IEICE TRANS. COMMUN., VOL.E88–B, NO.12 DECEMBER 2005

PAPER

A Cost-Effective Dynamic Content Migration Method in CDNs

Hiroyuki EBARA[†a)], *Member*, **Yasutomo ABE**[†*], **Daisuke IKEDA**[†**], **Tomoya TSUTSUI**[†], **Kazuya SAKAI**[†], *Nonmembers*, **Akiko NAKANIWA**[†***], *and* **Hiromi OKADA**[†], *Members*

SUMMARY Content Distribution Networks (CDNs) are highly advanced architectures for networks on the Internet, providing low latency, scalability, fault tolerance, and load balancing. One of the most important issues to realize these advantages of CDNs is dynamic content allocation to deal with temporal load fluctuation, which provides mirroring of content files in order to distribute user accesses. Since user accesses for content files change over time, the content files need to be reallocated appropriately. In this paper, we propose a cost-effective content migration method called the Step-by-Step (SxS) Migration Algorithm for CDNs, which can dynamically relocate content files while reducing transmission cost. We show that our method maintains sufficient performance while reducing cost in comparison to the conventional shortest-path migration method. Furthermore, we present six life cycle models of content to consider realistic traffic patterns in our simulation experiments. Finally, we evaluate the effectiveness of our SxS Migration Algorithm for dynamic content reconfiguration across time.

key words: *content distribution network (CDN), Internet, content migration, dynamic content allocation, shortest path*

1. Introduction

There has been an explosive increase in the number of users in multi-media networks, particularly on the Internet. Furthermore, the advancement and greater availability of access lines, e.g., ADSL and optical fiber, has enabled users to obtain more bandwidth at a reasonable price, and many content providers are providing more and more attractive content. Content Distribution Networks (CDNs) have attracted growing interests [3], [7], [8]. A CDN is a network optimized to deliver specific content, such as static Web pages, transaction-based Web sites, streaming media, and even real-time video or audio. It is essential to make efficient use of the limited network and server resources to ensure that users can enjoy the services in a stress-free manner. Since user requests concentrate on specific servers containing popular content, an effective load balancing technique is essential to maintain high availability of networks and quality of service (QoS). Popular content files are frequently replicated in multiple servers or caches on the Internet to take the load off of origin servers and to improve the response time for users. A CDN is useful to balance traffic among multiple servers within a network. Load balancing for both servers and a network is possible, if multiple copies of content files are provided on several servers at different locations, and requests are distributed among these servers.

One of the most important issues to realize these advantages of CDNs is content allocation. Currently, caching technology is often used in CDNs. However, we regard this as insufficient to adapt to users' demands, which are explosively increasing. It is much more preferable to allocate multiple copies of content files onto several servers that are geographically separated. Since content allocation strongly affects system cost and performance, it is quite important for content providers to determine how they can efficiently allocate content on their networks. In addition, requests for content change over time. A large number of requests for particular content which are not replicated to multiple servers may significantly decrease load balancing performance in a network. There have been much research on the content allocation problem, but none have considered load fluctuation across time [1], [2], [9], [10]. It is essential to dynamically reallocate content on the server group of the network to maintain the benefit of load balancing. In recent work, dynamic content allocation methods have been widely considered for this reasons [4]–[6], [11], [12].

In this paper, we focus on the content migration method for dynamic content reconfiguration, which refers to changing the entire content allocation in the whole system. A reconfiguration of the entire content allocation obviously causes an increase in the number of content migrations and thus the transmission cost. As far as we know, it has been difficult to find previous research on this issue. In this paper, we propose a Step-by-Step (SxS) Migration Algorithm, which provides the following two features:

(1) A new reconfiguration model of content allocation that makes it possible to reduce transmission cost.
(2) An effective migration algorithm that can be applied to this reconfiguration model.

This reconfiguration model considers the time that it takes to carry out the reconfiguration as a constraint, and it aims to minimize the reallocation cost within this restricted reconfiguration time. We compare the SxS Migration Algorithm with the conventional shortest-path migration method, and we show that this SxS Migration Algorithm is efficient in terms of cost and that it is useful for CDNs where popularity of content dynamically change. Moreover, we present six

Manuscript received January 17, 2005.
Manuscript revised March 30, 2005.
[†]The authors are with the Faculty of Engineering, Kansai University, Suita-shi, 564-8680 Japan.
[*]Presently, with Mitsubishi Electric Corporation.
[**]Presently, with Nomura Research Institute, Ltd.
[***]Presently, with Osaka Sangyo University.
a) E-mail: ebara@eip.kansai-u.ac.jp
DOI: 10.1093/ietcom/e88–b.12.4598

life cycle models of content to consider more realistic traffic patterns in simulation experiments. Finally, we evaluate the validity of the SxS Migration Algorithm for dynamic content reconfiguration across time.

