File Replication Method Based on Demand Forecasting of File Download in P2P Networks

Mamoru Kobayashi[1], Jun Kageyama[2], Susumu Shibusawa[3], Tatsuhiro Yonekura[4]
[1]Graduate School of Science and Engineering
Ibaraki University, Japan

[2]F-COM Co., Ltd
Japan

[3, 4]Department of Computer and Information Sciences
Ibaraki University
Japan
[1]08nd308g@hcs.ibaraki.ac.jp, [2]k07nm703s@yohoo.co.jp, [3]sibusawa@mx.ibaraki.ac.jp,
[4]yone@mx.ibaraki.ac.jp

Journal of Digital Information Management

ABSTRACT: *In peer-to-peer (P2P) networks that support file-sharing services, the level of access demand can vary widely between different files. Since fewer nodes store files for which there is a lower demand, these files are more likely to be lost from the P2P networks if users leave the network or delete the files. File loss can cause users to seek alternatives to P2P services, and lead to degradation in service quality. We propose and evaluate a replication method based on demand forecasting that aims to prevent the loss of low-demand files. In this method, the number of file replicas to be placed is determined based on the forecast demand for the file, so that the loss of low-demand files is likely prevented by placing replicas at nodes that frequently use P2P services. Based on simulation results, we compared our proposed method with basic replication methods in terms of the number of files and the amount of storage used. Our experimental results show that the proposed method prevents file loss by preserving low-demand files over extended periods of time. We also confirmed that the node storage resources consumed by this method are efficiently used.*

Categories and Subject Descriptors
D.4.3 [File Systems Management]; C.2 [Computer-Communication Networks]: Data Communications

General Terms: P2P Networks; File Transfer, P2P Services

Keywords: P2P File sharing, P2P Networks, Replication

Received: 11 December 2009; Revised 19 January 2010; Accepted 21 Janurary 2010

1. Introduction

P2P services have received much attention as Internet file-sharing applications. In a P2P service, the demand for files is not uniform, and files that are in greater demand are exchanged between more nodes and spread more widely across the P2P network. Meanwhile, files that are less in demand are held at fewer nodes, and are thus more likely to be lost from the P2P network if users leave the network or delete the files. The file loss leads to dissatisfaction among users, causing them to abandon P2P services, resulting in a poorer quality of service.

Many replication methods have been proposed for spreading files to other nodes to maintain the quality of P2P networks [1], [2], [3], and most replication methods are geared towards the efficient delivery of high-demand files. Since most nodes in P2P networks join and leave a network many times, several studies on this process and the lifetime of nodes have been presented to maintain the quality of P2P networks [4], [5], [6], [7], [8]. From the measurement results of the P2P networks, the lifetime of a node shows a long-tailed subexponential distribution rather than an exponential distribution. However, few studies on efficient methods to prevent file loss have been presented.

To prevent file loss by replication, it is first necessary to ensure that a certain number of files are kept on the network, and this number should not reach zero even if the nodes where the files are stored either leave the network or the files are deleted by users. We propose a replication method that uses demand forecasting to prevent the loss of low-demand files in P2P networks. This method involves estimating the demand for files and determining the number of replicas to be placed on the network based on this estimated value. The loss of low-demand files is prevented by placing replicas at nodes that are used frequently and continuously by the P2P service.

To evaluate the proposed method, we implemented it in a simulation system and performed simulation experiments in which it was compared with other basic replication methods. For the internal network of this simulation, we used a super-node P2P model. As a result of long-term experiments, we found that files were preserved when using the proposed method, but were lost when using other replication methods. We also confirmed that the proposed method is effective in reducing the consumption of node storage resources.

In Section 2 we show related work on replication methods, and in Section 3 we introduce a demand model of P2P services. In Section 4 we describe our proposed method. Section 5 presents the construction of our simulation system, and Section 6 discusses the experiment results. Section 7 concludes this paper with a summary.

