International Journal of Information and Education Technology, Vol. 1, No. 1, April 2011
ISSN: 2010-3689

Applying Low-Latency Peer-to-Peer Muliticast to Online Collaborative Language Learning

Sun Xiao and Sun Min

Abstract—**Peer-to-Peer computing technology brings satisfactory distance education to a large number of people at anytime anywhere with a low cost. While many online learning systems connect students asynchronously or provide them with recorded lectures, few systems focus on facilitating synchronous collaborative work among learners. Low latency requirement, dynamic group members and limited out-degree of nodes pose great challenge on P2P solutions to virtual interactive learning environments. Most application level multicast protocols are heuristic algorithms of minimum spanning tree, aiming to reach the optimal parameters like latency concerning all nodes. However, few studies are focusing on providing stringent low perceived latency based on specific features of audio group communication. In this paper, we propose a gossip-enhanced multicast protocol called GellCast for online language learning to provide "natural" group discussions in problem solving. We then present an online collaborative language learning system called PPCOEC, a distributed Virtual interactive Oral English Classroom, which is powered by GellCast. Making realistic presumptions on processing delay at each node, GellCast guarantees stringent low latency among discussion participants while providing relatively low latency among listeners. Hence, it provides satisfying low perceived latency to all users. Simulation results show the feasibility and efficiency of the approach.**

Index Terms—**P2P; distance education, virtual interactive environment, audio group communication, language teaching.**

I. Introduction

Recent researches suggest that computer-mediated learning and face-to-fac learning often achieve similar learning outcomes [1, 2 and 3]. Further studies specify interaction among others to be the key factors in affecting computer-mediated learning [4, 5 and 6]. Encouraged by the work, King et al. [7] design a P2P-based teaching platform called ePBL to realize interprofessional health science teaching. Bader-Natal [8] launched a web application called Grockit to provide live interactive learning activities in a virtual group study format

Believing interaction among learners themselves and between teacher and students is the key to online learning, and especially to virtual English leaning, we launched an online interactive English course for hundreds of participants.

As we are targeting this service to a wide audience, the ability to serve high volume concurrent interactive participants is the first goal we tend to meet. Basically, there are 2 different approaches: the server-based solution and the P2P-based one. In the server-based solution, a set of edge servers are strategically deployed and managed so that high bandwidth and computation demands can be wisely distributed. In the P2P-based one, each user is a client and 'server' at the same time. The server-based approach has a better QoS guarantee but is too expensive for a non-profit service like our virtual English class. Hence, we chose the P2P-based scheme.

The dominant component of most online courses is lectures, which can be realized by constructing a single-source multicast tree [9]. Unlike them or the traditional English class focusing on grammar or drills, our virtual English class tends to create "natural" discussions on certain issues like economy, politics or campus life, among users to improve their language competence while using it in collaborative problem solving. They may argue or even quarrel about those problems in this virtual class, which is different from conversations with clear structures like the question-reply pattern in most "interactive" audio chat systems.

To create "natural" interactive environment for the virtual English class, we need to consider the following features apart from good scalability mentioned above. Firstly, we must provide stringent low latency for discussion participants in the class. Secondly, we tend to provide "smooth" discussion even when some listener suddenly strikes in. Thirdly, class members can strike in the discussion "naturally", hence we avoid using floor control module in the design. Finally, nodes users may have limited resources which must be considered as limit out-degree of nodes in design.

After the observation of both face-to-face and computer-mediated English teaching, we found only a small portion of the class members are discussion participants at a given time, which is supported by studies of Blundell [10] and Zimmermann[11]. We propose GellCast, a Gossip-enhanced Low latency multicast as the fundamental protocol to realize the multi-party audio interaction in our virtual English classroom. The core idea of GellCast is to provide stringent low latency among discussion participants (called interactive nodes in GellCast) while lower latency among other members as much as possible.

Manuscript received March 18, 2011.

SUN Xiao is with the Department of Equipment Command and Management Ordnance Engineering College, Shijiazhuang, Hebei Province, China (phone: 0311-879-94663; e-mail: sunxiao@hotmail.com). He is a IACSIT senior member.

SUN Min is with the English Department of the School of Humanities and Social Sciences, National University of Defense Technology, Changsha Hunan Province, China(e-mail: sunmin010@gmail.com).