The rest of this paper is organized as follows. We introduce the proposed content reconfiguration model and algorithm in Sect. 2. In Sect. 3, we show the comparison results of the proposed content reallocation method and the conventional method, and we conclude this paper in Sect. 4.

2. SxS Migration Algorithm

2.1 A System Model

Figure 1 shows an example of the system model used in this paper. There exist n servers in the system, and each server is denoted by $S_i (1 \leq i \leq n)$. It is assumed that each user in the system is connected to one of the servers, called a local server. Server S_i has the storage capacity of SC_i. We give the topology of the network by an adjacency matrix $\boldsymbol{A}(a_{ij})$, whose elements have weights relative to the distance between the pair of nodes. We calculate the shortest path matrix $\boldsymbol{Q}(q_{ij})$ from $\boldsymbol{A}(a_{ij})$ to estimate the transmission cost and delay.

There exist l communication links between servers in the system, and each link between servers S_i and S_j is denoted by L_{ij}. We define B_{ij} as the bandwidth of the link L_{ij}. The number of distinct files of content in the system is m, and each file is denoted by $C_k (1 \leq k \leq m)$. Also, it is assumed that the size of the content file C_k is denoted by u_k.

The content allocation of the whole system is expressed by the allocation matrix whose elements are 0-1 variables $FA_{ik}(t)$, which determine whether each content file is allocated on which server (Eq. (1)).

$$FA_{ik}(t) = \begin{cases} 1 \text{ (The content } C_k \text{ is stored in the server } S_i) \\ 0 \text{ (otherwise)} \end{cases} \tag{1}$$

2.2 Reconfiguration Model of Content Allocation

In the content migration method, the problem we deal with is to select the best server from which content is retrieved and to send this content to another server using the best route in terms of reducing transmission cost.

We propose a content reconfiguration model for this purpose. Here, we assume that the old content allocation ($\boldsymbol{Old\_FA}(t)$) before the reconfiguration and the new content allocation ($\boldsymbol{New\_FA}(t)$) after the reconfiguration are known and provided by the allocation matrix defined above. The difference between the old and new content allocation gives us the sets of content files and servers to be additionally allocated $A\_FA_{ik}(t)$ and deleted $D\_FA_{ik}(t)$.

We show an example of the migration method in Fig. 2. The value of a link in Fig. 2 is the distance between adjacent servers. The left-hand side of the figure shows the old content allocation before the content reconfiguration, and the right-hand side shows the new content allocation after the content reconfiguration. In this figure, for example, content file C_2 is added to server S_1 after the reconfiguration, and thus $A\_FA_{12} = 1$. Also, C_3 and C_5 are removed from S_1 resulting in D_FA_{13} and $D\_FA_{15} = 1$. Using this figure, we explain how to reconfigure the content allocation using the conventional method. Let us look at content file C_1. Servers S_1 and S_2 have C_1 before the content reconfiguration. On the other hand, after the content reconfiguration, servers S_1, S_5, and S_6 need to have C_1. The conventional method determines the source and destination servers based only on the shortest route. Thus, server S_5 gets C_1 from server S_1, and server S_6 gets C_1 from server S_2. In our SxS Migration Algorithm, we attempt to reconfigure the content allocation more efficiently. First, server S_5 gets C_1 from server S_1, and next, server S_6 can get C_1 from server S_5 which is closer. The reason why we select server S_5 first is that

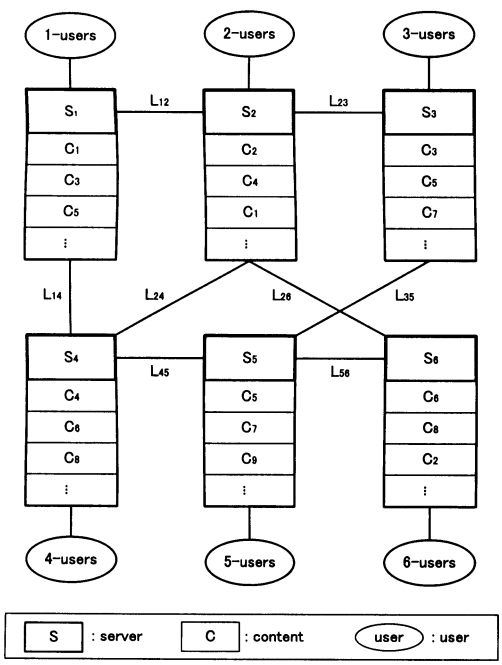


Fig. 1 Example of CDN system model.

