Management of Streaming Multimedia Content using Mobile Agent Technology on Pure P2P-based Distributed e-Learning System

Masayuki Higashino, Tadafumi Hayakawa, Kenichi Takahashi, Takao Kawamura, and Kazunori Sugahara
Department of Information and Electronics
Graduate School of Engineering, Tottori University
4-101, Koyama-Minami, Tottori 680-8552, Japan
Email:{s032047, s042047, takahashi, kawamura, sugahara}@ike.tottori-u.ac.jp

***Abstract*—Nowadays, a lot of e-Learning systems are widely deployed in educational schools. Typical e-Learning systems are implemented as client-server model. In the client-server model, the number of clients affects on the load of the server. In order to reduce the load on the server, we developed a P2P-based distributed e-Learning system. The proposed system consists of a lot of mobile agents which manage study contents and some functions such as scoring, showing questions, and correct answers. When a learner requests content, a mobile agent who has its content comes to the learner's computer, and then he/she can start the study. Here, a mobile agent has to manage multimedia data, which may be a huge size of data. Thus, a mobile agent has to migrates to the learner's node with a huge size of data. Then, the learner cannot start the study until the mobile agent finished to migrate. In order to solve this problem, we divide multimedia data into fragments and prepare mobile agents which manages each fragments. Since each mobile agents become small, a learner can start the study soon without waiting for the migration of a mobile agent which has a huge size of multimedia data. We, however, have to search a mobile agent which manages their fragments. Therefore, a mobile agent which manages n-th fragments informs its location to the agent which manages (n+1)-th fragments, and vice versa. Since each agent knows the location of a mobile agent which manages next and previous fragment; and fragments are cached into learner's node, a learner can play multimedia data smoothly without finding the location of mobile agent which manages holding next fragment. Experiment results show the effectiveness of our method.**

***Keywords*-e-learning system; mobile agent; multi-agent; distributed system; P2P; multimedia; streaming;**

I. Introduction

E-Learning systems, especially asynchronous Web-Based Training systems (WBT) [1], [2] are very popular. A WBT does not require interactions with instructors and allows a learner to study on his/her own time and schedule.

The mainstream e-Learning systems are based on the client/server model. The features of the client/server model are that all are to execute management and to offer the contents by the server machine. Although the client/server model has an advantage of easy construction and maintenance, however, the client/server model requires expensive initial investments in order to achieve fault-tolerance. Furthermore, in cases where the systems is discontinued due to economic or social reasons, the systems infrastructures and communities will be lost in a moment. Such systems are not open learning infrastructures in that the system administrators would restrict the system user's publications or subscriptions. It is desirable that learners be able to get learning opportunities freely in a low-cost way. Furthermore, in recent years, a paradigm shift called e-Learning 2.0 [3] has been occurring in the world of e-Learning. In e-Learning 2.0, learners not only can get passively learning opportunities, but also can publish or subscribe contents freely. Learners can enjoy the creative and voluntary learning from getting two-way learning relationship using self-made contents.

In order to realize such e-Learning systems, P2P (Peer to Peer) technologies are well suited. In a pure P2P architecture, every user's computer (hereafter we refer to such a computer as a node) plays the role of a client or a server, thus users are able to publish or subscribe contents freely. While a user uses the system, user's node is a part of the system. The systems can work with not only a stand-alone node but also a cluster of multiple nodes that communicate through a network. The nodes maintain an equal relationship each other. Therefore, the systems have no a center server, and the systems are able to have fault-tolerance. Moreover, the systems are able to coordinate and manage its performance and required investment, in accordance with the number of user's nodes. Of cause there is a method of increasing redundancy of a central server in order to increase system fault-tolerance. However, this method is only available if a system owner has sufficient funds to maintain the system. If there is a no owner, then the system will be lost.

Recently, multimedia distribution services (e.g., iTunes Store [4], Netflix [5], etc.) that provide music and videos via the Internet are becoming increasingly popular. In addition, in multimedia sharing services (e.g., Dailymotion [6], Youtube [7], Nico Nico Douga [8], Ustream.tv [9], etc.), the ordinary users can not only watch but also publish self-made multimedia contents. In the past, the Internet users usually communicate with each other by using texts or images mainly. However, nowadays they interact with each other by using multimedia generally and easily. It has been happening even in the field of education. e-Learning systems which support publications or subscription of multimedia contents have been popular because multimedia contents increase efficiency of learning. By using multimedia contents, e-Learning systems are able to provide more understandable learning contents compared with texts or images. In this paper, we propose basic functions of a pure P2P based e-Learning system, and focus on a method of multimedia contents management in particular.