II. Related Work

A. Issues of Latency in Voice over the Internet

Latency in voice over the internet (VoIP) consists of packetization delay, pre-processing delay (like silence-suppression and compression), network delay, decompression delay and finally playout-buffering delay.

Many studies into user tolerance of roundtrip latency in audio conferencing agree that if other factors excluded, audio interaction requires it to be 300ms at most [12 and 13].

As end-to-end latency increases, it might be misunderstood by users as extended pause in speech, causing confusion and hence results in their loss of synchronization with the conversation.

Among the components of latency in VoIP, network delay is the least predictable and most-dominant part for delay in voice over the internet. Jiang et al. [13] studies the components of audio transmission delay on the internet, and concludes that the maximum of network-transmission delay is 180ms if end-to-end delay is expected to be below 300ms to realize interactive applications. To realize real-time communication over application level multicast, the key to maintain low network delay is to construct an efficient data-transmission overlay.

B. Application Level Multicast Protocols Concerning Interactive Applications

Although P2P-based distributed interactive applications greatly vary, they share many of the same requirements like low latency, good scalability, limited out-degree of nodes and dynamic group member management. Those requirements pose great challenge in designing algorithms to create and maintain efficient (low latency) data distribution paths with light workload especially when taken the underlying physical location of users into consideration.

Among many application level multicast schemes, CAN-Multicast [14], a P2P-based architecture based on Distributed Hash Table (DHT), constructs overlay that usually doesn't match the physical network and cannot be optimized dynamically, hence cause unacceptable long latency. NICE [15] can cluster adjacent nodes to construct low-latency hierarchic data sharing multicast tree, but fails to limit the out-degree of nodes, hence may result in lower QoS for application users. Narada [16] can construct single-source multicast tree aiming at low latency, but requests every node to maintain the state information of all node. This poor scalability constrains Narada to small sized conferencing.

C. Application Level Multicast Protocols Targeting Real-time Audio Group Communication

Traditional multicast protocols optimize their constructed overlay by converging nodes according to parameters like latency and to react to changes in the underlying physical network [14, 15 and 16].

In interactive audio communication, parties in a conversation tend to consider what they have heard or haven't heard as "what is happening". Only when the reply for the other party has not been received as expected, will the speaker sense the latency. Some researches use this character to design the heuristic algorithms specifically for audio group communication.

Application-Level Network Audio-Conferencing (ALNAC): Blundell et al. [10] discovered from a preliminary experiment that listeners generally cannot sense the unnaturalness of a conversation as long as the latency is not more than 1000ms. Based on this discovery, they designed ALNAC, a light-weight application level network audio conferencing routing protocol, to optimize audio-packet delivery for those conferencing participants who are most sensitive to perceived latency while minimizing the impact of such optimization on members as listeners that are less sensitive to it at a given time. They also find who will be the next speaker is highly related to a few previous speakers, and hence developed prediction algorithms and constructed a single source multicast tree to allow the current speaker first to send audio packets to those predicted nodes by flooding.

ALNAC is suitable for audio interaction with clear structures like formal meetings, but fails to adapt to natural conversation with frequently changing participants. And the designer admits the scheme needs a certain amount of time to recover from prediction faults.

Adaptive Core-based Tree for Interactive Virtual Environments (ACTIVE): Zimmermann et al. [11] discovered that at any given time, only a small number of group members take part in the conversation, called active users. They proposed ACTIVE algorithm to realize audio chats in distance education.

ACTIVE firstly distinguishes active users (speakers) form passive users (listeners), then builds a core-based tree that includes all group members, and dynamically optimizes the tree to minimize the average delay among active users. In simulation, Zimmermann et al. compared ACTIVE's performance to trees generated by Prim's minimum spanning tree algorithm (MST) with same physical network in a NS-2 environment. And they found ACTIVE provide smaller delay among active users in most cases while providing good service for all nodes compared with MST.

ACTIVE architecture is suitable for structured conversations like question-answer or role-based discussion. It has not clearly stated how to decide an active node, and the change standard between active user and passive user. If a passive user far from the root suddenly "speaks", the accumulated latency will greatly affect most users in the group. What more, frequent change of multicast tree may cause loss of audio packets, resulting in poor QoS.

III. Design of GellCast

The core idea of GellCast is to construct and optimize overlay topology matching with the underlying physical network applying Gossip mechanism, to guarantee stringent low latency for interactive nodes by providing direct audio data transmission among them and to provide satisfying latency for ordinary nodes by selecting the available shortest path of data transmission.