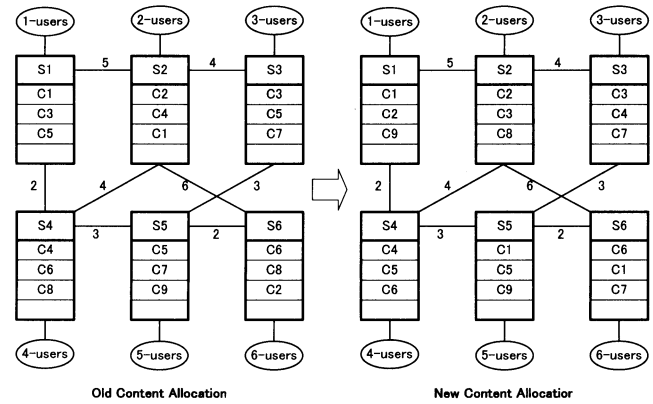


Fig. 2 Example of reconfiguration model.

the distance between servers S_1 and S_5 is shorter than that between servers S_2 and S_6. Our algorithm first deals with the server that retrieves one content file within the shortest distance. Consequently, we can reduce the total cost of migrating content files across the entire system.

In this paper, we propose a reconfiguration model for content allocation, such that the total cost in the reconfiguration of the content allocation in the whole system (hereafter referred to as the reconfiguration cost) is minimized. Moreover, we consider the reconfiguration time taken to carry out the reconfiguration as a constraint. In the next section, we give the detailed formulation of the objective function and the constraint.

2.2.1 Objective Function

In this paper, we define the reconfiguration cost as the cost of changing the old content allocation to the new content allocation throughout the entire system. We especially consider the transmission cost. However, we do not consider the initial setup cost. It is assumed that the reconfiguration cost $Total\_FA\_Cost(t)$ consists of the transmission cost $CT(t)$ and the deletion cost $CD(t)$. The transmission cost $CT(t)$ is similar to that in [11]. Let $\boldsymbol{C}_{tr}$ be the transmission cost coefficient and $\boldsymbol{C}_d$ be the deletion cost coefficient. The total cost is formulated as follows.

$$\begin{aligned} Total\_FA\_Cost(t) &= CT(t) + CD(t) \\ &= \sum_i \sum_k C_{tr} \cdot q\_\min_{ik} \cdot A\_FA_{ik}(t) \cdot u_k \\ &\quad + \sum_i \sum_k C_d \cdot D\_FA_{ik}(t) \end{aligned} \tag{2}$$

where $q\_\min_{ik}$ is the shortest path to migrate the content file C_k to server S_i.

$$q\_\min_{ik} = \min_l + (FA_{lk}(t) \cdot q_{il}) \tag{3}$$

Here, $\min_l +$ is the minimum function on the positive side and q_{il} is the element of the shortest path matrix $\boldsymbol{Q}$.

2.2.2 Restrictive Condition

We set the reconfiguration time as the restrictive condition regarding link capacity. Let T be the restriction of the content reconfiguration time and $E_{ijk}(t)$ be the 0-1 variables indicating whether or not each content migration uses link L_{ij}. The expression of the restrictive condition is as follows.

$$B_{ij} \times T \geq \sum_k E_{ijk}(t) \times u_k \tag{4}$$

$$E_{ijk}(t) = \begin{cases} 0 \text{ (Content item } C_k \text{ does not use link } L_{ij}.) \\ 1 \text{ (Content item } C_k \text{ uses link } L_{ij}.) \end{cases} \tag{5}$$

2.3 Content Migration Algorithm

We propose a content migration algorithm for the content reconfiguration, which can be applied to the reconfiguration model proposed in Sect. 2.2.

When a content file needs to be migrated to multiple servers, this algorithm does not migrate it to all of these servers simultaneously, but rather, it is migrated to each server incrementally. Thus, we can include the server which newly stores the content file as the candidate of source servers. This gives us more opportunities to select closer servers and reduces the transmission cost.

In the following, we give a description of this algorithm.

Step.1 Set $FAO_{ik}(t) = Old\_FA_{ik}(t)$
$FAA_{ik}(t) = A\_FA_{ik}(t)$
$FAD_{ik}(t) = D\_FA_{ik}(t) \quad (1 \leq i \leq n, 1 \leq k \leq m)$

Step.2 Sort all added content files in decreasing order of their content size.

Step.3 Repeat Steps 4–7 for the largest content file $\boldsymbol{C}_k$ which is not yet migrated, until the migration of all content files is completed.