In general centralized e-Learning systems, users cannot add answer formats (e.g., closed question, multiple-choice question, open-ended question, etc.). These answer formats are defined by system developers, and system users are allowed to use these formats by system administrators. In pure P2P based e-Learning systems, it is not possible to assume that the system administrators exist because all nodes belong to its end-users. For this reason, the systems require users to customize or update system programs to some extent. For example, if there is a user who wants to create a content which has new answer formats, then the system has to give this user a permission to program the new answer formats. Furthermore, in a pure P2P e-Learning system, given that all nodes belong its end-users, each and every node is not a high-performance machine such as a centralized server. Thus, the systems need (e.g., CPU time, memory usage, disk usage, network band wide, etc.) to balance load into each nodes. If not, teacher's nodes will go down because teacher's nodes that hold contents have a higher than student's nodes loads. For these reasons, a pure P2P e-Learning system is required high-flexibility and high-scalability.

II. Pure P2P-based Distributed e-Learning System

A. Overview

All of learning contents (exercises) in the proposed system are classified into categories, such as "Math/Statistic", "English/Grammar", etc. A user can obtain exercise one after another through the category. While a user joins the e-Learning system as a client, his/her node must be a part of the system as a server. When it joins the system, the node receives some number of categories and exercises from other node. Then, it has responsibility to send appropriate exercises request to requesting nodes.

The categories managed in each node are independent of categories in which the node's user is interested, as shown in Figure 1. This figure illustrates that the request from user A is forwarded first to neighbor nodes, then their neighbor nodes forward the request to next neighbor nodes, and finally, the request reaches to the node which manages a category requested.

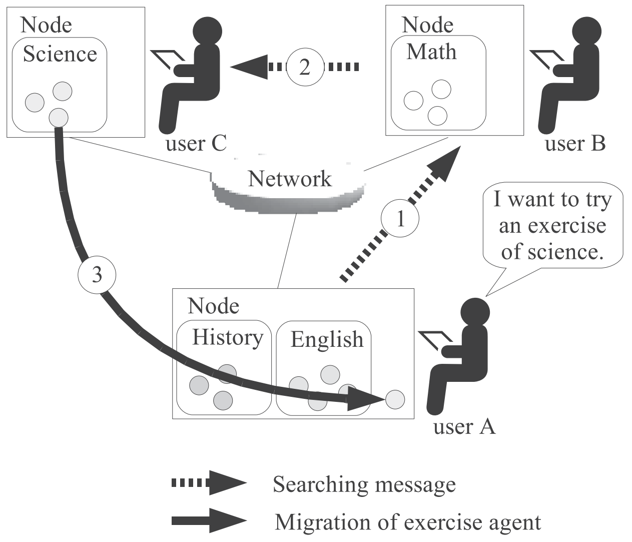


Figure 1. Migration of an exercise agent.

B. P2P Network

In existing P2P-based file sharing systems (e.g., Napster, Gnutella, Freenet, etc.) [10] each shared file are distributed among all nodes [11]. On the other hand, the categories in the proposed system are concentrated. When a new node joins the system, not only location information of a category but the category itself must be handed to the new node. Considering that, the P2P network of the proposed system can be constructed as a CAN [12].

Our P2P network is constructed with 2-dimensional coordinate space $[0, 1] \times [0, 1]$ to store exercise categories, as shown in Figure 2. The figure shows the situation that node **C** has just joined the system as the third node. Before node **C** joins, node **A** and node **B** shared the whole coordinate space half and half. At that moment, node **A** managed "Math/Geometry", "Math/Statistics", and "History/Rome" categories and node **B** managed "English/Grammar", "English/Reader" and "His tory/Japan" categories, respectively. When node **C** joins the system, node **C** is mapped on a certain coordinate space according to a random number and takes the responsibility of management of some categories. For example, node **C** takes on the "History/Japan" category from node **B**, then all learning contents in the category moves to node **C**. After node **C** joins the system, node **C** gets a list of IP addresses of neighbor nodes in the coordinate space. This enables neighbor nodes to communicate with each other.

