Virtual Distance: A Generalized Metric for Overlay Tree Construction

Suat Mercan
Computer Engineering
Zirve University
Gaziantep, Turkey
suat.mercan@zirve.edu.tr

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

***Abstract*—Overlay multicasting enabled many new overlay and peer-to-peer applications such as P2PTV, large-scale content sharing, and live video streaming. Sensitivity of these applications differs against various network performance metrics such as delay, loss, or bandwidth. We propose a generalized method of calculating overlay trees to increase user-perceived quality of performance-sensitive applications. We define and use the concept of virtual distance for constructing overlay trees. Abstracting applications' sensitivity to various performance metrics within the virtual distances, we aim to find the most appropriate parent for a peer according to the application's purpose. Calculating the virtual distance based on different criteria, but without protocol modification, makes the overlay multicast protocol satisfy different quality expectations. We show by simulation experiments that the protocol automatically calculates overlay trees based on delay or loss, depending on which is more important for the application under consideration.** [1] [2]

***Keywords*-overlay multicast; peer-to-peer; peer-to-peer TV; path stretch**

I. Introduction

Recently emerged Internet applications such as IPTV [1]–[3], tele-conference and online education requires group communication, also known as multicast. The fact that the Internet bandwidth has become capable of carrying data-rich multimedia applications brought the expansion of Internet usage as well. The IP convergence is progressing and content providers are increasingly transporting multimedia content over the Internet. Multimedia streaming and live video distribution applications such as IPTV, P2PTV [4], [5] are already constituting a significant portion of the Internet traffic and expected to grow further in near future. This unavoidable trend of converging video and multimedia traffic on to the Internet is calling for mechanisms with efficient and scalable transfer of content to many receivers from a single source. Such content delivery to many receivers is desired to be seamless to the multi-provider operation.

Many researchers have put their research focus on achieving a robust and efficient way of sending traffic via multicast. Network layer multicast, a.k.a. IP multicast [6], attracted the attention for years; however, it has not become a widely used protocol for end-users because of its various deployment issues. IP multicast was proposed to provide efficient group communication, and can be implemented by integrating additional algorithms and tables to routers. Even though it provides bandwidth efficiency, ISPs are reluctant to support it since it introduces extra workload and complicates network management. Since IP multicast does not get much support from network operators, application layer multicast (ALM) [7], [8], [11], has emerged as a promising alternative solution 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. Only the end-hosts constitute the multicast group which moves functionality from the network layer to the application layer. This makes ALM protocols easy to deploy across multiple ISP domains and underlying network technologies.

ALM enabled many new overlay and peer-to-peer applications such as P2PTV, large-scale content sharing, live video streaming, and video conferencing with more than two users. Thus, the diversity of overlay applications via ALM increased significantly. Sensitivity of such applications differs against various network performance metrics. For instance, smaller *delay* is crucial for video conferencing which includes interactivity while it is not so significant for video streaming that is sensitive to *loss*. There are also applications, such as high-definition TV, requiring a minimum *bandwidth* to function properly. Responding to these varying application-specific requirements or sensitivities is a key step in bridging the gap further between the overlay applications and ALM protocols.

We propose a generalized method of calculating overlay trees to increase user-perceived quality of performance-sensitive applications. We define and use the concept of "virtual distance" for constructing overlay trees. The virtual distance can be generalized to metrics other than delay, such as loss or bandwidth. Abstracting applications' sensitivity within the virtual distances, we aim to find the most appropriate parent for a peer according to the application's

[1]Dr. Suat Mercan was with the CSE Department of the University of Nevada, Reno during the course of this work.

[2]This work was supported in part by National Science Foundation awards 0721452 and 0721612.

978-1-4673-2713-8/12/$31.00 ©2012 IEEE

purpose. *Our key goal is to automatically calculate overlay multicast trees such that they can be seamlessly customized to applications' performance goals* We embed the virtual distance method in our previous overlay multicast tree protocol, VDM [12], and show that the protocol automatically calculates overlay trees based on delay or loss, depending on which is more important for the application under consideration.

We organize our paper as follows: We survey related work in Section II. In Section III, we briefly describe Virtual Direction Multicast (VDM). Section IV gives a detailed description of the proposed virtual distance concept. Simulation setup and results of a comparative performance evaluation are presented in Section V. Finally, in Section VI, we summarize our work with conclusions and possible future work.