Step.4 List the combination of source servers S_j for which $FAO_{jk}(t) = 1$ and destination servers S_i for which $FAA_{ik}(t) = 1$.

Step.5 For each combination, find a shortest path between the source server and the destination server subject that all the links included in the path satisfy the restrictive condition.

Step.6 Select the combination of servers which have the shortest distance, and determine the destination and source servers.

Step.7 Migrate the content files from the source to the destination server, and update $FAO_{ik}(t)$ to "1" and $FAA_{ik}(t)$ to "0."

Step.8 Delete the content files from those servers where $FAD_{ik}(t)$ equals "1." Then update all $FAD_{ik}(t)$ that are "1" to "0."

3. Performance Evaluations

3.1 Content Migration Methods for Performance Comparison

In this section, we compare two methods: our proposed SxS Migration Algorithm and the conventional method based on the shortest path. When content reallocation over the entire system (i.e. reconfiguration) is executed at every unit time to deal with load fluctuation, there is different content allocation before and after the reconfiguration. When the old and new allocations are given, we carry out the reconfiguration using our proposed method and the conventional method.

In the conventional method, content files are simply migrated along the shortest route. On the other hand, the

SxS Migration Algorithm migrates content files using our cost-effective proposed algorithm. Here, we do not consider the multicast protocol in this paper. Multicast may use UDP for the transport protocol in which no sophisticated control is implemented such as retransmission control of data, error control, order control, flow control, and response confirmation. Consequently, multicast is likely to increase the re-delivery traffic, thus decreasing the reliability of the data.

3.2 System Parameters for Numerical Results

We set the values of the system parameters, considering video on demand (VoD), as follows. The capacity of each server is 100 (Gbyte) (SC_i. = 100000). There exist 200 types of content files in the whole system (k = 200), and the size of each content file is 300–700 [Mbyte] (u_k = 300–700). Also, there must be at least one or more replicated copies of each content file allocated in the system. The bandwidth of each link L_{ij} has the same value 100 [Mbps] (B =100). We assume the initial content allocation ($\boldsymbol{Old\_FA}(t)$) is determined randomly; that is, each content file is allocated to at least one server up to the number of allocated content files (1000–9000) randomly. Similarly the content allocation after the reconfiguration ($\boldsymbol{New\_FA}(t)$) is determined randomly. The difference between the old and new content allocation determines the content files to be added ($A\_FA_{ik}(t)$) and deleted ($D\_FA_{ik}(t)$).

We evaluate the total cost and the reconfiguration time depending on the number of content files, the number of links, the restriction of reconfiguration time, and the number of servers. In the next section, we show the comparison results of the conventional and the proposed method and discuss these results.

3.3 Performance Evaluation of SxS Migration Algorithm

In this section, we evaluate the performance of our proposed method for a single time point over 100 experiments, where the network configuration is generated randomly. All results in this section are averages.

3.3.1 Reconfiguration Cost Characteristics

We show the characteristics of the reconfiguration cost under the constraint condition. In Fig. 3, we show the reconfiguration cost vs. the restriction of reconfiguration time T in the two methods. Here, there are 50 servers (n = 50), all of which have enough capacity to store all content files. Both the old and new content allocations include 3000 content files, each of which is allocated to 15 servers from among the 50 servers on the average. Three network models of varying levels of sparsity are evaluated. Network model (I) is sparse with 150 links. Network model (II) is average with 350 links, and network model (III) is dense with 1000 links. We use the same network models for the rest of the section.

In Fig. 3, first, we can see that the reconfiguration cost of our SxS Migration Algorithm is smaller than the cost of

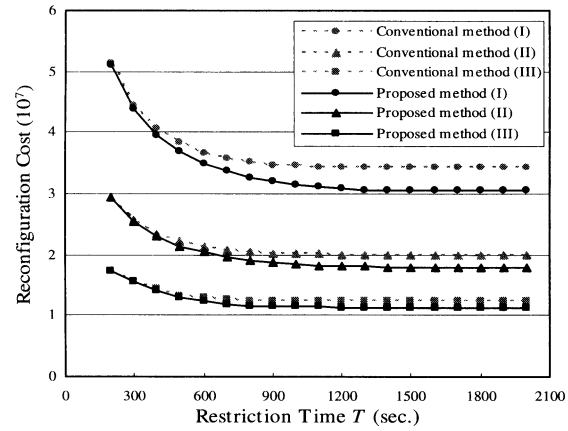


Fig. 3 Reconfiguration cost characteristics.

