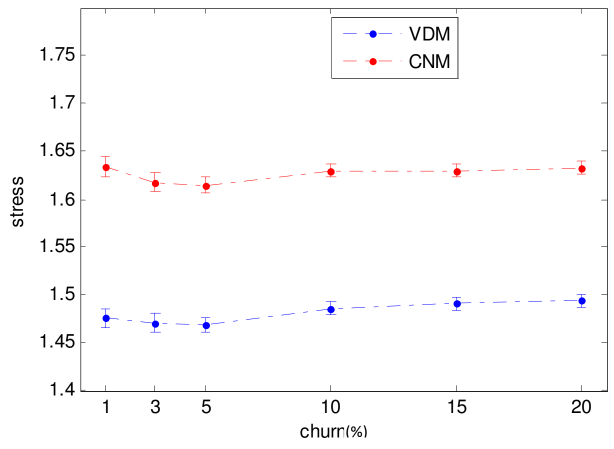

Fig. 11. Stress vs. Churn

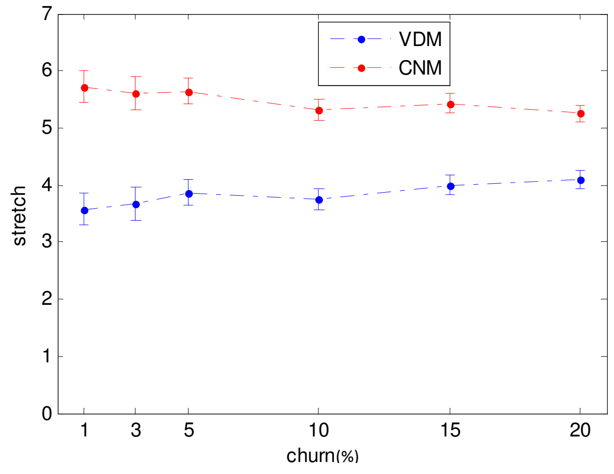

Fig. 12. Stretch vs. Churn

Figure 13 shows the comparison between VDM and CNM for overhead. Overhead should be kept small to put less load to network. It cannot be prevented from increasing with an increase in churn rate, but it shouldn't be exponential. Figure 10 depicts that overhead increases linearly as churn rate increases. Overhead is around 2.5% for VDM when churn rate 20%.

Figure 14 shows average loss rate for all nodes. End users are especially interested in continuity and quality of streaming. High loss rate dissatisfies end users. Average loss rate is below 3% for VDM under 20% churn.

## VI. Conclusions and Future Work

In this paper, we proposed a new overlay multicast protocol, Virtual Directional Multicast (VDM), that uses directionality instead of closeness among nodes to construct the multicast tree. By using the concept of directionality, VDM attempts to build its overlay tree congruent to the underlying network so that network resources are utilized efficiently while satisfying end-users in terms of perceived quality.

Simulation results showed that VDM achieves better performance compared to a scheme, Closest Node Multicast (CNM),

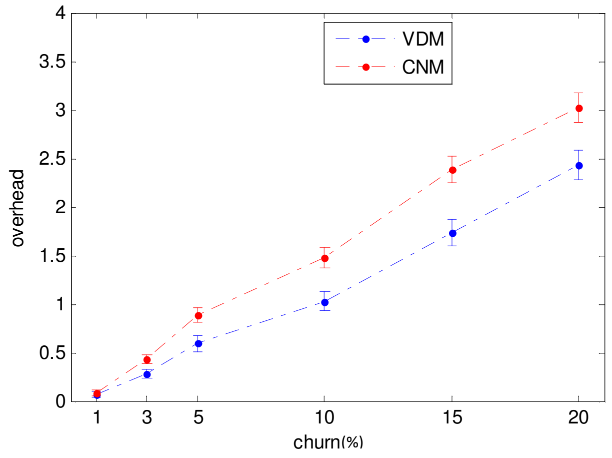

Fig. 13. Overhead vs. Churn

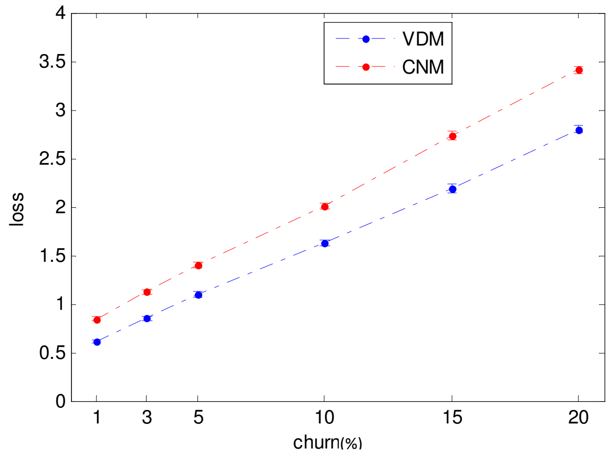

Fig. 14. Loss vs. Churn

using closeness among nodes in terms of various metrics like stretch, stress, loss and overhead. VDM improved the network's stress, which is important for the underlying network infrastructure the overlay multicast is working on. VDM also improved the path stretch which effects both overlay tree participants and physical network. Another improved metric is packet loss that is important for applications with real-time and/or reliable. Finally, we also showed that VDM causes less messaging overhead, that is a key factor for scalability of overlay multicast applications.