C. Components

We have to consider not only the distribution of learning contents but also the distribution of functions to provide above services. Our system consists of mobile agents [13],

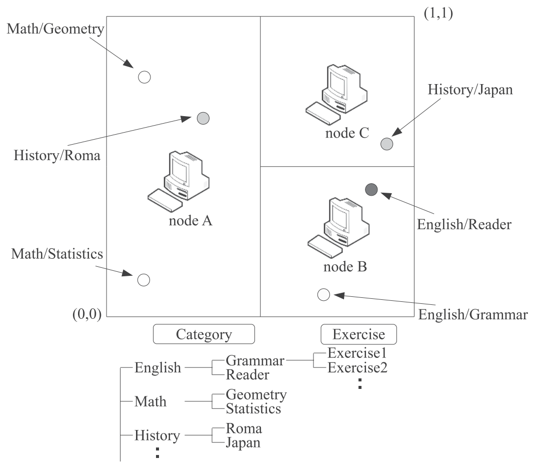


Figure 2. P2P network of proposed e-Learning system.

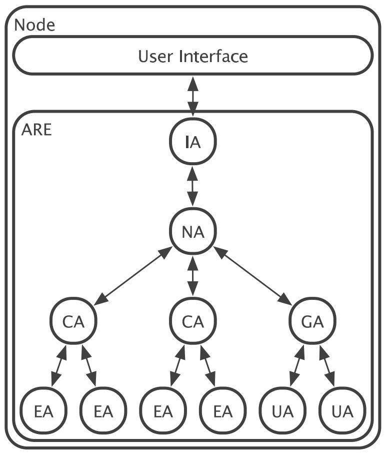


Figure 3. Interaction between components of our pure P2P-based Distributed e-Learning system. Each mobile agent runs on ARE (Agent Runtime Environment).

[14] and user interface program. In our system, functions are provided as the part of mobile agent. A mobile agent migrates from one node to another node with a function it provides. These mobile agents are implemented in the mobile agent framework called Maglog [15], [16]. Figure 3 shows a interaction between agents.

- **Node Agent (NA)**: Each node has one node agent. It manages the zone information of a CAN and forwards messages to the Category Agents in the node.
- **Exercise Agent (EA)**: Each Exercise Agent has questions and functions to score user's answers, to tell the correct answers, and to show some related information about the exercise.
- **Category Agent (CA)**: One category agent is installed in one learning category. A category agent manages exercise agents related on its category and dispatches the learning request to them.
- **User Agent (UA)**: Each user has its own User Agent. A User Agent manages user's learning information that includes login name, password, IP address of the user's computer, online/offline status, and log of learning and/or a list of created exercises.
- **Group Agent (GA)**: Each group agent manages user agents.
- **User Interface**: One user interface is on each node which a user logs in as a student. It provides a user interface program for learning.
- **Interface Agent (IA)**: There is one interface agent for each user interface, such as a student interface and an exercise manager interface on each node. It intermediates between the interface program and agents, and between agents and applications.

III. Management of Streaming Multimedia Data

A. Content Distribution

In the Pure P2P-based distributed e-Learning system, when considering a location where content is stored, as one example, there is a way that a teacher stores accountably contents into own node. However, in this way, teacher's node will be SPOF (single point of failure). For example, if a teacher provides highly popular contents then the concentrated load occurs at the teacher's node. Additionally, when a failure occurs at the teacher's nodes, or the teacher's nodes go offline, students cannot access the contents. Therefore, the learning contents must be distributed in a network of the e-Learning system.

In ordinary P2P-based file sharing systems each shared file are distributed among all nodes. Their methods are able to realize a fault-tolerance by file units level. However, given that many of learning contents are text data and its size is small, a learning content lookup cost is unable to disregard at the time of learning. Thus, the learning contents must be managed as a group (e.g., category).