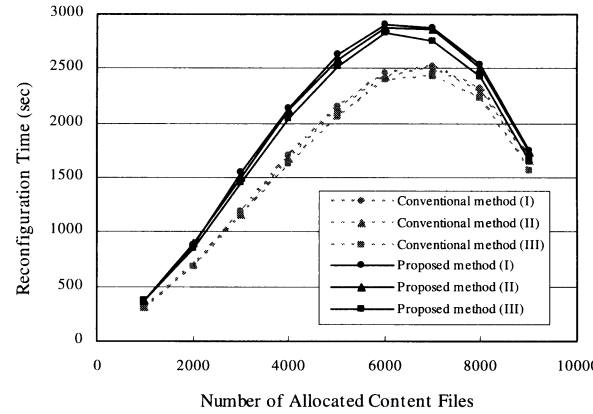


Fig. 4 Reconfiguration time characteristics.

the conventional migration method. Second, we see that as the number of links increases, the reconfiguration cost decreases. This is because we can select closer servers as the number of links increases. We can say that our proposed method can reduce the reconfiguration cost more efficiently for sparse networks where there are fewer links.

In terms of the influence of the constraint, when the restrictive time for the content reconfiguration is long enough, the total cost of the proposed method is smaller than that of the conventional method. However, when the restrictive time is very short, the reconfiguration cost of the proposed method is similar to that of the conventional method. This implies that as the restrictive time is decreased, the capacity of the links that the algorithm can obtain decreases. Consequently, it becomes difficult to select closer servers, and the transmission cost increases even using the SxS Migration Algorithm.

3.3.2 Reconfiguration Time Characteristics

In Fig. 4, we show the reconfiguration time characteristics vs. the number of allocated content files. We assume that the old and new content allocation include the same number of content files. Here, the restriction of the reconfiguration

time T is set to 10000 so as to disregard any influence of the restriction, enabling us to obtain the minimum time for the reconfiguration.

The reconfiguration time of our SxS Migration Algorithm is larger than that of the conventional method. In the proposed method, reconfiguration cost can be reduced using a cost-effective algorithm, but this increases the time it takes to carry out the reconfiguration. In other words, there is a trade-off between the reconfiguration cost and the reconfiguration time. However this disadvantage is indeed negligible. We demonstrate this using an example in network model (I) where we compare the reconfiguration times of the conventional method and the proposed method. When the number of allocated content files is 3000, the reconfiguration time of the conventional method is approximately 1200 (sec.), while the proposed method takes approximately 1500 (sec.), as shown in Fig. 4. Returning to Fig. 3, the corresponding points when the restrictive times are 1200 (sec.) and 1500 (sec.) in the proposed method are approximately 30800000 and 30500000, respectively. This indicates that the reconfiguration costs in both cases are almost equal, meaning that even if the restrictive time is set to 1200 (sec.), the increase of the reconfiguration cost is very little in the proposed method, clearly demonstrating that it is still superior to the conventional method.

3.3.3 Characteristics on the Network Scale

We now illustrate the scalability of the network size when the density is fixed. In Fig. 5, we show the reconfiguration cost vs. the number of servers in the two methods. Here, we assume that both the old and new content allocation include 5000 content files, each of which is allocated to 25 servers on the average. The average degree of each node (server) is set to 6, 14, and 20, for varying levels of network sparsity.

As was shown in Fig. 3, in Fig. 5, the reconfiguration cost of our SxS Migration Algorithm is smaller than that of the conventional method. Also, the reconfiguration cost increases as the number of servers increases, while it decreases as the degree of the servers increases. Moreover, we can see that as the number of servers increases, the difference of the reconfiguration cost between the proposed method and the conventional method increases. It is remarkable that our proposed method can reduce the reconfiguration cost more efficiently for larger networks.

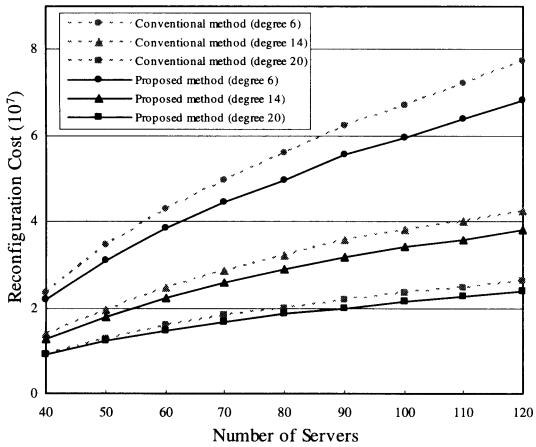


Fig. 5 Characteristics on the network scale.