II. Related Work

Application layer multicast is flexible and easy to deploy, but its performance heavily depends on some of its design techniques. Numerous algorithms have been proposed using different techniques to achieve a successful overlay structure for live video streaming. 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.

In overlay tree structure, a peer can choose one of many possible paths. This path can be calculated using various metrics between two peers. Most of the overlay methods and Network Coordinate (NC) systems are using delay to locate peers. Then, they construct overlay tree based on this estimation. There are some other techniques that exploit loss rate or bandwidth for tree construction. Our work differs from others in a way that we attempt to make an abstraction in terms of input metric to construct more efficient and application friendly overlay trees. Though this generalization method could be used with other overlay techniques, we use it with [12] in this work. By generalization method, we try to improve user-perceived quality and resource utilization using the same amount of resources.

A peer in overlay serves its children while being served by its parent. Bandwidth capacity between two peers should be enough to establish a virtual link. Because of this limitation, the optimal overlay tree may not be built. Authors in [13] propose a new bandwidth estimation technique for dynamic path selection to optimize user-perceived quality. They claim that selecting path based on available bandwidth should give better result than other metrics.

In [14], they focus on quality improvement for VOIP. The paper first proposes a retransmission protocol for loss rate improvement. If the problem persists in path then it looks for path optimization using combination of delay and loss rate. It also investigates the performance of different distance calculation methods such as hop count, best route, expected latency and loss rate.

iPlane nano [15], a modified version of iPlane, introduces a system for end-to-end measurement. It is a small sized information data set usable by other applications. They aim to provide a lightweight metric prediction system for large scale P2P networks. Real time loss rate estimation between two points may not be as quick and easy as delay. There are specific measurement studies [16], [17] on this subject. These systems can be used as a third party service provider. We are more interested in showing the advantage of using generalization method rather than measurement study for this work.

There exists many well defined overlay multicast tree protocols such as Scribe [7], SplitStream [10], Narada [8] and CAN [9]. In this work, what we are doing is not proposing a new overlay multicast tree scheme as we did in [12] in which we compared it to some existing protocols. Instead, we propose a tuning or refinement procedure that can be used with some other overlay techniques to be able to improve user satisfaction in terms of desired metric.

III. Virtual Direction Multicast

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 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. It is possible to establish "virtual directions" based on performance metrics such as loss or bandwidth.

A pseudo-code for the Join procedure is given in Figure 1. 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

```
S = source
N = newnode
Join
  while
    N pings S and all children of S
    E = find the directional node
    if E is not NULL        (if Case II or CaseII exist)
      if N is between S and E        (Case II)
        S becomes parent of N
        N becomes parent of E
        exit while
      else if 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 if
  end while
```

Figure 1. Join procedure

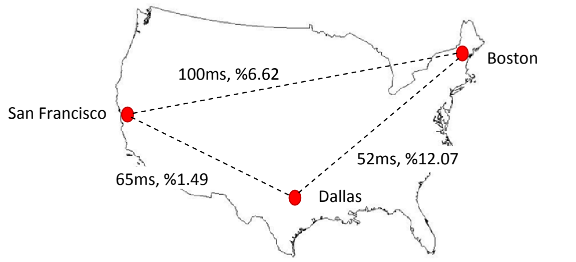


Figure 2. Delay and loss rate measurement among San Francisco, Boston and Dallas

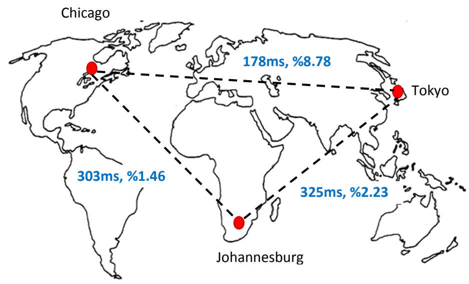


Figure 3. Delay and loss rate measurement among Chicago, Tokyo and Johannesburg

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.