2. Related work

P2P networks have three characteristics: self-organization, symmetric communication and distributed control. A self-organizing P2P network autonomously adapts to the arrival, departure, and failure of nodes. Communication is symmetric in that nodes act as both clients and servers. Many replication

methods have been proposed for spreading files to other nodes to improve the quality of P2P networks [1], [2], [3], and most replication methods aim at the efficient delivery of high-demand files [9], [10].

Liv et al. [1] propose a query algorithm based on multiple random walks that reduce P2P network traffic and show that uniform random graphs yield the best performance among various network topologies. Cohen et al. [2] show that two strategies, uniform and proportional for replication in unstructured P2P networks yield the same average performance on queries, and that the optimal strategy lies between the two strategies. Liben-Nowell et al. [3] give a general lower bound on the rate that the Chord network remains connected as nodes join and leave the network.

Basic replication methods include Owner Replication, Path Replication, and Random Replication in terms of the number and place of replicated files [1]. Owner Replication is a method used by Gnutella [15], where replica files are only placed at the requester's node when a search hit occurs. Since the number of replica files placed in a single search can be no more than 1, this is a simple technique that minimizes the costs associated with network loads, but it takes a long time for replicas to spread adequately across the network. Figures 1 (a) and (b) show examples of a search request and the placement of replicas using Owner Replication, respectively.

Path Replication is a method used in Freenet [16], where replicas are placed at all nodes along the search path from the search requester to the file holder. Since this method places multiple replicas at the same time, it allows content to spread more easily, but increases the network resources needed for file storage. An example of this replication is shown in Figure 1 (c). Random Replication is a method where replicas that are equal in number to the nodes on the search path are placed at randomly selected nodes on the P2P network.

Most nodes in a P2P network join and leave the network many times. The lifetime of nodes and the process of arrival and departure are important factors for the design and evaluation of networks, and many studies are in progress. Stutzbach et al. [4] examine node participation in three familiar P2P networks,

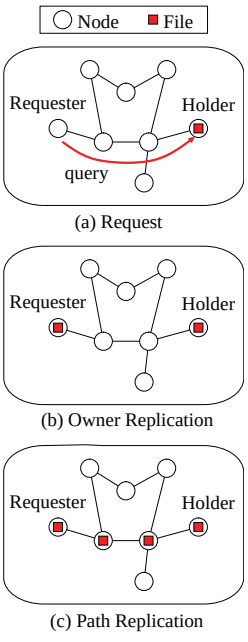

Figure 1. Search request and examples of replication methods

and the results reveal that session lengths are fit by Weibull distribution rather than exponential distribution. Leonard et al. [5] examine the resilience of node failures in random graphs for deriving the relationship between the node arrival/departure and connection of networks, and show that networks with heavy-tailed lifetime distributions are more resilient than those with exponential distribution. Stutzbach et al. [6] state that the Gnutella network has a topology that is highly resilient to the random departure of nodes and even systematic attacks, and describe that most long-lived nodes form a well-connected core in the network.

Distributed hash tables (DHTs) provide scalable and key-based lookup of objects in dynamic networks. Falkner et al. [7] measure the usage of the Azureus BitTorrent DHT and provide a modified DHT lookup algorithm, which reduces DHT lookup time by an order of magnitude. Rhea et al. [8] propose a method that can addresses the continuous process of node arrival and departure in DHTs, and examine the table lookup procedure for node arrival and departure on their emulator.

On the other hand, much research has been done on customer demand in the economic theory of consumer behavior. Blattberg et al. [11] propose a method for maximizing customer equity using the customer acquisition and retention rates of exponential curves. Abe [12] derives the stochastic consumer departure rate by incorporating a consumer behavior model into Recency/Frequency (RF) analysis, and performs a simulation experiment. Axsäter [13] explains an exponential smoothing method and demand models to forecast consumer demand.