3.4 Dynamic Reconfiguration Characteristics

In the previous section, we demonstrated the performance of our proposed method at a single time point when content reconfiguration is carried out. Our results showed that our proposed content migration method is useful for content reconfiguration. In this section, we apply our proposed SxS Migration Algorithm to illustrate its dynamic content reconfiguration characteristics across time. Here, the content reconfiguration is carried out at every consecutive unit of time to deal with load fluctuation.

We propose several life cycle models of content to perform realistic simulation experiments; the traffic pattern for each content file is changed over time. It is necessary to reallocate or remove some content files according to the demand of users for load balancing among servers.

There are many kinds of life cycle models depending on the type of content. We classify the life cycle models of content into six types based on an investigation of various content types on the Internet, and we list the definitions of the access flow to the content $\lambda_i(t)$ of each life cycle model in Table 1. This determines the number of allocated content files across time.

First, we explain the life cycle model of Video type content. Figure 6 shows an example of the access flows of the Video type life cycle model. In this model, we assume the content of video on demand systems. One feature of this Video type life cycle model is that the access flow to content is largest when it first appears and then decreases over time. On the other hand, the life cycle model of R-video type content has the opposite feature of the Video type life

Table 1 Access flow of content.

Type of Life Cycle	Access Flow
Video	$\lambda_i(t) = \beta_i \cdot \exp \alpha_i (t - T_a) \quad (T_a \le t \le T_d) \quad (\alpha_i < 0)$
R-video	$\lambda_i(t) = \beta_i \cdot \exp \alpha_i (t - T_a) \quad (T_a \le t \le T_d) \quad (\alpha_i > 0)$
Live	$\lambda_i(t) = K_i \cdot \delta(t - T_a) \quad (T_a \le t \le T_d)$
Campaign	$\lambda_i(t) = \frac{a_i}{2}\left(e^{\frac{2t-T_a-T_d}{2\alpha_i}} + e^{-\frac{2t-T_a-T_d}{2\alpha_i}}\right) + k_i \quad (T_a \le t \le T_d)$
Season	$\lambda_i(t) = \frac{a_i}{\sqrt{2\pi}\sigma_i} \exp\left\{\frac{(t-(T_a+T_d)/2-\mu_i)^2}{2\sigma_i^2}\right\} + k_i \quad (T_a \le t \le T_d)$
Constant	$\lambda_i(t) = W_i \quad (T_a \le t \le T_d)$

T_a : start time; T_d: end time;
α_i, β_i, K_i, a_i, k_i, W_i : parameters assigned random values

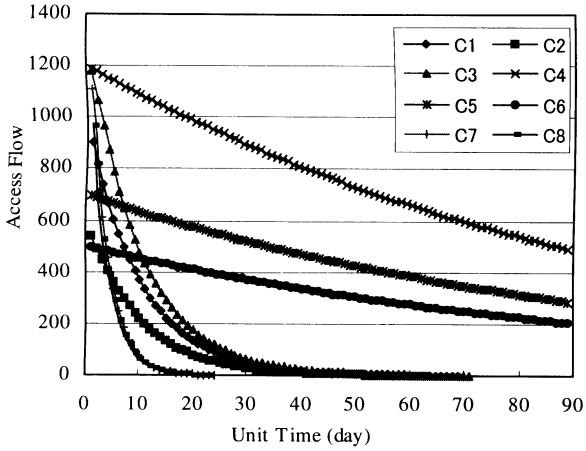


Fig. 6 Access flows of video type life cycle model.

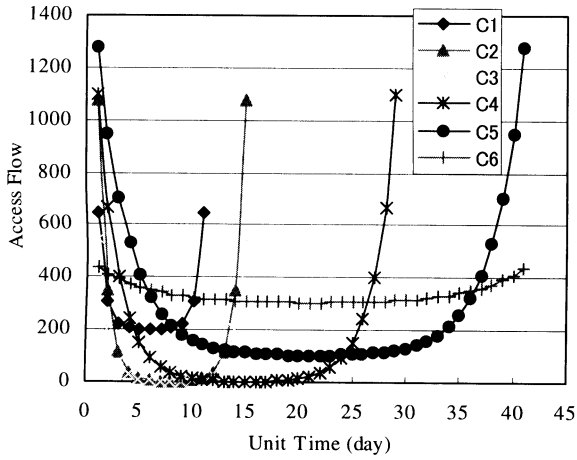


Fig. 7 Access flows of campaign type life cycle model.

cycle model. That is, the access flow to content increases gradually. The life cycle model of Live type content represents live streaming content. A feature of this type is that the access flow to content is concentrated during the live streaming. In the life cycle model of Campaign type content, we consider the content in websites with prizes. A feature of this type is that the access flow to content becomes largest at the beginning and at the end (Fig. 7). The access flow to Season type content has the feature that it becomes large periodically, and the access flow to Constant type content is constant regardless of time.