A. Classifying Nodes

GellCast distinguishes interactive nodes, discussion participants at a given time, from ordinary nodes, who are temporarily listeners.

B. Constructing Basic Control Topology

GellCast constructs a mesh basic control topology, in which each node maintains information of adjacent nodes in a basic cache. The topology is maintained via periodic exchange of information in node's basic cache between neighbor nodes. As is shown in Table I, a node exchanges information in the basic cache of a randomly chose adjacent nodes, update its basic cache by selecting relatively closer neighbors from that new information and hence optimize data transmission link to realized the match between the P2P topology and the physical network.

TABLE I. BASIC STRUCTURE OF THE GOSSIP-ENHANCED TOPOLOGY MANAGEMENT PROTOCOL

Active Thread	Passive Thread

do forever
wait (T) //wait for interval T
Peer = myview. selectPeer() //select a neighbor Peer
Send(Peer, view)
//send view in local cache to Peer
PeerView = receive() //receive view
update(view, PeerView, R-Policy)
// update view using R-Policy
	do forever
	PeerView = receive()
	// receive view sent by Peer
	Send(Peer, view)
	// send local view to Peer
	update(view, PeerView, R-Policy)
	// update local view using R-Policy

C. Constructing Core Area of Overlay

In GellCast, interactive nodes form a unique gossip network, which is call core area of overlay topology. In this network, each interactive node needs to maintain an interactive cache for information of its interactive neighbors as well as its basic cache as is shown is Table II(a).

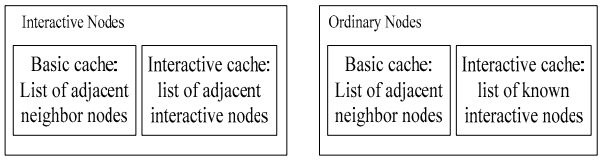
When a certain interactive node "speaks", it selects closest nodes as many as K, which is determined by its node degree (Dmax=K), to distribute the audio-packets. If a certain node receives several same audio-packets, it chooses the data transmission with the lowers latency and informs others nodes to stop data transmission.

If a certain interactive node is capable of sending data to other nodes than its interactive neighbors, then it will select the closest ordinary neighbors from its basic cache to do data transmission.

D. Constructing the Rest Area of Overlay

The ordinary nodes form the rest areal of overlay topology in GellCast. As is shown in Table II (b), ordinary nodes maintain some known interactive nodes in interactive cache while maintaining its basic cache. They periodically probe the interactive nodes listed in the interactive cache to test if they still function as interactive nodes and update the cache by exchanging information with interactive nodes in the cache. When an ordinary node suddenly "speaks", every node adds it to interactive cache so that the system can guarantee low latency data transmission among interactive nodes.

TABLE II. NEIGHBOR INFORMATION MAINTAINED BY INTERACTIVE NODES AND ORDINARY NODES



(a) Interactive Nodes (b) Ordinary Nodes

The ordinary node usually receives audio data from the closest neighbor. When the node fails to function or the latency is too high, it may send data transmission request to the neighbors in its basic cache and select the one with lower latency to do it after comparing their reply time.

E. Role Change of Nodes

GellCast presumes there is no deceit from nodes. Once a node "speaks", it spontaneously becomes an interactive node. To dynamically control the number of active nodes, we apply a Threshold Point (TP) mechanism.

When a nodei joins the group, it is given a TP value recorded as TPi by its client end. When nodei speaks, the TPi will be reset to 0; If nodei remains "silent", TPi will become bigger and bigger as time goes by. When TPi >TPt (the threshold point value), an interactive node will change into an ordinary one.

Dynamic change of TPt value can control the number in interactive nodes to suit for different applications. In extreme situation where TPt is too large, all nodes can be interactive node, which has great impact on overlay construction and protocol performance. In simulation, we limit the number of interactive nodes to around 15% of all nodes, based on observation of both face-to-face and distance English course.

After becoming an interactive node, it updates its interactive cache by exchanging information with interactive nodes in its original one (When a node joins the network, it can receive information of at least one interactive node). In this way, the node can get a new interactive cache containing adjacent interactive nodes to guarantee low latency data transmission with other interactive nodes.