In P2P services, the upload/download of files in a P2P network can be considered as a user's demand for files. The demand for files is not uniform, and files that are in greater demand are exchanged between more nodes and spread more widely across the P2P network. Meanwhile, files that are less in demand are held at fewer nodes, and are thus more likely to be lost from the P2P network if users leave the network or delete the files. There has not been much progress in developing efficient methods for preventing file loss.

3. Demand model of P2P services

In the economic theory of consumer behavior, customer's purchasing is modeled for the purchasing frequency and heterogeneity [11], [12]. The process from the time that customers start purchasing items to the time that they quit for various reasons, such as changing shops or their movement, can be modeled using stochastic processes. In the consumer behavior theory, it is often assumed that purchasing occurs randomly and purchasing period fits the exponential distribution. Moreover, consumer purchasing distribution and period are assumed to be different in each other.

On the other hand, it has been reported that the lifetime of nodes on P2P networks shows a long-tailed subexponential distribution rather than an exponential distribution. To facilitate our simulation of file replication using demand forecasting, we assume distributions of the user's participation/departure and the download/upload of files as follows:

Assumption 1. User participation and departure from services occur at random.

Assumption 2. File download and upload follow the compound Poisson process.

Assumption 2 assumes that file download and upload occur randomly regardless of past occurrences. It also includes the possibility that more than one file is downloaded or uploaded,

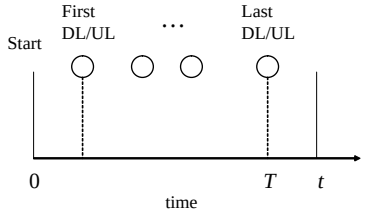

Figure 2. Relationship between t and T during the measurement period.

and that the number of files follows a certain distribution. Figure 2 shows an example where four download/upload operations are performed in a certain measurement interval t. Here, T is the interval from the beginning of the measurement to the last of these download/upload operations.

We especially pay attention to file downloading, and call a download request demand. When a download request follows the compound Poisson process with the occurrence rate λ, the probability $P^k(t)$ that k downloads occur in time interval t is expressed as follows:

$$P^k(t) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}, \ k = 0,1,2,... \quad (1)$$

When the download request follows a compound Poisson process, the number of files per download is also a stochastic variable. The number of files for a download does not depend on that for other downloads and their distribution [14].

Let f_j be the probability that the number of files per download is j, where j =1, 2,..., and let f_j^k be the probability that the number of files per k downloads is j, where $k \le j$ and k = 1,2,... When the number of files downloaded in time interval t is $D(t)$, the probability that the total number of files is j in time interval t is expressed as follows:

$$P(D(t) = j) \equiv P_j(t) = \sum_{k=0}^{\infty} P^k(t) f_j^k = \sum_{k=0}^{\infty} \frac{(\lambda t)^k}{k!} e^{-\lambda t} f_j^k \quad (2)$$

In the compound Poisson process, the Poisson process and the event that occurs along with the Poisson process are mutually independent, and a simple relation exists between these two events [13], [14]. When Assumption 2 holds for the download of files, a similar relation as that of the general compound Poisson process is also satisfied. Let stochastic variable K be the number of downloads during a unit of time, and stochastic variable J be the number of files per download. Also, let stochastic variable Z be the number of files by all downloads during a unit of time. Because K and J are independent in the compound Poisson process, Z is simply the sum of the number of files for K independent downloads. For expected value μ and variance σ of Z, the following property holds.

Lemma. When a file download follows the compound Poisson process of the occurrence rate λ, using the number of files per download, J, the expected value μ and variance σ for the total files Z downloaded during a unit of time are expressed as follows:

$$\mu = E(Z) = \lambda E(J) = \lambda \sum_{j=1}^{\infty} j f_j \quad (3)$$

$$\sigma^2 = Var(Z) = \lambda E(J^2) = \lambda \sum_{j=1}^{\infty} j^2 f_j \quad (4)$$

where notations E and Var are the expected value and variance, respectively, and f_j is the probability that the number of files per download is j.