IV. Generalization of Virtual Distance

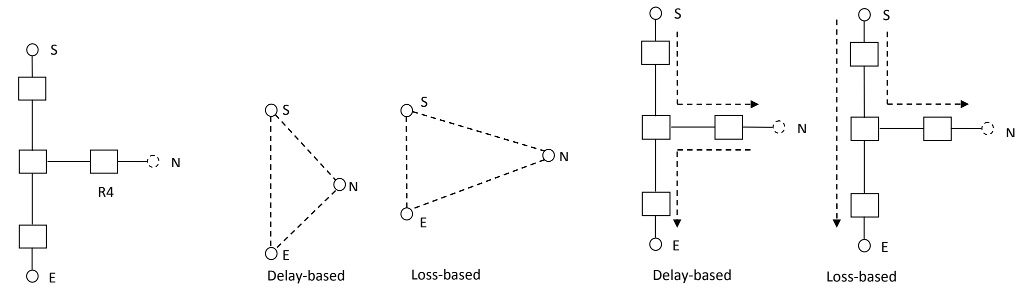
Live multimedia streaming is a real-time application that requires *minimal delay*, where the delay is defined as time needed for a packet to reach its receiver(s). The data packets should ideally traverse the minimum path while being transferred from source to destination. Overlay tree design should provide possible minimum delay for each multicast receiver. However, delay and loss rate between two nodes may be uncorrelated because of background and cross traffic on routers. So, a peer might experience high loss rate on a good path in terms of delay. Sensitivity of multimedia applications differs against various network performance metrics such as delay, loss, or bandwidth. This requires to take other factors into account when building overlay tree.

A key property of VDM is the capability of virtualizing the underlying network in different ways. It is possible to establish "virtual directions" based on performance metrics delay, loss or bandwidth. Different values of these metrics 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. Calculating the virtual distance based on different criteria, but without protocol modification, makes the overlay multicast protocol satisfy different quality expectations. Our key goal in this part is to automatically calculate overlay multicast trees such that they can be seamlessly customized to applications' performance goals.

In order to corroborate the generalization method, we took simple measurement statistics from [18]. It shows latency and loss rate among three cities in United States and in three different countries. Values among San Francisco, Boston and Dallas are shown in Figure 2. Ratio among three values for latency and loss rate is different, thus overlay tree to be constructed among three nodes in these cities will be different. As an another example, we look at the measurements among Chicago, Tokyo and Johannesburg, values are shown in Figure 3, which also gives different ratio for latency and loss rate.

We also took sample inter-PoP measurement dataset from [16] which has latency and loss rate information. From this dataset, we pick three points A, B, C among the links whose loss rate is not zero. We look at delay of A-B (d1) and B-C (d2), and loss rate of A-B (l1) and B-C (l2). When we compare the ratios d1/d2 and l1/l2, 44% of this dataset is inversely correlated. And, the rest doesn't give the same ratio.

We illustrate a topology in Figure 4.a. 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 4.b when we do distance measurement in terms of



a. A sample topology b. Relative virtual distances c. Differently formed overlay trees

Figure 4. Overlay Topology Construction Based on Delay and Loss.

delay and loss. As a result, overlay tree will be formed in different ways as in Figure 4.c. For this specific topology, this difference is caused by the traffic characteristic on router R4.

We propose a generalized method of calculating overlay trees to increase user-perceived quality of performance-sensitive applications. We define and use the concept of "virtual distance" to determine "virtual direction" for constructing overlay trees. Abstracting applications' sensitivity within the virtual distances, we aim to find the most appropriate parent for a peer according to the application's purpose. We embed the virtual distance method in our protocol, VDM, and show that the protocol automatically calculates overlay trees based on delay (VDM-D) or loss (VDM-L), depending on which is more important for the application under consideration.

V. Performance Evaluation

In this section, we evaluate the performance of VDM-D (delay-based) and VDM-L (loss-based) in order to show the efficiency of the virtual distance concept in automatically customizing the overlay tree for application-specific performance goals. We analyze protocol behaviors as we vary the churn rate in the overlay network.

A. Simulation Setup

We use NS-2 [19] to conduct simulation experiments for evaluating our protocol. We generated a transit-stub model topology consisting of 792 nodes using GT-ITM [20]. Degree_limits of nodes are uniformly distributed within the range from 2 to 5. Each physical link in topology is assigned a random error rate between 0% and 2%. End-to-end loss rate between two points depends on number of links between them. 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 the 792 nodes constitute the overlay multicast tree.

We run the simulation for 5,000s, and dedicate 2,000s for the join process at the beginning. We take 200s 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 200s time interval. At the end of every time interval, we give 50s for tree to get stabilized, 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. Some nodes may join and leave several times while some never join. There is no super node in that all nodes are considered equal, and their degree_limits are assumed to be drawn from the same distribution.

