PAPER

A Cost-Effective Dynamic Content Migration Method in CDNs

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

SUMMARY Content Distribution Networks (CDNs) are highly advanced features for the Internet, and provide low latency, scalability, fault tolerance, and load balancing. One of the most important issues to realize these advantages of CDNs is to consider dynamic content allocation to deal with temporal load fluctuation. The dynamic content allocation provides mirroring of contents in order to distribute user accesses. Since user accesses for contents change depending on time, the contents need to be reallocated. In this paper, we propose a cost-effective content migration method named as a Step-by-Step (SxS) Migration Algorithm in CDNs. We can dynamically relocate contents while reducing the transmission cost by using our content migration method. We show that our method accomplishes sufficient performance and reduces the cost in comparison with the conventional shortest-path migration method. Furthermore, we present six life cycle models of contents to consider more realistic traffic pattern to the simulation experiments. Finally, we evaluate the availability of SxS Migration Algorithm for the dynamic content reconfiguration using our method 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 in the Internet. Furthermore, the progress of access line, e.g., ADSL and optical fiber, has enabled users to obtain more bandwidth at reasonable price, and many content providers are providing more and more attractive contents. Content Distribution Networks (CDNs) have attracted growing interests [3], [7], [8]. A CDN is a network optimized to deliver specific contents, such as static Web pages, transaction-based Web sites, streaming media, or even real-time video or audio. It is essential to make efficient use of the limited network and the server resources to ensure that users can enjoy the service in a stress-free manner. Since user requests concentrate on specific servers including popular contents, an effective load balancing technique is essential to maintain high availability of networks and quality of service (QoS). Popular contents are frequently replicated in multiple servers or caches in the Internet to load origin servers off and to improve the response time of 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 contents are provided in 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 the content allocation. Currently, the caching technology is well used in CDNs. However, we consider this is insufficient to adapt to users' demands that are explosively increasing. It is much more preferable solution to allocate multiple copies of contents in several servers geographically separated. Since the content allocation strongly affects the system cost and performance, it is quite important for content providers to find how they can efficiently allocate contents in their networks. In addition, requests for contents change depending on time. A large number of requests for particular contents which are not replicated to multiple servers may decrease load balancing performance in a network significantly. There have been many research works on the content allocation problem, but none of them have considered the load fluctuation across time [1], [2], [9], [10]. It is essential to dynamically reallocate contents in the server group of the network to maintain the benefit of load balancing. In recent works, 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. The reconfiguration means to change the entire content allocation in the whole system. The reconfiguration of the entire content allocation obviously causes to increase a number of content migration and thus the transmission cost. As far as we know, it is hard to find out the research works on this issue. In this paper, we propose a Step-by-Step (SxS) Migration Algorithm. This SxS Migration Algorithm consists of the following two.

1) A new reconfiguration model of the content allocation that makes it possible to reduce the transmission cost.

Manuscript received January 2001.
Manuscript revised March 1, 2001.

[†] The authors are with Faculty of Engineering, Kansai University, 3-3-35 Yamate-cho, Suita-shi, Osaka, 564-8680 Japan.
* He is now with Mitsubishi Electric Corporation.

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 aims to minimize the reallocation cost within the restriction of the reconfiguration time. We compare SxS Migration Algorithm with the conventional shortest-path migration method, and show this SxS Migration Algorithm is efficient in terms of the cost and useful for CDNs where popularity of contents dynamically changes. Moreover, we present six life cycle models of contents to consider more realistic traffic pattern to the simulation experiments. Finally, we evaluate the validity of SxS Migration Algorithm for the dynamic content reconfiguration across time.

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

2. SxS Migration Algorithm

2.1 A System Model

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

There exist l communication links between servers in the system, and each link between the server 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 contents in the system is m, and each content is denoted by C_k $(1 \le k \le m)$. Also, it is assumed that the size of the content C_k is denoted by u_k.

The content allocation in the whole system is expressed by the allocation matrix whose elements is the 0-1 variables $FA_{ik}(t)$, which determine whether each content is allocated in each server. The 0-1 variable $FA_{ik}(t)$ on the allocation of contents in the sever is defined in equation (1).