From the above Lemma, the expected value μ' and variance σ'^2 of the number of files in time interval t are expressed as follows:

$$\mu' = \mu t, \ \sigma'^2 = \sigma^2 t$$

When the number of downloaded files is always one, the download demand becomes a pure Poisson process because $f_1 = 1$, $f_j = 0$ ($J > 2$) and $\mu = \sigma^2 = 1$. In the compound Poisson process, since the number of files per download is $J \ge 1$ during a unit of time, the ratio of the expected value μ to variance σ^2 of the total number of files is as follows:

$$\frac{\sigma^2}{\mu} = \frac{E(J^2)}{E(J)} = 1 \quad (5)$$

Thus, it is not possible to model a file download with $\sigma^2/\mu < 1$ as the compound Poisson process.

When the demand is comparatively low in stochastic consumer behavior models, the compound Poisson process is usually used [13]. It is more convenient to use continuous distributions for items with higher demand, and the normal distribution is usually used. We assume the compound Poisson process for low-demand files in P2P networks.

4. Proposed method

To prevent the loss of low-demand files, replication is performed in the following two steps.

- Replication based on forecast demand
- Selection of nodes where replicas are to be placed

First, demand forecasting is performed for each file, and low-demand files are identified. Next, a replication process is performed for these low-demand files. During the replication process, the destination nodes where replicas are placed are selected according to the criteria outlined below.

4.1 Replication based on forecast demand

4.1.1 Replication scheme

Two different replication schemes are used depending on the file demand and whether or not a download has been requested.

- When there is a download request

Replication is performed in the same way as with Owner Replication. Owner Replication is a replication method where a single replica is produced for a single download request. It is therefore suitable for replication of low-demand files whose replicas do not need to be generated in large quantities.

- When there is no download request

The forecast demand for each file is calculated to identify low-demand files, and replication processing is performed for files numbering less than a fixed quantity.

In Owner Replication, since replication processing is only performed when a download request has been issued, it is impossible to address the risk of losing low-demand files as a result of deletion by users or the disconnection of nodes from the network. Consequently, the demand for files is periodically assessed, and pre-emptive replication is performed for low-demand files.

4.1.2 Demand forecasting method

Low-demand files are identified for performing a replication process. We assume hereafter that the download demand

for each file follows a pure Poisson process and express the number of downloads of the file in time interval t as $d(t)$. We estimate the total number Δ of downloads of a file during sufficient amount of time. When the ratio of $d(t)$ to Δ falls below the value $\delta(0) < \delta < 1$, the file is considered as a low-demand file. Let t_y be the time that the ratio of $d(t)$ to Δ is equal to d, and we express the relation as follows:.

$$\frac{1}{\Delta}\int_{y}^{\infty} d(t)\,dt = \delta \tag{6}$$

We use $\delta = 0.05$ in this paper.

Next, the demand for a file is measured at every time interval, and the forecast demand is calculated by exponential smoothing [13]. Prediction is performed using the results of previous demand measurements and predictions. The measured and predicted values of the demand at time t are expressed as Md_t and Fd_t, respectively. The forecast demand Fd_{t+1} at time t+1 can be expressed as follows:

$$Fd_{t+1} = \alpha \times Md_t + (1-\alpha) \times Fd_t \tag{7}$$

In the above formula, a is a smoothing constant that is set to 0.5 in this paper. Low-demand file replication is performed in cases where $Fd_{t+1} < d(t_y)$.

4.1.3 Replication of low-demand files

When performing replication of low-demand files, it is necessary to decide how many replicas to be placed. This involves deciding on a safe number of files needed to prevent losses, which is determined from the number of current files and the ratio of the predicted and measured demand. File loss is prevented by placing enough replicas so that the total number exceeds this safe number. If N_t and S_t are the number of files and the safe number of files at time t, respectively, then the safe number of files can be expressed as follows:

$$S_t = \beta \times N_t \times \frac{Fd_{t+1}}{Md_t} \tag{8}$$

where β is a safety factor.