B. Multimedia Content Distribution

If a learning content is a multimedia such as a video, a size of a multimedia will be reached MiB or GiB. There really is difference of data size between a text content and a video content. Thus, simply distributing a learning content as a unit of a file or a group, the inequality of resource usage (e.g., CPU time, memory usage, disk usage, network band wide, etc.) between nodes occurs. Additionally, in the case of a transfer of a text content, a transfer from a content requested node to a content request node is finish at short time. However, in the case of a video content, it takes a long time. This means that the concentrated load, a concurrent video streaming processing, occurs at a node holding a video contents.

Therefore, if a learning content contains a multimedia data, then the multimedia data is divided to small fragments, and this learning content has a description of a multimedia and references of locations of these fragments. Thus, the possibility that a long time and multiple load occurs at a node which requested a multimedia content is low, because this node transfers only of an description and references.

C. Distributed Multimedia Content Streaming

Each fragment is mapped on DHT (Distributed Hash Table) based on its keys. Since a multimedia content is composed of some fragments, we have to know the location of these fragments on DHT. Therefore, each fragment has the key of a next fragment. Then, each fragment can find next fragment according to its key. However, it needs a lot of message to find a node which manages next fragment. Because our system uses 2-dimensional CAN, $O(\sqrt{n})$ messages are required to find next fragment. Here, n is the total number of nodes. This may impede smooth playing of the multimedia data. To solve this issue, before the multimedia content is required from learners, each fragment finds a node which manages next fragment and records its location. Thus, every fragment is linked in the order of time series of the multimedia data. Consequently, we can reduce messages to find the location of fragments except first fragment's search.

When a node joins or leaves (churn), fragments are transferred to other nodes. If a fragment is transferred, the link between fragments becomes useless anymore. To keep the link, when a node joins or leaves, each fragment records not only the location of a next fragment but also a previous fragment. When a fragment is transferred, the fragment notifies its new location both to a next and previous fragment. Thus, the link between fragments is kept continuously even when a node joins or leaves.

D. Distributed Multimedia Content Caching

When a user learns using a multimedia content, fragments of the multimedia content temporarily gather in the learner's node. Therefore, this node can be used as a cache node. When a user learns using a multimedia content, the user's node stores the caches of the fragments into own node; and a node that has an original fragment, stores references of cache stored nodes. In this case, the node with original fragments collects references of caches of fragments. Thus, this node can balance a self-load like a DNS round robin (Figure 4).

IV. Experiments

A. Smooth Play of Multimedia Data

In order to show the effectiveness of our proposed method, we have evaluated whether a multimedia data is played smoothly. Measuring a difference time length between a time length of multimedia data and an actual playing time length, we are able to evaluate whether a multimedia data is played smoothly. This difference time length is a total pausing time length when the multimedia data is played, and close to zero means that the multimedia content is played smoothly. Thus, we have measured a total pausing time length in the case of *with link* and *without link*.

This experiment has done with our proposed e-Learning system that runs on computers (Intel Core i5 processor 3.2 GHz and 4 GB RAM with JRE (Java Runtime Environment) 1.6 and Debian GNU/Linux 5.0.5) connected through Ethernet of 100 BASE-T. In this experiment, we have defined an about 16 MiB sized video file which has 21 seconds as multimedia data, and the video file have divided into fragments of 200, 225, 250, 275, or 300 KiB by time series.

Figure 5 shows the result of this experiment. In *without link*, a total pausing time length increases with decreases in division size. For example, when division size is 200 KiB, the total time of pausing is 25 seconds. This is because each MA has to lookup next MA on DHT. In contrary, in *with link*, a total pause time length does not depend on division size. Thus, the multimedia data can be played smoothly. This result shows that as the division size smaller, in other words, as more the multimedia data is divided and distributed in a DHT network, our proposed method is more effective.

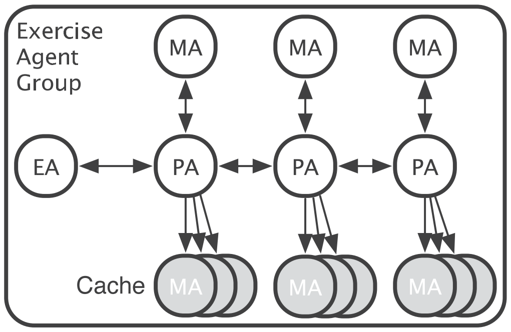


Figure 4. Distributed multimedia content caching by mobile agents. A PA is a pointer agent which manages MAs and caches of MAs. A MA is a multimedia agent which holds a fragment of a multimedia data. Each agent is mapped on a DHT network.