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

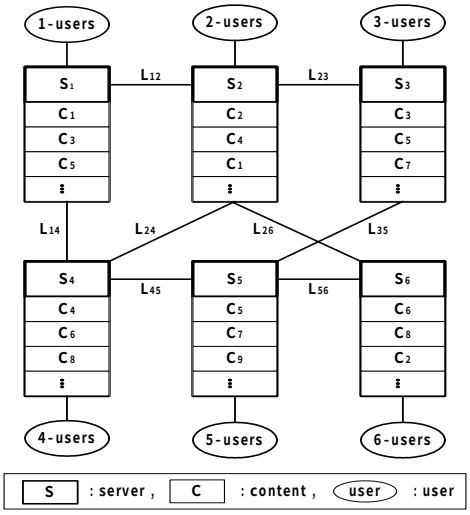

Fig.1 Example of CDN System Model

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 we get a content to the other server and the best route to this server in terms of reducing the transmission cost.

We propose the content reconfiguration model for this purpose. Here, we assume the old content allocation (**Old_FA**(t)) before the reconfiguration and new content allocation (**New_FA**(t)) after the reconfiguration are known and given using the allocation matrix defined above. The difference between the old and new content allocation gives us the sets of contents and servers to be allocated additionally A_FA_{ik} (t) and to be deleted D_FA_{ik} (t).

We show an example of the migration method in Fig.2. The value on a link in Fig.2 is a distance between adjacent servers. The left figure shows the old content allocation before the content reconfiguration. The right figure shows the new content allocation after the content reconfiguration. In this figure, for example, the content C_2 is added to the server S_1 after the reconfiguration, and thus A_FA_{12} = 1. Also, C_3 and C_5 is removed from S_1 and thus D_FA_{13} and D_FA_{15} = 1. Using this figure, we explain how to reconfigure the content allocation in the conventional method. Suppose, we would pay attention to the content C_1. The server S_1 and S_2 have the content C_1 before the content reconfiguration. On the other hand, after the content reconfiguration, the server S_1, S_5, and S_6 need to have the content C_1. The conventional method determines the source and destination server depending on only the shortest route. Thus, the server S_5 gets the content C_1 from the server S_1, and the server S_6 gets the content C_1

from the server S_2. In our SxS Migration Algorithm, we attempt to reconfigure the content allocation more efficiently. First, the server S_5 gets the content C_1 from the server S_1, and next, the server S_6 can get the content C_1 from the server S_5 which is closer. The reason why we select the server S_5 first is that the distance from the server S_1 to the server S_5 is shorter than the distance from the server S_2 to the server S_6. Our algorithm first deals with the server that gets one content in the shortest distance. Consequently, we can reduce the total cost to migrate contents in the whole system.

In this paper, we propose the reconfiguration model of the content allocation, such that the total cost in the reconfiguration of the content allocation in the whole system (the reconfiguration cost, hereafter) is minimized. Moreover, we consider the reconfiguration time that it takes to carry out the reconfiguration as a constraint. In the following, we give the details of the formulation of the objective function and the constraint.

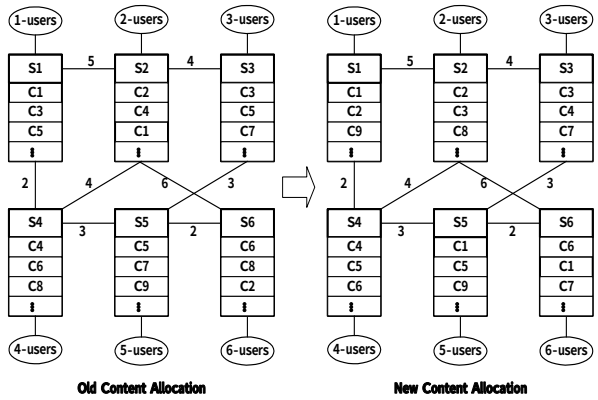

Fig.2 Example of Reconfiguration Model

(a) Objective Function

In this paper, we define the reconfiguration cost as the cost to change the old content allocation to the new content allocation in the whole system. We especially consider the transmission cost. However, we don't 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.