Next, we present the simulation results from combining the above six types of access flows. Figure 8 plots the reconfiguration cost over the whole system vs. unit time for the two methods. Solid lines indicate averages. We generate the access flows to content at unit time 0 and show the results from unit time 100 as the access flows stabilize. Here, the network model has 50 nodes and 350 links, and there are 200 files of content (100 Video type, 15 R-video type, 10 Live type, 30 Campaign type, 30 Season type, and 15 Constant type). We randomly allocate replicated copies of each content file to servers every unit time, such that the

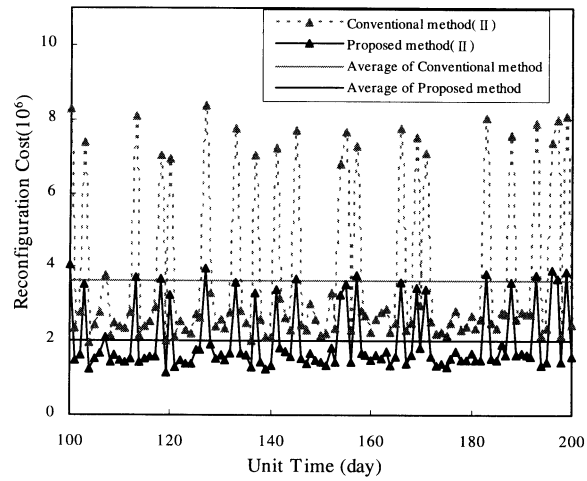


Fig. 8 Dynamic reconfiguration characteristics.

number of content files is proportional to the access flow of each life cycle model. Also, the restriction of the reconfiguration time T is 10000 so that any influence of the restriction is disregarded.

In this figure, first, we can see that the reconfiguration cost of our SxS Migration Algorithm is smaller than that of the conventional method, which is clear when we compare the average reconfiguration cost. Considering temporal change, there are times when the difference of the reconfiguration cost is large. We can imagine that many content files are migrated at these unit times. As the number of reallocated content files increases, the cost of our SxS Migration Algorithm improves.

4. Conclusions

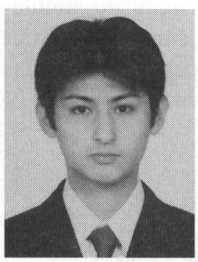
In this paper, we have proposed the SxS Migration Algorithm as a cost-effective content migration method for dynamic content reconfiguration in CDNs. In this content migration method, we have aimed to relocate content files in an efficient way such that we can reduce the transmission cost. We have compared the SxS Migration Algorithm with the conventional shortest path method, and we have shown that our method is capable of reducing transmission cost while maintaining sufficient performance. We have also presented various life cycle models of content and evaluated dynamic content reconfiguration characteristics in realistic simulation environments. Consequently, we can conclude that our content migration method is useful to relocate content files and that it is highly applicable to CDNs where popularity of content dynamically changes.

References

[1] W.W. Chu, "Optimal file allocation in a multiple computer system," IEEE Trans. Comput., vol.C-18, no.10, pp.885–890, Oct. 1969.
[2] C.C. Bisdikian and B.V. Patel, "Cost-based program allocation for distributed multimedia-on-demand systems," IEEE Multimedia, vol.3, no.3, pp.62–72, Fall 1996.