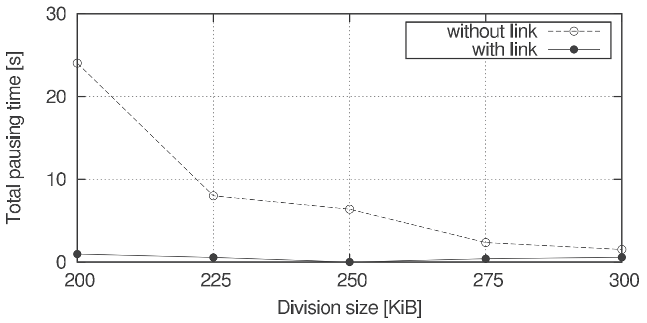


Figure 5. Total pausing time according to division size.

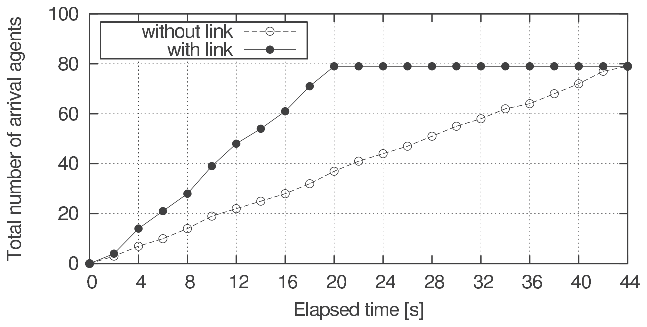


Figure 6. Number of arrival agents according to elapsed time.

B. Total Number of Arrival Agents in Each Elapsed Time

In order to examine more details of effectiveness in our proposed method on a system level not a user level, we have measured a total number of arrival agents in each elapsed time.

This experiment has done with the same configurations of the e-Learning system, computers, and video file in Section IV-A; but a division size has set only to 200 KiB.

Figure 6 shows this experimental result. In *without link*, many agents are migrating after exceeding the 21 seconds; therefore, the file data cannot be played smoothly. In contrary, in *with link*, many agents has arrived until 21 seconds, however, note that the load on the node will be concentrated. The influences about this situation are examined after this Section.

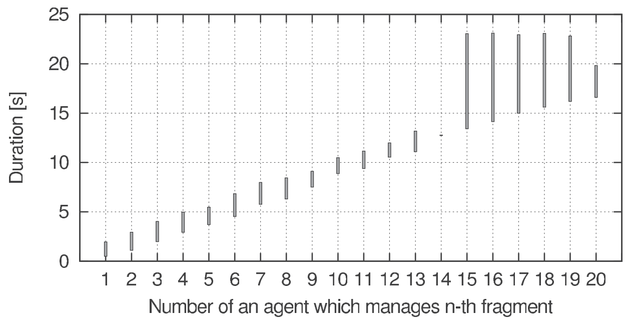
C. Limitation of Simultaneous Agent Migrations

If division size is large, the multimedia file may not be smoothly played because some agents migrate to a requesting node simultaneously. The simultaneous agent migrations will cause extreme resource consumption. Thus, playing a multimedia file, the node has to limit to the number of agent migrations simultaneously.

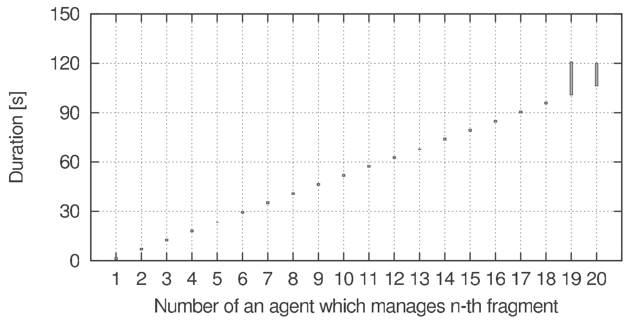
We have investigated in four cases. In case 1, an agent sends a requesting message to next agent immediately. In case 2, an agent sends a requesting message to next agent after it waits for 5 seconds from a requesting message from previous agent. In case 3, an agent sends a requesting message to next agent after it waits for 10 seconds. In case 4, an agent sends a requesting message to next agent after the migration to a requesting node finishes.