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

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

$$q\_\min_{ik} = \min_{l}{}^{+}\left(FA_{lk}(t) \cdot q_{il}\right) \quad (3)$$

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

(b) 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 whether each content migration uses each link. The expression of the restrictive condition is as follows.

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

$$E_{ijk}(t) = \begin{cases} 0 & (The\ content\ C_k\ \ doesn't\ \ pass\ \ the\ \ link\ \ L_{ij}.) \\ 1 & (The\ content\ C_k\ \ passes\ \ the\ \ link\ \ L_{ij}.) \end{cases} \quad (5)$$

2.3 Content Migration Algorithm

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

When each content needs to be migrated to multiple servers, this algorithm doesn't migrate it to these servers simultaneously, but to each server incrementally. And thus, we can include the server which newly store the content in the candidate. 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 contents in the decreasing order of their content size.

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

Step.4 List the combination of the source servers S_j for which $FAO_{jk}(t) = 1$ and the 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 satisfies the restrictive condition.

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

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

Step.8 Delete contents from servers where FAD_{ik} (t) equals "1". We update All "1"s of FAD_{ik} (t) to "0"s.

3. Performance Evaluations

3.1 Content Migration Methods for Performance Comparison

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

In the conventional method, contents are simply migrated along the shortest route. On the other hand, in the SxS Migration Algorithm, we migrate the content using our cost-effective proposed algorithm. Here, we don't consider the multicast protocol in this paper. The multicast may use the UDP for the transport protocol in which no sophisticated control is implemented such as the retransmission control of data, the error control, the order control, the flow control, and the response confirmation. Consequently, the multicast is likely to increase the re-delivery traffic, and decrease the reliability of data.

3.2 System Parameters for Numerical Results

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

We evaluate the total cost and the reconfiguration time with changing the number of contents, 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 describe our considerations regarding these results.

3.3 Performance Evaluation of SxS Migration Algorithm

In this section, we evaluate the performance of our proposed method on a certain time when the content reconfiguration is carried out. We make 100 experiments, where network configuration is generated randomly. Each result in this section is an average.

(a) 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 exist 50 servers (n = 50) such that servers have enough capacity to store all contents. The both old and new content allocations include 3000 contents, where each content is allocated to 15 servers among 50 servers on the average. The network model (I) has 150 links, which means this network is sparse. The network model (II) has 350 links, which is general. The network model (III) has 1000 links, which is dense. We use the same network models for the rest of the section.

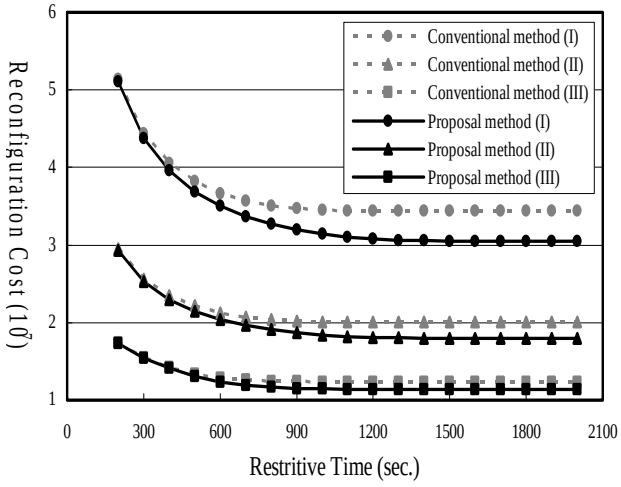

Fig.3 Reconfiguration Cost Characteristics

In this figure, first we can see that the reconfiguration cost in our SxS Migration Algorithm is smaller than the cost in the conventional migration method. Second, the more the number of links is, the less the reconfiguration cost is. This is because we can select nearer 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 the number of links is less.

In terms of the influence of the constraint, when the restrictive time for the content reconfiguration is long enough, the total cost in the proposal method is smaller than in the conventional method. However, when the restrictive time is very short, the reconfiguration cost in the proposal method is almost similar with in the conventional method. We can consider that the stricter the restrictive time is, the less capacity of links the algorithm can obtain. Consequently, it becomes difficult to select nearer servers and the transmission cost increases even in the SxS Migration Algorithm.