When an interactive node switches to an ordinary one, it will stop exchanging information about the interactive cache with other interactive node. Only when some interactive node fails to function or switch too, will the node request to get information of other interactive nodes to maintain the information of enough number of interactive nodes. Normally an ordinary node doesn't necessarily receive audio-packets from interactive ones, but select the closest node as its data source to reduce latency.

F. Node Join and Leave

When a node joins the network, it gets information of at least one interactive node and a list of partial nodes in it. With periodical exchange of neighbor information with those known nodes, it receives a list of adjacent nodes, the basic cache.

When a node leaves the network, it firstly informs neighbor nodes in the basic cache, and those neighbors propagate the leave information to the whole network using gossip. If the node provides audio data transmission, it will

inform those nodes before it leaves. If a node suddenly fails to function, those node in its basic cache will discover and ensure its leave using probe. If a node ensures that its data sender fails to function, it will communicate with nodes in its cache to find the closest neighbor among them as its new data sender.

IV. Experimental Evaluation

A. Simulator

We evaluate GellCast performance using PeerSim, a platform provided by BISON project. PeerSim, a simulator based on components and realized by JAVA, provides better scalability and mobility for P2P network. It uses both cycle-driven and event-driven modules. More details can be found in literature [17].

Studies on Vivaldi network reference frame indicate that if we map network nodes into points in multi-dimensional Euclidean space [18]. The link transmission latency can be approximately simulated as distance between the nodes. Based on this finding, PeerSim simulates large overlay network with a class of points in Euclidean space, and collects metrics like distance between nodes or link transmission latency.

B. Simulation Model

We compared GellCast's performance with ACTIVE using PeerSim in the same scenarios. The main performance metric is latency of different kinds of nodes in two protocols.

We used distance of points in Euclidean space to simulate link transmission latency. While processing delay of nodes in ACTIVE is regulated as 30ms, we randomly assign processing latency ranging from 20ms to 40ms to nodes.

ACTIVE maintains some ratio between active nodes and all nodes according to the size of group. In simulation, they selected active nodes ranging in size from 4 to 20 in uniformly distributed groups ranging in size from 4 to 400. In GellCast, we maintain the ratio by dynamically changing the TPt. In our simulation, the number of interactive nodes in GellCast and that of active nodes in ACTIVE is about 15% of all nodes, and out-degree of each node is limited to be 3.

C. Simulation Results

We simulated GellCast and ACTIVE in different group ranging in size from 50 to 250, which is common size of virtual English classroom, and collects network latency of different kinds of nodes. As is shown in Fig. 1, the mean latency between interactive/active nodes keeps at a low level, which proves both protocol can provide low latency among conversation speakers. Meanwhile, GellCast can provide much lower mean latency of all nodes than ACTIVE, and it's getting more obvious as the size of the nodes increases. The simulation shows the efficiency of GellCast in lower latency of all nodes while guarantee low latency among active nodes.

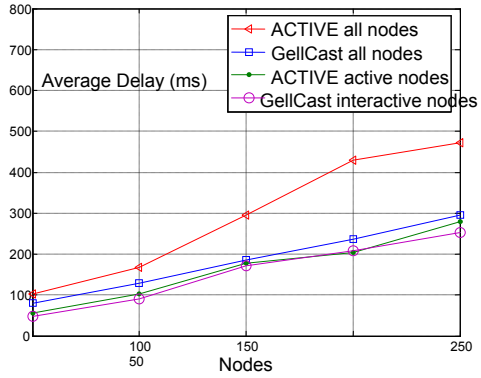


Figure 1. Comparison of latencies among different node in ACTIVE and GellCast

Fig. 2 shows the latency differences of different nodes in ACTIVE and GellCast when the number of the group reaches 100. ACTIVE and GellCast achieve similar low pairwise latency for active/interactive nodes shown in Fig. 3 (b), but our GellCast provides much lower latency among ordinary nodes than that among passive nodes in ACTIVE shown is Fig. (C). In scenarios that active/interactive nodes are relatively stable, GellCast and ACTIVE reaches similar performances shown in Fig. 3 (b). However, in scenarios that active/interactive nodes frequently change, GellCast performs much better in that the optimization on accumulated latency among ordinary nodes greatly lowered its impact on the whole network shown in Fig. 3(d).