In this experiment, we have investigated duration of agent migrations to a requesting node. Each agent manages a fragment of 10 MiB that is a part of a video file of 192 MiB, and the video file has 372 seconds. All other experiment configurations are the same as Section IV-A.

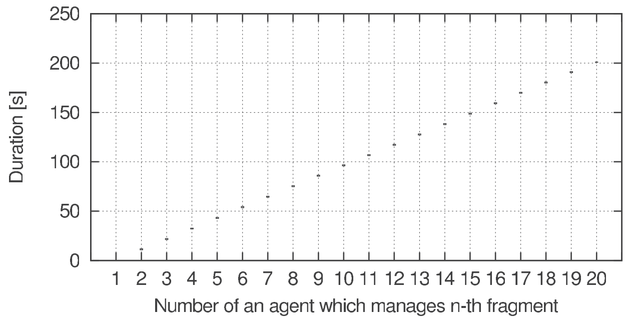
Figure 7 shows the results of these experiments. We are able to see from Figure 7(a) that the case 1 spends mush time for agent migrations. For example, a 1st agent starts migration at time 0 and finishes at time 2, the 15th agent



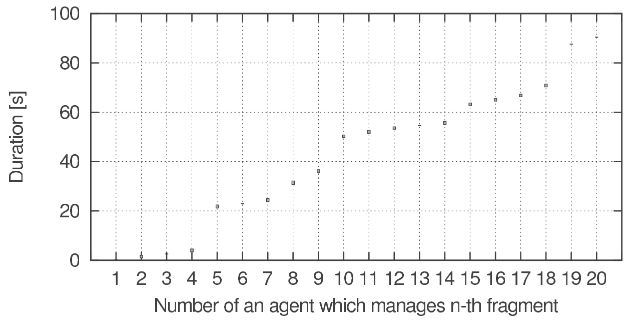
(a) Case 1: Serial requests each has **no** delay; and parallel agent migrations occur.



(b) Case 2: Serial requests each has **5** seconds of delay; and parallel agent migrations occur.



(c) Case 3: Serial requests each has **10** seconds of delay; and parallel agent migrations occur.



(d) Case 4: Serial requests and agent migrations with **no** delay.

Figure 7. Duration to finishing the agent migrations from starting to play a multimedia file.

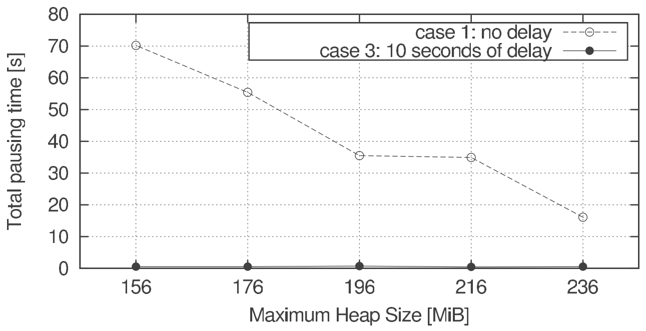


Figure 8. Effect of maximum heap size on total pausing time.

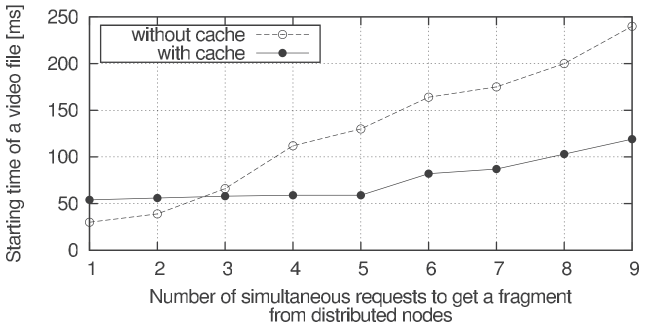


Figure 9. Comparison of response time between *with cache* and *without cache*.

starts migration at time 14 and finishes at time 24. However, in other cases appear in Figure 7(b) and Figure 7(c), almost agents are able to migrate to a requesting node in shorter time. Thus, The agent migration time decreases with increasing delay time. However, the duration of agent migrations have to less than the playing time of the multimedia file. For example, in the case 1 appears in Figure 7(d), agent migration time is shorter than other cases, but the video file played not smoothly.