(b) Reconfiguration Time Characteristics

In Fig.4, we show the reconfiguration time characteristics vs. the number of the allocated contents. We assume the old content allocation and the new content allocation include the same number of contents. Here, the restriction of the reconfiguration time T is 10000 that means we consider no influence of the restriction, and thus the reconfiguration takes minimum time.

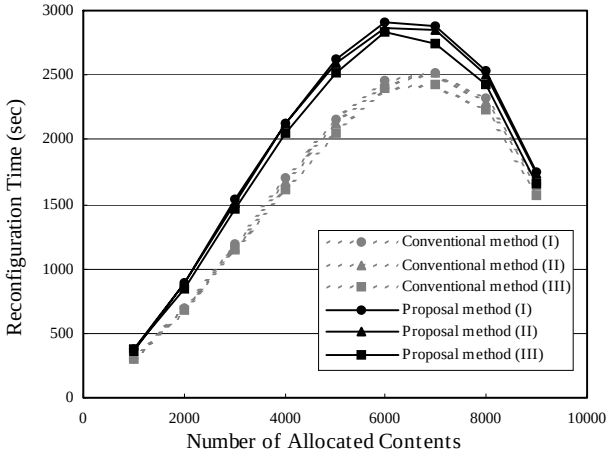

Fig.4 Reconfiguration Time Characteristics

The reconfiguration time in our SxS Migration Algorithm is larger than in the conventional method. In the proposal method, reconfiguration cost can be reduced using the cost-effective algorithm, while the time it takes to carry out the reconfiguration becomes longer. Namely there is a trade-off between the reconfiguration cost and the reconfiguration time. However this disadvantage is indeed negligible. Now we demonstrate it by using an example in the network model (I). We compare the two values of the reconfiguration time in the conventional method and the proposal method. When the number of allocated contents is 3000, the reconfiguration time in the conventional method is about 1200(sec.), while in the proposal method it is about 1500(sec.) in Fig.4. Now go back to Fig.3, and see the correspondent points in the case that the restrictive time is 1200(sec.) and 1500(sec.) in the proposal method. The reconfiguration costs of the case of restrictive time 1200(sec.) and 1500(sec.) are approximately 30800000 and 30500000 respectively. We consider that the reconfiguration costs in the both cases are almost equal. This means even if we set the restrictive time as 1200(sec.), the increase of the reconfiguration cost is very little in the proposal method and it still superior to the conventional method obviously.

(c) Characteristics on the network scale

We show the characteristics on the network size in which the density is fixed. In Fig.5, we show the reconfiguration cost vs. the number of servers in two methods. Here, we assume the both old and new content allocation includes 5000 contents. Each content is allocated to 25 servers on the average. The average degree of each node (server) is set as 6, 14, and 20 - sparse, general, and dense, respectively.

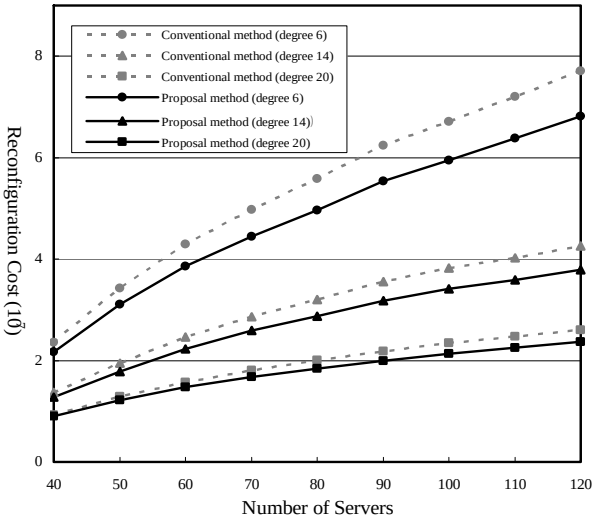

Fig.5 Characteristics on the network scale

In this figure, similarly to shown in Fig.3, the reconfiguration cost in our SxS Migration Algorithm is smaller than in the conventional method. Also, the more number of servers is, the more the reconfiguration cost is.