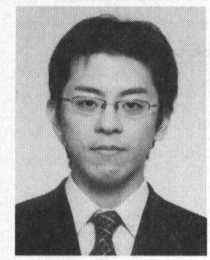
[3] N. Kamiyama, "A server selection method in content delivery networks," IEICE Trans. Commun., vol.E86-B, no.6, pp.1796–1804, June 2003.
[4] T. Watanabe, A. Mori, and Y. Yamamoto, "Mobile cache protocol: A dynamic object relocation protocol for wide area networks," IEEE International Conference on Distributed Computing Systems (ICDCS 2000), pp.420–427, April 2000.
[5] Y. Chen, R.H. Katz, and J.D. Kubiatowicz, "Dynamic replica placement for scalable content delivery," International Workshop on Peer-to-Peer Systems, LNCS 2429, pp.306–318, Springer-Verlag, March 2002.
[6] B. Gavish and O.R.L. Sheng, "Dynamic file migration in distributed computer systems," Commun. ACM, vol.33, no.2, pp.177–189, Feb. 1990.
[7] I. Lazar and W. Terrill, "Exploring content delivery networking," IEEE IT Professional, vol.3, no.4, pp.47–49, July/Aug. 2001.
[8] Stardust.com, White Paper—The Ins and Outs of Content Delivery Networks, Dec. 2000.
[9] A. Nakaniwa, M. Onishi, H. Ebara, and H. Okada, "File allocation in distributed multimedia information networks," Proc. IEEE Globecom '98, pp.740–746, Nov. 1998.
[10] A. Nakaniwa, M. Onishi, H. Ebara, and H. Okada, "Sensitivity analysis in optimal design for distributed file allocation systems," IEICE Trans. Commun., vol.E84-B, no.6, pp.1655–1663, June 2001.
[11] J. Takahashi, A. Nakaniwa, Y. Abe, H. Ebara, and H. Okada, "Load fluctuation-based dynamic file allocation with cost-effective mirror function," IEICE Trans. Commun., vol.E86-B, no.4, pp.1317–1326, April 2003.
[12] A. Nakaniwa, J. Takahashi, Y. Abe, H. Ebara, and H. Okada, "A dynamic file allocation model for serious load fluctuation in the Internet," Proc. CNRS CESA'2003, S3-R-00-0172, July 2003.

Hiroyuki Ebara received the B.S., M.S., and Ph.D. degrees in Communications Engineering from Osaka University, Osaka, Japan, in 1982, 1984, and 1987, respectively. In 1987 he became Assistant Professor of Osaka University. Since 1994 he has been with Kansai University, where he is currently Associate Professor. His main research interests are in computational geometry, combinatorial optimization, and parallel computing. He is a member of IEEE, ACM, SIAM, Information Processing Society of Japan, and OR Society of Japan.

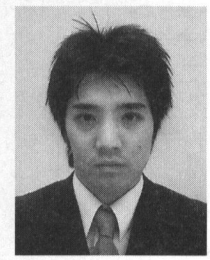


Yasutomo Abe is received his B.S. and M.S. degrees in Electronics Engineering from Kansai University, Osaka, Japan in 2002 and 2004. He extended his research in CDN, Server Load Balancing, and content allocation. He has worked for Mitsubishi Electric Corporation, Tokyo, Japan, since 2004. Currently, he belongs to Information Systems Planning & Engineering Section in Mitsubishi Electric Corporation Communication System Center.

Daisuke Ikeda is received his B.S. and M.S. degrees in Electronics Engineering from Kansai University, Osaka, Japan in 2003 and 2005. He extended his research in CDN, Server Load Balancing, and content allocation. He has worked for Nomura Research Institute, Ltd., Tokyo, Japan, in 2005. Currently, he belongs to IT Platform Services Division in NRI Data Services, Ltd.



Tomoya Tsutsui is received his B.S. degree in Electronics Engineering from Kansai University, Osaka, Japan in 2004. Currently, he is M.S. student in the Information Network Lab. at Kansai University. His research interests are CDN, anycast, and IPv6.



Kazuya Sakai is received his B.S. degree in Electronics Engineering from Kansai University, Osaka, Japan in 2004. Currently, he is M.S. student in the Information Network Lab. at Kansai University. His research interests are P2P, distributed system, and so on.

Akiko Nakaniwa received her B.S., M.S., and Ph.D. degrees in Electronics Engineering from Kansai University, Japan, in 1997, 1999, and 2002, respectively. From 2002 to 2004, she was a post-doctoral fellow in the Research Center of Socionetwork Strategy, Kansai University. Since April 2005, she has been with Osaka Sangyo University as an Assistant Professor. Her main research interest lies in the optimization design of distributed networks. She is a member of IEEE, ACM, and IPS (Information Processing Society) Japan.

Hiromi Okada received the B.S., M.S. and Ph.D. degrees on Communications Engineering, all from Osaka University, Japan, in 1970, 1972, and 1975, respectively. From 1975 to 1983, he was an assistant professor of Osaka University. From 1983 to 1987, he was an associate professor of Kobe University. From 1987 to 1996, he was with Osaka University as an associate professor. Since April 1996, he has been with the department of Electronics Engineering, Faculty of Engineering, Kansai University as a professor. His current research interest includes ATM multicasting networks, wireless ATM networks, ad-hoc wireless networks, performance evaluation and optimization of distributed information networks. He is the author of several books entitled "Information Networks" (in Japanese, Baifukan Pub.), "Introduction of Computer Systems" (in Japanese, Shokodo Pub.) and so on. He is a member of IEEE and IPS (Information Processing Society) Japan.