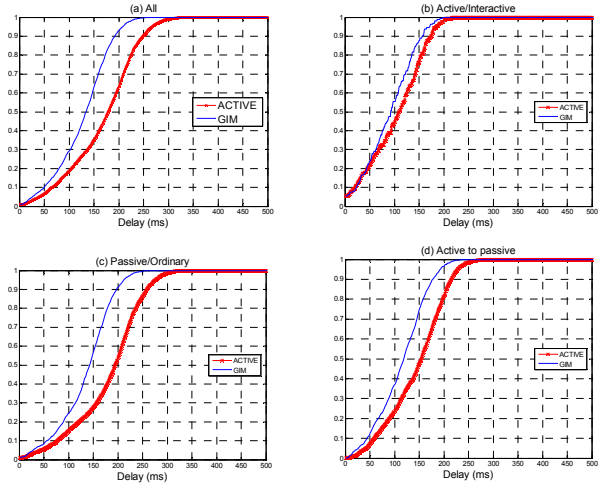


Figure 2. Comparison of latency among different kinds of nodes in ACTIVE and GellCast when N=100

Fig. 3 shows the comparison of latency between different kinds of nodes in two protocols, when the group size reaches 200. With the group increasing, mean latencies among nodes in two protocols all increases shown in Fig. 3(a). The curves of latency among interactive/active nodes of two protocols nearly overlapped when the group size is 200 shown in Fig. 3(b). But there are greater differences in latency among other nodes between ACTIVE and GellCast Fig. 3(c and d). It further shows the necessity of considering latency among other nodes to achieve better performance in the system.

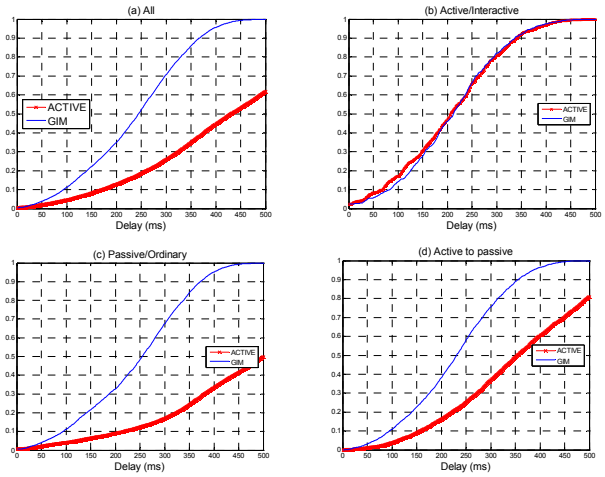


Figure 3. Comparison of latency among different kinds of nodes in ACTIVE and GellCast when N=200

V. Application Using GellCast

Unlike most courses of which the main purpose is to impart knowledge by teachers, language learning especially like the oral English course requires personal exercise and grouped practice via oral work besides lecture of oral English strategies from teachers. Here we describe an online language system called PPCOEC, a P2P-based collaborative oral English Classroom, which has been used in the online oral English course on campus. Built on GellCast, it aims to foster collaboration among participants and higher level of interaction between students themselves and between teachers and students to effectively improve students' communicative skills through oral tasks. It provides an interactive platform where groups of students can "naturally" talk, discuss and even "quarrel" in class. As an online course, it contains modules of lecture, exercise, and assessment. GellCast is implemented to realize inner group communication in the modules of lecture and exercise.

A. System Architecture

We apply a distributed manner in designing the collaborative Oral English Classroom. As is shown in Fig.4, it consists of following components: Authentication Server, Schedule Server, Documents Server, Extended Information Database, Bootstrap Node, and ordinary users.

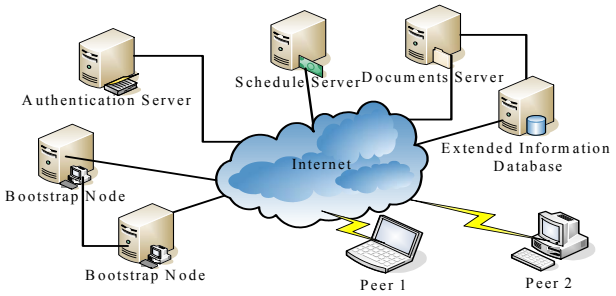


Figure 4. System Infrastructure

The Bootstrap Nodes, performing like other super nodes, are clients with spare computing power and network bandwidth. They are responsible for helping ordinary users to get authenticated, keeping information of ordinary nodes directly connected to them, performing searching and relaying data and traversing NAT and Firewall. Compared with other super nodes, they are more stable, reliable and always online.