And, the more degree of servers is, the less the reconfiguration cost is. Moreover, we can see that as the number of servers increases, the difference of the reconfiguration cost between the proposal method and the conventional method becomes large. It is remarkable that our proposed method can reduce the reconfiguration cost more efficiently for larger networks.

3.4 Dynamic Reconfigration Characteristics

In the previous section, we have shown the performance of our proposed method on a certain time when the content reconfiguration is carried out. As a result, it has shown that our proposed content migration method is useful for the content reconfiguration. In this section, we apply our proposed SxS Migration Algorithm to show the dynamic content reconfiguration characteristics across time. Here, the content reconfiguration is carried out at every unit time to deal with load fluctuation.

We propose the life cycle models of contents to make the simulation experiments more realistic. The traffic pattern for each content changes along with time. It is necessary to reallocate or remove some contents according to the demand of users for the load balancing among servers.

There are many kinds of the life cycle models depending on the type of contents. We classify the life cycle models of contents into six kinds through checking up contents on the internet, and define the access flow to contents $\lambda_i(t)$ of each life cycle model in Table 1. The access flow to each content determines the number of allocated contents.

Table 1. Access Flow of Contents

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	$\lambda_i(t) = \frac{a_i}{2}\left(e^{\frac{2t - T_a - T_d}{2a_i}} + e^{\frac{2t - T_a - T_d}{2a_i}} \right) + k_i \quad (T_a \le t \le T_d)$
Season	$\lambda_i(t) = \frac{a_i}{\sqrt{2\pi}\sigma_i} \exp\left\{ \frac{\left(t - (T_a + T_d)/2 - \mu_i\right)^2}{2\sigma_i^2} \right\} + k_i \quad (T_a \le t \le T_d)$
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 defined randomly;

First, we explain the life cycle model of Video type contents. Fig.6 shows an example of access flows of the Video type life cycle model. In this model, we assume the contents on video on demand systems. The feature of this Video type life cycle model is that the access flow to contents is the largest when appeared and then decreases as time goes by. In the other hand, the life cycle model of R-video type contents has the reverse feature of the Video type life cycle model. Namely, the access flow to contents increases gradually. In the life cycle model of Live type contents, we assume the contents of live streaming. The feature of this type is that the access flow to contents is concentrated during the live streaming. In the life cycle model of Campaign type contents, we assume the contents in websites with prizes. The feature of this type is that the access flow to contents becomes largest in the beginning and the end (Fig.7). The life cycle model of Season type contents has the feature that the access flow to contents becomes large periodically, and the access flow to Constant type contents is constant regardless of time.