We simulated the protocols under different churn rates from 1% to 20%, and repeated the simulation experiments 10 times with different seed values.

B. Performance Metrics

We are interested in efficiency of data delivery path and service quality that end-users are experiencing. In order to quantify these two targets, we focus on four performance metrics. Stress is the major factor for data delivery efficiency while stretch and overhead also have some impact. Service quality is basically measured with loss rate and delay.

- *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 present results for performance comparison between overlay trees constructed using VDM-D and VDM-L. We investigate the behavior of these metrics versus churn rate

978-1-4673-2713-8/12/$31.00 ©2012 IEEE

for both protocols. The variations especially in stretch and loss rate based on utilized technique will affirm our proposal. We expect that VDM-L reduces loss rate while trading off stretch and that VDM-D gives better results for stretch.

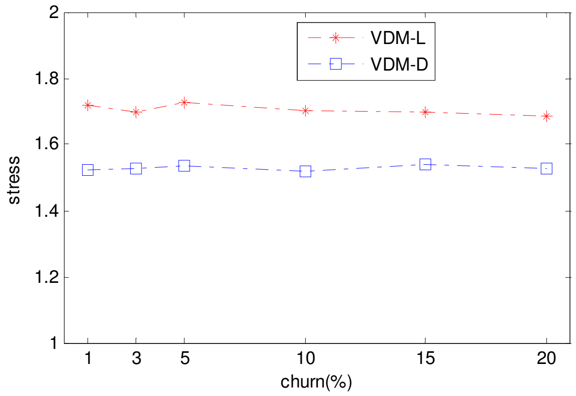


Figure 5. Stress vs. Churn.

In Figure 5, we show stress vs churn. Stress is one of the most important metrics for resource usage efficiency. Stress does not change significantly while churn rate increasing. Average stress is around 1.5 and 1.7 for VDM-D and VDM-L, respectively. The closeness of the stress for the two protocols is expected as the virtual distance does not focus on the stress. However, it is also expected that stress is a little higher in VDM-L since delay is known to be more correlated with the physical distance between nodes, and thus the overlay tree becomes closer to the IP multicast tree in VDM-D.

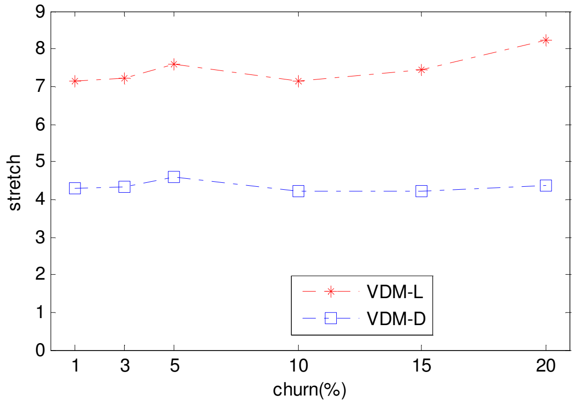


Figure 6. Stretch vs. Churn.

In Figure 6, we show stretch vs. churn. Stretch is important for efficient content delivery and efficient resource usage. Stretch doesn't get affected much with churn rate. Average stretch value is around 4 for VDM-D while it is around 7 VDM-L. We can infer from the graph that path length to source for end-users is reduced when using delay as basis for distance calculation. The results in Figure 7 shows a clear differentiation of the overlay tree based on which metric, delay or loss, the user application might choose.

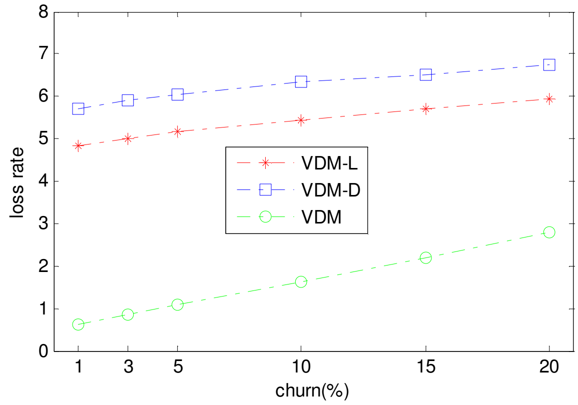


Figure 7. Loss Rate vs. Churn.