To better assure the quality of the course and of the oral English practice, we apply schedule server to arrange courses, document server to array teaching materials and assignments, and Extended Information Database to store and administer audio records completed by attended users, evaluation given to each user, their final scores and other necessary information.

Ordinary users represented by peer1 and peer2 in Fig. 4 refer to users like students and teachers, which will be attached to super nodes after logging in the system. They can be super nodes if the nodes have spare computer power and public IP address. If not, they will be assigned to a super node for future functioning.

B. Distributed Startup Process

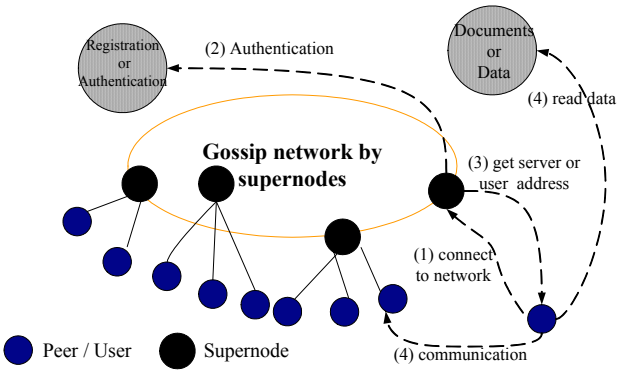


Figure 5. Overlay Network of the System

The users' end-points in our system can be classified as super nodes (including bootstrap nodes) and ordinary nodes in topology, shown in Fig. 5. An ordinary node is a system application's end-point which enables a user, either a teacher or a student, to attend the course for lectures and exercises. If an ordinary node has sufficient CPU, memory, network bandwidth and a pubic IP address, it may be selected as a super node at the startup process.

In starting up our PPCOEC system, the node connects to any one of bootstrap nodes, the IP address and port information of which is hard coded in the application.

The bootstrap super nodes will help the user contact with the registration server, to make sure the username and its password is unique. This super node is also responsible for temporarily storing this node's information (its presence or not, and whether he is in the Lecture model, in the Exercise model or at leisure at present), managing, searching and relaying data for the ordinary node. If the super node is too busy to handle the node's work, it will transfer this node to another super node. An ordinary node maintains a list of super nodes based on history connection in case the present connecting super node fails.

C. Accessing process in Different Models

If a system user wants to attend a lecture for the first time, firstly it needs to login in through its super node. Then the node connects Schedule Server to view necessary

information about lecture plans and also the address and port of Peer teacher after getting its address from the super node (the node itself will locally store this address for later usage). An on-line lecture will hence start, as is shown in Fig. 6 (a).

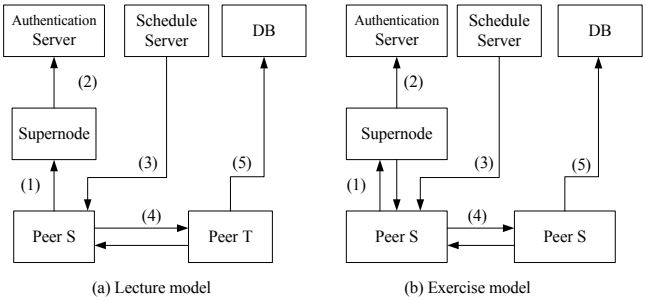


Figure 6. Accessing Process in Different Models

If the system user tends to do assignments and other exercise, firstly it needs to login in as well. After registration, it connects Schedule Server to view the required assignments and exercise. If the exercise is a cooperative one, the node requests its super node to search for the partners (the number of partner will depend on whether the task is a dialogue or a group discussion). After receiving the potential partners' addresses from the super node, he sets up a session with them for the task. When having finished the work and evaluation, one of the attending nodes will send the audio record and evaluation outcome to Extended Information Database and session completion information to its super node and to the Schedule Server. An on-line task is hence completed, as is shown in Fig.6 (b).

VI. Conclusion

We proposed a novel P2P protocol called GellCast targeting "natural" audio interaction for interactive online language learning. While taking priority to stringent low latency among interactive nodes, GellCast provide relatively low latency among other nodes to guarantee QoS in the interactive environments. Simulation results show GellCast outperforms ACTIVE in providing lower latency among all multicast members, and hence outperforms optimal solution generated by the minimum spanning tree algorithm. We also implement GellCast in our virtual collaborative oral English class on campus. Further work will be focused on extending functions like speaker recognition, integrating with existing distance education infrastructure on campus and performance improvement of the system.