Future work includes extension of VDM protocol to include a tree refinement component, so that the overlay multicast performance gets less affected from high churn. The protocol should be tested on various topologies with larger sizes to see its scalability performance versus topology size. One of the novelties that we proposed is virtual distance which allows VDM to adapt to various targets desired by applications. In this work, we only used delay to calculate virtual distances. We are planning to use other metrics in addition to delay to enhance the capability of handling different network dynamics using our virtualized scheme.

Another possible line of work is to extend the concept of "virtual directions" to 2-D space. In this paper, we assumed that a new node only uses two of the existing nodes and positions itself on a 1-D virtual line with the two existing nodes. We currently do not use information about any of the other existing nodes in determining the virtual directions to attain simplicity in the decision making process. It is an interesting investigation to explore the tradeoff between the complexity of using multiple dimensions (e.g. 2-D or 3-D) for establishing the virtual directions and the accuracy of the virtual directions yielding an efficient overlay multicast tree.

## VII. Acknowledgement

This work is supported in part by National Science Foundation award 0721452.

## References

[1] "IPTV News," http://www.iptvnews.net.

[2] Ajay Mahimkar, Zihui Ge, Aman Shaikh, Jia Wang, Jennifer Yates, Yin Zhang, and Qi Zhao, "Towards Automated Performance Diagnosis in a Large IPTV Network"In *Proc. of ACM SIGCOMM*, 2009.

[3] M. Cha, W. A. Chaovalitwongse, Z. Ge, J. Yates, and S. Moon, "Path protection routing with SRLG constraints to support IPTV in WDM mesh networks" In *Proc. of IEEE Global Internet Symposium*, 2006.

[4] "P2PTV," http://en.wikipedia.org/wiki/P2PTV.

[5] E. Alessandria, M. Gallo, E. Leonardi, M. Mellia, M. Meo, "P2P-TV Systems under Adverse Network Conditions: A Measurement Study" In *Proceedings of IEEE INFOCOM*, pages 100-108, April 2006.

[6] S. Deering and D. Cheriton, "Multicast routing in datagram internetworks and extended LANs," *ACM Transactions on Computer Systems*, vol. 8, no 2, pp. 85-110, May 1990.

[7] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron, "Scribe: A large-scale and decentralized application-level multicast infrastructure" In *IEEE Journal on Selected Areas in Communications*, Oct. 2002.

[8] Y. Chawathe, S. McCanne, and E. A. Brewer, "RMX: Reliable multicast for heterogeneous networks" In *Proc. of IEEE IEEE INFOCOM*, Mar. 2000.

[9] Michael Bishop and Sanjay Rao, "Considering Priority in Overlay Multicast Protocols under Heterogeneous Environments," In *Proc. of IEEE INFOCOM*, 2006.

[10] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth Multicast in Cooperative Environments," In *9th ACM Symp. on Operating Systems and Principles (SOSP)*, pages 298-313, 2003.

[11] V. Venkataraman, P. Francis, and J. Calandrino, "Chunkyspread: Multi-tree Unstructured Peer-to-Peer Multicast," In *6th IPTPS*, 2006.

[12] X. Zhang, J. Liu, B. Li, and Y. Yum, "CoolStreaming/DONet: A Data-driven Overlay Network for Peer-to-Peer Live Media Streaming," In *IEEE INFOCOM*, 2005.

[13] D. Helder and S. Jamin, "End-host multicast communication using switch-trees protocols," In *Proc. of 2nd Workshop on Global and Peer-to-Peer Computing on Large Scale Distributed Sys.*, 2002.

[14] Beichuan Zhang, Sugih Jamin , and Lixia Zhang, "Host Multicast: A Framework for Delivering Multicast To End Users," In *Proc. of IEEE INFOCOM*, 2002.

[15] "The network simulator - ns-2," http://www.isi.edu/nsnam/ns/.

[16] "Gt-itm: Georgia tech internetwork topology models," http://www.cc.gatech.edu/fac/Ellen.Zegura/graphs.html.

[17] Y.H. Chu, S. G. Rao, and H. Zhang, "A Case for End System Multicast" In *Proc. of ACM SIGMETRICS*, 2000.

[18] P. Francis, "Yoid: Extending the Multicast Internet Architecture" In *White paper http://www.aciri.org/yoid*, 1999.

[19] J. Liebeherr, M. Nahas, and W. Si., "Application-layer multicasting with delaunay triangulation overlays" In *IEEE Journal on Selected Areas in Communications*, vol. 20, no 8, pp. 1472-1488, Oct. 2002.

[20] D. Pendarakis, S. Shi, D. Verma, and M.Waldvogel, "ALMI: An application level multicast infrastructure" In *Proceedings of the 3rd USENIX Symposium on Internet Technologies and Systems*, 2001.