D. Influence of Video Smoothness on Memory Size

We have investigated the total pausing time in the case 1 (no delay) and the case 3 (10 seconds of delay) in each maximum memory size.

In this experiment, we have adopted the maximum heap size of JVM (Java Virtual Machine) [17] to the maximum memory size of a node, and the heap size have set a value which from 156 to 236 with step 20 MiB. All other experiment configurations are the same as Section IV-A.

Figure 8 shows the result of this experiment. In the case 1 (no delay), since many agents have to migrate simultaneously, the heap usage becomes full soon. Therefore, many agents are swapped out, and it obstructs the smooth play of multimedia data. In contrary, in the case 4 (10 seconds of delay), this does not happen. Thus, it is necessary to control the migration of agents when a many agents try to migrate simultaneously.

E. Effectiveness of Distributed Multimedia data Cache

We have investigated the effectiveness of distributed multimedia data cache.

This experiment has done with the same configurations of the e-Learning system, computers in Section IV-A; but the video file size is 5.6 MiB and its file is divided into three fragments.

Figure 9 shows the result of this experiments. When the number of simultaneous requests to get a fragment from distributed nodes is 1 or 2, the response time of *with cache* is longer than *without cache*. Because, in spite of the requested node has no large load, the requested node routes these requests to other nodes which have cache of fragment of video file. However, when the number is greater than or equal to 3, the response improving. Thus, greater the number of requests, the cache is more effective.

V. Related Work

Several researches have proposed P2P-based e-Learning system. EDUTELLA [18] is a one of the earliest a P2P-based e-Learning system. This is based on JXTA [19]. JXTA is a P2P application framework. There are JXTA-based e-Learning systems such as PLANT [20] and HYDRA [21]. In addition, several researches have proposed a new overlay network for a P2P-based e-Learning system such as PROSA [22] and PeerLearning [23]. These systems are different from our proposed system in that multimedia contents are divided into fragments and managed by DHT.

Tagashira et al. [24] has proposed an index caching mechanism for CAN. In the other hand, our proposed system caches not only indexes but also data of the learning contents.

Huang et al. [25] has proposed an agent-based collaborative virtual environment architecture using grid technologies, however, this is not based on a pure P2P model. On the other hand, our proposed system is based on a pure P2P model.

In Nearcast [26], a locality-aware P2P live streaming method has proposed. However, in a live streaming, a source node will be a single point of failure. If the source node forced outage, the streaming will stop. On the other hand, in our proposed system, a multimedia content is preliminarily divided and is preliminarily distributed into many nodes before all node is notified of the multimedia content existence. Therefore, the multimedia content streaming will not be forced outage by economic or social reasons.

This characteristic is similar to P2P-based file sharing systems; but these systems do not expect to the multimedia streaming and cannot stream multimedia because the fragments are transferred in random order.

VI. Conclusion

In this paper, a method to play multimedia data smoothly on a pure P2P-based distributed e-Learning system is proposed and its implementation is described. In our system, multimedia data is divided into multiple fragments by time series, each media agent manages each fragment, and their media agents are linked bidirectional. If division size of multimedia data is huge, multimedia data cannot be played smoothly because a lot of agents have to migrate to a requesting node simultaneously. Therefore, we devised the timing of sending a requesting message to next agent. This method enabled smooth play of multimedia data.

REFERENCES

[1] M. Driscoll, *Web-Based Training: Creating e-Learning Experiences*, 2nd ed. New York, USA: John Wiley & Sons, 2002.

[2] V. Uskov, Ed., *Web-based Education*. ACTA Press, 2010.

[3] S. Downes, "E-learning 2.0," in *eLearn Magazine*. ACM, 2005.