4.2 Selecting where to place the replicas

To make low-demand files easy to find for searches in P2P networks, the files are put on nodes to contribute to file exchange. The contribution level of nodes is determined as follows based on the number of files uploaded *NUL* and downloaded *NDL* and the average uptime.

$$Contribution = \frac{NUL}{NUL + NDL} \times AverageUptime \tag{9}$$

Nodes with a large contribution use the P2P network continuously and frequently. Placing low-demand files at these nodes, the files survive for longer, have more opportunity for being discovered when users search for them, and are thereby protected against being lost.

5. Simulation system

To evaluate the proposed method, we used an overlay construction toolkit called Overlay Weaver [17], [18] to construct a simulation system. This system consists of the following three parts:

- Simulator
- Scenario file generator
- Scenario file

First, starting parameters are input into scenario file generator, which generates a scenario file based on these parameters.

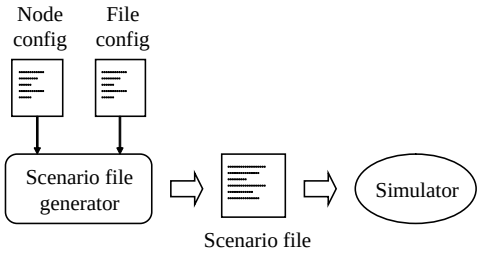

Figure 3. Procedure up to the start of simulation.

This scenario file is read into the simulator to start a simulation. The procedure up to the point where the simulation starts is shown in Figure 3.

For the internal network structure of the simulator, we used a super-node type of P2P model. This network model is characterized by its use of an efficient indexed search system and its superior performance at the collection of data inside the network [19], [20]. These characteristics make the system an efficient tool for searching for low-demand files and for collecting data.

The simulation system consists of super-nodes and ordinary nodes. The network structure is shown in Figure 4. The super-nodes form individual groups consisting of themselves and their subordinate ordinary nodes, where they collect information about the files owned by the nodes in the group and information needed when calculating demand forecasts and deciding where to place replicas. The search system is an indexing system that uses file lists stored in the super-nodes as representatives of each group. The super-nodes are networked together, allowing them to efficiently discover rare low-demand files by consulting each other's file lists. It is also possible to perform efficient replication processing by communicating between super-nodes when forecasting demand and selecting destinations for replica placement. The super-nodes are selected from among the ordinary nodes, and super-nodes themselves also allow files to be uploaded and downloaded.

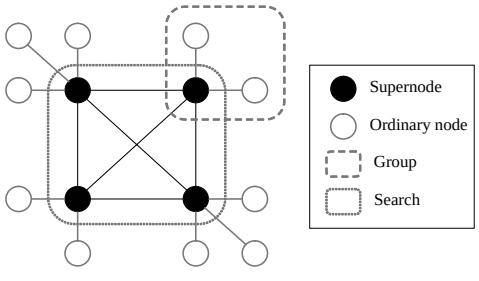

Figure 4. Configuration of super-node type P2P model

6. Simulation experiments

We performed simulation experiments to test and evaluate the proposed method. We checked the operation of the replication process and performed a comparative evaluation of the proposed method with basic replication methods.

6.1 Experimental setup

The interval between the acquisition of information and updating of this information in the simulation system is called a period. The simulation environment used in experiments is shown below.

- Number of nodes: 2,000 (20 super-nodes, 1,980 ordinary nodes)
- Each super-node has roughly the same number of ordinary nodes assigned to it

- Number of file types: 50
- Measurement interval: Once per period
- Duration of scenario: 500 periods

In addition to the proposed replication method, we also used Owner Replication and Path Replication methods. For these three methods, we compared changes in the number of files and storage capacity.