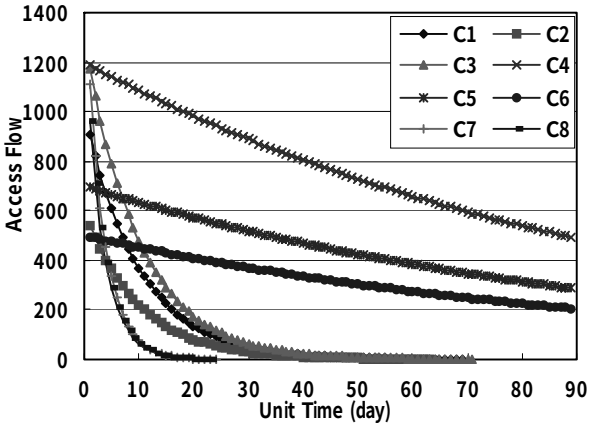

Fig.6 Access Flows of Video Type Life Cycle Model

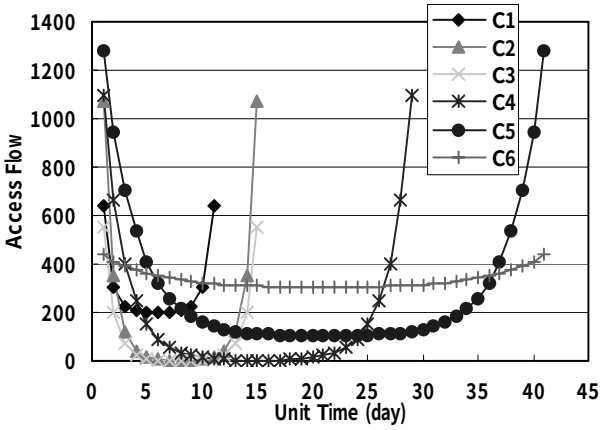

Fig.7 Access Flows of Campaign Type Life Cycle Model

In this paper, we show the simulation results under applying the combination of the above 6 kinds of access flows. Fig.8 shows the reconfiguration cost in the whole system vs. the unit time in the two methods. Lines show the averages on the two methods. We generate the access flows to contents at the unit time 0 and show the results from the unit time 100 as the access flows become steady. Here, the network model has 50 nodes and 350 links, and there exist 200 kinds of contents (100 kinds of Video type contents, 15 kinds of R-video type contents, 10 kinds of Live type contents, 30 kinds of Campaign type contents, 30 kinds of Season type contents, and 15 kinds of Constant type contents). We allocate replicated copies of each content to servers randomly every unit time, whose number is proportional to the access flow of each life cycle model. And, the restriction of the reconfiguration time T is 10000 that means we consider no influence of the restriction.

In this figure, first we can see that the reconfiguration cost in our SxS Migration Algorithm is smaller than in the conventional method, which is clear when we compare the average reconfiguration cost. When paying attention at the temporal change, there exits time when the difference of the reconfiguration cost is large. We can imagine many contents are migrated in these unit times. The more the number of reallocated content is, the more the cost of our SxS Migration Algorithm is improved. We think that these life cycle models make it possible to simulate more realistic network environments.

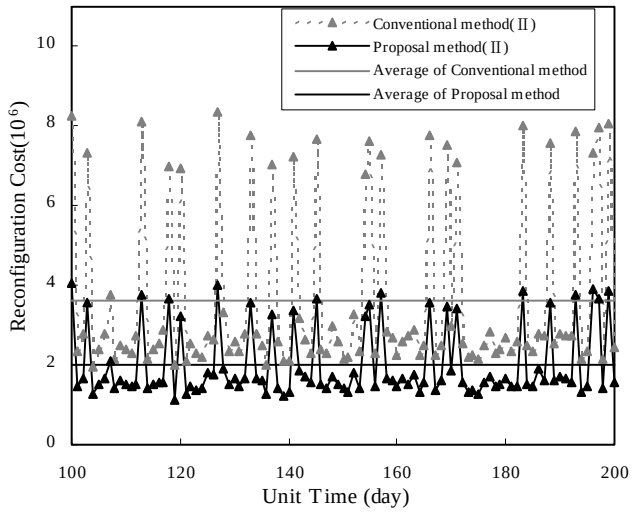

Fig.8 Dynamic Reconfiguration Characteristics

4. Conclusions

In this paper, we have proposed SxS Migration Algorithm as a cost-effective content migration method for the dynamic content reconfiguration in CDNs. In this content migration method, we have aimed to relocate contents 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 have shown that our method is capable of reducing the transmission cost while accomplishing enough performance. We have also presented the life cycle models of contents, and evaluated the dynamic content reconfiguration characteristics in more realistic simulation environments. Consequently, we can conclude that our content migration method is useful to relocate contents and expected to be applicable to CDNs where popularity of contents 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, B.V. Patel, “Cost-based program allocation for distributed multimedia-on-demand systems,” IEEE Multimedia, pp.62-72, Fall 1996.
[3] N. Kamiyama, “A Server Selection Method in Content Delivery Networks,” IEICE Trans. Commun., vol.E86-B, no.6 Jun 2003, pp1796-1804.
[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) pp420-P427, Apr. 2000.
[5] Y.Chen, R.H.Katz and J.D.Kubiatowicz, "Dynamic Replica Placement for Salable Content Delivery," International Workshop on Peer-to-Peer Systems, Mar. 2002.
[6] Bezalel Gavish, Olivia R. Liu Sheng, "Dynamic file migration in distributed computer systems," Communications of the ACM, Vol.33, Issues 2, Feb. 1990.
[7] I.Lazar, W.Terrill, "Exploring content delivery networking," IEEE IT Professional, Vol.3, Issue 4, Jul/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.of IEEE Globecom ’98, 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, Jun. 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, Apr 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", in Proceedings of CNRS CESA'2003, Jul. 2003.