[4] Apple Inc. (2012, Aug.) Apple - itunes - everything you need to be entertained. [Online]. Available: http://www.apple.com/itunes/

[5] Netflix, Inc. (2012, Aug.) Netflix - watch tv shows online, watch movies online. [Online]. Available: http://www.netflix.com/

[6] Dailymotion. (2012, Aug.) Dailymotion - Watch, publish, share videos. [Online]. Available: http://www.dailymotion.com/

[7] YouTube, LLC. (2012, Aug.) Youtube - broadcast yourself. [Online]. Available: http://www.youtube.com/

[8] Niwango, inc. (2012, Aug.) Nico nico douga. [Online]. Available: http://www.nicovideo.jp/

[9] Ustream, Inc. (2012, Aug.) Ustream - you're on - broadcast live streaming video, watch online events, chat live, send a tweet, follow on facebook, myspace, record your live shows. [Online]. Available: http://www.ustream.tv/

[10] S. Androutsellis-Theotokis, "White paper: A survey of peer-to-peer file sharing technologies," ELTRUN, Athens University of Economics and Business, Tech. Rep., 2002.

[11] T. Chothia and K. Chatzikokolakis, "A survey of anonymous peer-to-peer file-sharing," in *Proceedings of the 2005 international conference on Embedded and Ubiquitous Computing*, 2005, pp. 744–755.

[12] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A scalable content-addressable network," in *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications*, 2001, pp. 161–172.

[13] A. R. Hurson, E. Jean, M. Ongtang, X. Gao, Y. Jiao, and T. E. Potok, "Recent advances in mobile agent-oriented applications," in *Mobile Intelligence: Mobile Computing and Computational Intelligence*, 2010, pp. 106–139.

[14] A. Outtagarts, "Mobile agent-based applications : a survey," *International Journal of Computer Science and Network Security*, vol. 9, no. 11, pp. 331–339, 2009.

[15] T. Kawamura, S. Motomura, and K. Sugahara, "Implementation of a logic-based multi agent framework on java environment," in *Proceedings of IEEE International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, 2005, pp. 486–491.

[16] S. Motomura, T. Kawamura, and K. Sugahara, "Logic-based mobile agent framework with a concept of "field"," *IPSJ Journal*, vol. 47, no. 4, pp. 1230–1238, 2006.

[17] Oracle and/or its affiliates. (2012, Aug.) java - the java application launcher. [Online]. Available: http://docs.oracle.com/javase/6/docs/technotes/tools/solaris/java.html

[18] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmér, and T. Risch, "Edutella: a p2p networking infrastructure based on rdf," in *Proceedings of the 11th International Conference on World Wide Web*, 2002, pp. 604–615.

[19] J. Verstrynge, *Practical JXTA II*. Lulu Enterprises Inc., 2010.

[20] M. Li, H. Zhu, and Y. Zhu, "Plant: a distributed architecture for personalized e-learning," in *Proceedings of the 6th International Conference on Advances in Web Based Learning*, 2007, pp. 20–30.

[21] I. A. Zualkernan, "Hydra: A light-weight, scorm-based p2p e-learning architecture," in *Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies*, 2005, pp. 484–486.

[22] V. Carchiolo, A. Longheu, G. Mangioni, and V. Nicosia, "Adaptive e—learning: An architecture based on prosa p2p network," in *Proceedings of the 21st International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems: New Frontiers in Applied Artificial Intelligence*, 2008, pp. 777–786.

[23] G. Wang, Y. Yuan, Y. Sun, J. Xin, and Y. Zhang, "Peerlearning: A content-based e-learning material sharing system based on p2p network," *World Wide Web*, vol. 13, no. 3, pp. 275–305, 2010.

[24] S. Tagashira, S. Shirakawa, and S. Fujita, "Proxy-based index caching for content-addressable networks," *IEICE Transactions on Information and Systems*, vol. E89-D, no. 2, pp. 555–562, 2006.

[25] C. Huang, F. Xu, X. Xu, and X. Zheng, "Towards an agent-based robust collaborative virtual environment for e-learning in the service grid," in *Proceedings of the 9th Pacific Rim International Conference on Agent Computing and Multi-Agent Systems*, 2006, pp. 702–707.

[26] X. Tu, H. Jin, X. Liao, and J. Cao, "Nearcast: A locality-aware p2p live streaming approach for distance education," *ACM Transactions on Internet Technology (TOIT)*, vol. 8, no. 2, Article 7, pp. 7:1–7:23, 2008.