Acknowledgment

We wish to express our appreciation to many wonderful people who helped us design, implement and improve our system. We acknowledge (in no particular order) the help of Nick Blundell, Pengyi FAN, Hui WANG and Jin LIU. Many of their thoughts and comments have been incorporated into this work.

References

[1] B. Means, Y. Toyama, R. Murphy, M. Bakia and K. Jones, "Evaluation of evidence-based practices in online learning: A meta-analysis and review of online learning studies" Technical report, U.S. Department of Education, Office of Planning, Evaluation, and Policy Development, Washington, D.C., 2009.

[2] Y. Zhao, J. Lei, B. Yan, C. Lai, and S. Tan, "What makes the difference? A practical analysis of research on the effectiveness of distance education." *The Teachers College Record*, 107(8) PP.1836-1884, 2005

[3] R. Bernard, P. Abrami, Y. Lou, E. Borokhovski, A. Wade , L. Wozney, P. Wallet, M. Fiset, and B. Huang, "How does distance education compare with classroom instruction? A meta-analysis of the empirical literature" *Review of Educational Research*, 74(3), PP.379, 2004.

[4] R. Luppicini, *Online Learning Communities* Ed., Greenwich: Information Age Publishing. 2007.

[5] M. K. Tallent-Runnels, J. A. Thomas, W.Y. Lan, S. Cooper, T.C. Ahern, S. Shaw, and X. Liu, "Teaching courses online: A review of the research" *Review of Education Research*, 76, PP. 93 – 135, 2006

[6] B. Harvard, J. Du, and J. Xu, "Online collaborative learning and communication media" *Journal of Interactive Learning Research*, 19(1), PP. 37-50, 2008.

[7] S. King, E. Greidanus, M. Carbonaro, J. Drummond, P. Boechler, and R. Kahlke, "Synchronous Problem-Based e-Learning (ePBL) in Interprofessional Health Science Education", *Journal of Interactive Online Learning*, 9(2), PP.133-150, 2010.

[8] A. Bader-Natal, "Interaction Synchronicity in Web-based Collaborative Learning Systems" *Proceedings of World Conference on ELearning in Corporate, Government, Healthcare, and Higher Education 2009*, PP. 1121-1129, Vancouver, Canada, October 2009.

[9] W.-K. Xie, Z.-Q. Zhang and C.-P. Lu, "PPClass – A Classroom Lecture Broadcast Platform Based on P2P Streaming Technology" *Proceedings of 2009 International Symposium on Computing, Communication, and Control Singapore*, 9-11 October, 2009, pp.482-488.

[10] Nick Blundell, Norbert Egi and Laurent Mathy. Voice over Application-Level Multicast. in Proceedings of the Second International Workshop on Multimedia Systems and Networking (WMSN 2006), Phoenix, Arizona, 2006.

[11] Leslie S. Liu and Roger Zimmermann. Adaptive low-latency peer-to-peer streaming and its application. Multimedia Systems, 2006, 11(6)(REGULAR PAPER):497–512.

[12] International Telecommunication Union. 1996. One-way Transmission Time (Recommendation G.114).

[13] Wenyu Jiang, Kazuumi Koguchi and Henning Schulzrinne. QoS Evaluation of VoIP End-Points. IEEE International Conference on Communications (ICC 2003), Anchorage, Alaska, May, 2003.

[14] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Application-level Multicast Using Content-Addressable Networks," in Third International Workshop Networked Group Comm., 2001.

[15] S. Banerjee, B. Bhattacharjee and C. Kommareddy. Scaleable application layer multicast. Proceedings of ACM SIGCOMM, August 2002.

[16] S. Banerjee, B. Bhattacharjee and C. Kommareddy. Scaleable application layer multicast. Proceedings of ACM SIGCOMM, August 2002.

[17] Gian Paolo Jesi. Build a topology generator for PeerSim 1.0. http://peersim. sourceforge.net/tutorial2/tutorial2.pdf.

[18] Russ Cox Frank Dabek, Frans Kaashoek, Robert Morris. Vivaldi: A Decentralized Network Coordinate System. ACM SIGCOMM 2004 Portland, Oregon, USA, 2004, 15-26.G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.