6.2 Confirmation of changes in the number of files

We checked the operation of replication processing in the proposed method. Figure 5 shows changes of 10 types of file selected from the 50 types used in the experiment. The vertical axis shows the number of files, and the horizontal axis indicates period. The second column in Table 1 shows periods in which ten types of file were uploaded.

For each type of file, the number of files increased initially, and thereafter decreased in successive periods. This is because the demand for files fluctuates according to a Poisson process and because files are deleted as time passes. Also, each file was maintained at a constant number of files from the vicinity of the respective periods shown in the third column of Table 1.

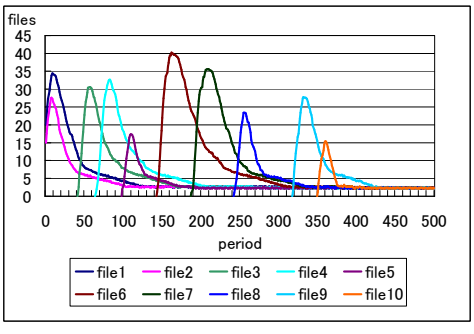

Figure 5. Changes in the numbers of files with the proposed method

Filename	Period when uploaded	Period when replication processing started
file1	0	122
file2	0	104
file3	40	170
file4	65	205
file5	100	174
file6	145	310
file7	190	330
file8	245	335
file9	320	420
file10	350	385

Table 1. Periods where files were uploaded, and where replication processing started

6.3 Comparison of changes in the number of files

With regard to changes in the numbers of *file1*, *file6* and *file10*, Figure 6 shows a comparison of the proposed method with other replication methods. This Figure plots the results in the same way as Figure 5. In all three replication methods, the number of files first increases and then decreases in successive periods. In the proposed method and Owner Replication method, *file1*, *file6* and *file10* changed to more or less the same numbers of files up to 122, 310 and 385 periods, respectively. If we focus on the numbers of each file at period 500, we can see that there

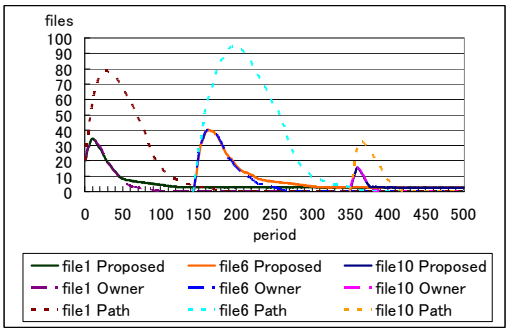

Figure 6. Comparison of changes in the numbers of files

are 2 or 3 files surviving in the proposed method, but none in Owner Replication and Path Replication methods. This shows that the proposed method was able to prevent the loss of *file1*, *file6* and *file10*. We confirmed that the proposed method similarly prevented the loss of the remainder of the 50 types of file used in the experiment.

6.4 Comparison of storage requirements

The storage requirement refers to the total number of files up to a given period, and expresses the consumption of storage resources in all nodes of the system. Figure 7 shows the total storage requirements for *file1*, *file6* and *file10* at each period in the proposed method and Owner Replication and Path Replication methods. The vertical axis shows the total storage requirements for *file1*, *file6* and *file10*, and the unit is the number of files. Storage requirements were the largest with Path Replication and the smallest with Owner Replication. Storage requirements of Path Replication and Owner Replication remained constant after a certain number of periods. On the other hand, we found that storage requirements of the proposed method were characterized by a linear increase as long as no new files are uploaded.