Figure 7 shows average loss rate vs. churn. End-users are especially interested in continuity and quality of streaming. High loss rate dissatisfies end-users. Loss in this graph is caused by packet drops over path and disconnection because of churn. Packets are dropped over links according to their error rate. Churn causes the loss rate to increase. The graph proves that VDM-L improves the loss rate compared to VDM-D. Loss rate for both could be considered high, but each link in this setup is assigned a loss rate on purpose as we wanted to observe how much customization the virtual distance concept can achieve in terms of loss rate. To give an idea about how much excessive loss the results have, we have plotted another result for "VDM" where link error rates are set to 0%. In this case, loss rate is caused only by disconnections and is relatively low compared to the other two cases.

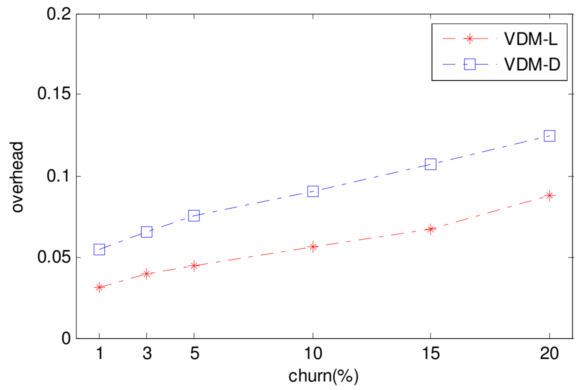


Figure 8. Overhead vs. Churn.

We finally looked at the overhead. We used the same number of probing messages to measure delay and loss rate even though delay can be measured with less number of messages. The overhead for VDM-L is a little less the the VDM-D, because the number of lost packets in VDM-L is fewer which makes denominator greater in the definition. Figure 8 depicts that the overhead increases sublinearly as the churn rate increases since the nodes send additional probing messages to be able to reconnect.

Our concern was to obtain better performance results for certain types of metrics which may be more important for different applications. When we think of all the results together, VDM-D, uses delay for distance estimation, and improves stress and stretch while giving higher loss rate. It could be used for delay sensitive applications. On the other hand, VDM-L achieves better performance in terms of loss rate. It can be chosen for more delay tolerant but loss sensitive applications.

VI. Conclusions and Future Work

In this paper, we proposed a method to generalize the virtual distance between overlay nodes for automatically calculating overlay trees custom to specific application performance targets. The proposed method is implemented on an overlay multicast tree protocol, VDM. By using the generalization concept, we aim to build target specific overlay trees that provides the ability to improve user perceived quality for specific purposes.

We used two different metrics, delay and loss, for calculating the virtual distances and experimented with two version of VDM: VDM-D and VDM-L. Simulation results showed that VDM-D achieves better performance in terms of path length and stress while degrading the loss rate. On the other hand, VDM-L improves the loss rate performance, as expected, by sacrificing from stress and stretch.

Future work includes implementation of the generalization method on a testbed. Another line of work is that to use other metrics, such as bandwidth or jitter, in addition to delay and loss rate to enhance the capability of handling different network dynamics using our virtualized scheme.

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

[9] S. Ratnasamy, M. Handley, R. Karp and S. Shenker, "A scalable content-addressable network" In *Proc. of ACM SIGCOMM*, 2001.

[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)*, 2003.

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

[12] S. Mercan and M. Yuksel, "Virtual Direction Multicast for Overlay Networks" In *Proceedings of IEEE International Workshop on Hot Topics in Peer-to-Peer Systems (HOTP2P)*, 2011.

[13] M. Jain and C. Davrolis, "Path selection using available bandwidth estimation in overlay-based video streaming" In *Lecture Notes in Computer Science, Comput. Netw.*, vol.52, No.12, pp.2411-2418, Springer.

[14] Y. Amir, C. Danilov, S. Goose, D. Hedqvist, and A. Terzis, "A1-800-OVERLAYS: using overlay networks to improve VoIP quality" In *Int. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, 2005.

[15] H. V. Madhyastha, E. Katz-Bassett, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "iPlane Nano: Path prediction for peer-to-peer applications" In *NSDI*, 2009.

[16] http://iplane.cs.washington.edu/.

[17] http://www.netdimes.org/new/.

[18] http://www.akamai.com/html/technology/dataviz2.html.

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

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