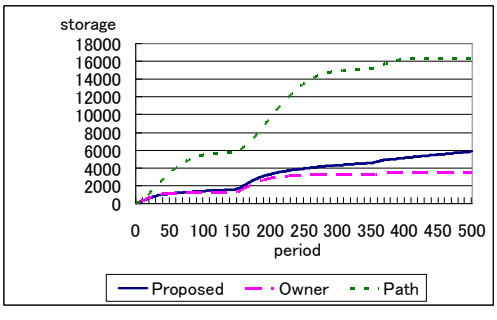

Figure 7. Total storage requirements of file1, file6 and file10

6.5 Discussion

(1) Preventing the loss of low-demand files. With regard to the number of files produced by the proposed replication method, we confirmed that the numbers of files remained more or less fixed once a certain time had passed after the start of experiment. This shows that each file was judged to be a low-demand file at or around the period shown in Table 1, and that low-demand files that are not downloaded are kept on the network by means of replication processing to ensure that the numbers of these files does not fall below some fixed number.

By comparing changes in the numbers of files, we were able to confirm that all the files survived to the end of experiment when using the proposed method, while the file loss occurred with Owner Replication and Path Replication methods. If we focus on the numbers of files generated in the proposed method and

in Owner Replication method, we can see that both methods result in similar behavior up to a certain number of periods, after which they start to diverge. This behavior arises from the replication algorithm of the proposed method, which performs replication operations in the same way as Owner Replication method as long as the file in question is not judged to be a low-demand file. In an experiment consisting of 500 periods, files survived intact with the proposed method but decreased to zero in Owner Replication and Path Replication methods. This shows that the proposed method can prevent the loss of low-demand files.

(2) Storage requirements. With regard to storage requirements, our results show that storage requirements were the largest for Path Replication and the smallest for Owner Replication, and that storage requirements of the proposed method continued to increase linearly. Path Replication causes multiple replica files to be placed each time a download occurs, so it requires more storage than the other two methods. On the other hand Owner Replication needs less storage because it produces just one replica per download request. The proposed replication method combines Owner Replication method with a replication scheme for low-demand files, so its storage requirements are larger than those of Owner Replication method alone. However, since the replication processing of low-demand files only produces a small number of additional files, the proposed method does not consume as many file storage resources as Path Replication.

Using the proposed method, it is possible to prevent the loss of low-demand files, but the survival of these files means that the consumption of node storage resources continues to increase. If we focus on storage requirements of a single file, its storage requirements increase in the proposed method as time passes, becoming larger than storage requirements of Path Replication method. For example, the storage of *file6* at period 500 requires 9,341 units in Path Replication method, and 2,851 in the proposed method. After period 500, if storage requirements for *file6* continue to increase by 2.5 units per period in the proposed method, then the predicted change in storage requirements for *file6* will be as shown in Figure 8. This figure is plotted in the same way as Figure 7.

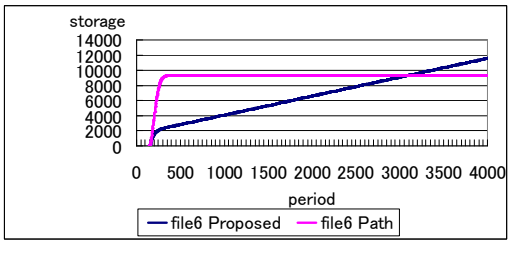

Figure 8. Predicted storage requirements for file6

At period 3,096, predicted storage requirements of the proposed method reach 9,341 units, exceeding storage requirements of Path Replication. However, if we focus on the overall storage requirements for all files in the network, the fact that new files are uploaded as time passes means that storage requirements of Path Replication method also increase.

7. Conclusion

In this paper, we have proposed a replication method based on demand forecasting that prevents the loss of low-demand files in P2P networks. We prevent the loss of files by predicting the demand for files and ensuring that replicas of files are placed in advance. Also, by selecting nodes with high merit as destinations for the placement of low-demand files, it is possible to ensure that these files survive for a long time. To evaluate the proposed method, we constructed a simulation system using a super-node P2P model and performed experiments in the system. From the experiments, the files had survived in the proposed method, but in the Owner Replication and Path Replication methods the number of files had dropped to zero, demonstrating that our proposed method was able to prevent the loss of low-demand files. We also confirmed that the storage costs incurred using the proposed method are larger than those of Owner Replication but smaller than those of Path Replication.

Since the proposed replication operation is performed to keep low-demand files from falling below a fixed number in the proposed method, the node storage requirements will increase if there are more files that are deemed to be in low demand. It is therefore necessary to consider a mechanism for suppressing storage requirements by deleting files for which there is no demand whatsoever. Also, it is necessary to access the detailed performance of the proposed replication method. In this paper we assumed that the demand for files follows Poisson process, but it is possible to assume the other stochastic processes for the demand of files. In the future, it will be necessary to investigate detailed human behavior models for the P2P services.

References

[1] Lv, Q., Cao, P., Cohen, E., Li, K., Shenker, S. (2002), Search and replication in unstructured peer-to-peer networks, In: *Proc. 16th ACM Int'l Conf. on Supercomputing*, 2002.

[2] Cohen, E., Shenker, S. (2002). Replication strategies in unstructured peer-to-peer networks, In: *Proc. of ACM SIGCOMM 2002*, p.177-190, Aug.

[3] Liben-Nowell, D., Balakrishnan, H.,Karger, D. (2002). Analysis of the evolution of peer-to-peer systems, In: *Proc. of the 21st Annual Symposium on Principles of Distributed Computing*, p.233-242.

[4] Stutzbach, D., Rejaie, R (2006). Understanding churn in peerto- peer networks, In: *Proc. of the 6th ACM SIGCOMM Conf. on Internet Measurement*, p.189-202.

[5] Leonard, D. Yao, Z. Rai, V. and Loguinov, D. "On lifetimebased node failure and stochastic resilience of decentralized peer-to-peer networks," *IEEE/ACM Trans. on Networking*, Vol.15, No. 5, pp.1-13, Oct. 2007.

[6] Stutzbach, D., Rejaie, R., Sen, S (2008). Characterizing unstructured overlay topologies in modern P2P file-sharing systems, *IEEE/ACM Trans. on Networking*, 16 (2) p.267-280, April.

[7] Falkner, J., Piatek, M., John, J. P., Krishnamurthy, A., Anderson, T. (2007). Profiling a million user dht, In: *Proc. of the 7th ACM SIGCOMM Conf. on Internet Measurement*, p.129-134.

[8] Rhea, S., Geels, D., Roscoe, T., Kubiatowicz, J. (2004). Handling churn in a DHT, In: *Proc. of the USENIX Annual Technical Conference*.

[9] Androutsellis-Theotokis, S., Spinellis, D (2004). A survey of peer-to-peer content distribution technologies, *ACM Computing Surveys*, 36 (4) 335-371. Dec.

[10] Risson, J., Moors, T (2006). Survey of research towards robust peer-to-peer networks: Search methods, *Computer Networks*, 50 (17) 3485-3521.

[11] Blattberg, R. C., Deighton, J (1996). Manage marketing

by the customer equity test, *Harvard Business Review*, 74 (4) 136-144.

[12] Abe, M. (2004). Incorporating a theory and a model in data analysis for CRM: RF analysis based on a consumer behavior model," Technical Report, CIRJE-J-121, University of Tokyo, Dec.

[13] Axsäter, S. (2006). *Inventory Control*, Second Edition, Springer.

[14] Taylor, H. M., Karlin, S (1998). *An Introduction to Stochastic Modeling*, Third Edition, Academic Press.

[15] Gnutella, http://gnutella.wego.com/

[16] Freenet, http://freenetproject.org/

[17] K. Shudo, Y. Tanaka and S. Sekiguchi, "Overlay Weaver: an overlay construction toolkit," *IPSJ Trans. on Computing Systems*, Vol.47, No.SIG12 (ACS 15), pp.358-367, Sept. 2006.

[18] Overlay Weaver, http://overlayweaver.sourceforge.net/index-j.html

[19] Skype, http://www.skype.com

[20] KaZaA, http://www.kazaa.com/