Scalable Maintenance for Strong Web Consistency in Dynamic Content Delivery Overlays

Zhou SU, Jiro KATTO, and Yasuhiko YASUDA, Member, IEEE

***Abstract*— Content Delivery Overlays improves end-user performance by replicating Web contents on a group of geographically distributed sites interconnected over the Internet. However, with the development whereby overlay systems can manage dynamically changing files, an important issue to be resolved is consistency management, which means the cached replicas on different sites must be updated if the originals change. In this paper, based on the analytical formulation of object freshness, web access distribution and network topology, we derive a novel algorithm as follows: (1) For a given content which has been changed on its original server, only a limited number of its replicas instead of all replicas are updated. (2) After a replica has been selected for update, the latest version will be sent from an algorithm-decided site instead of from its original server. Simulation results verify that the proposed algorithm provides much better consistency management than conventional methods with the reduced the hit ratio and network traffic**.

***Index Terms*— content delivery networks, consistency algorithm, web cache performance, network traffic**

I. INTRODUCTION

With the growth in popularity of the Internet and the wide availability of streaming applications, how to efficiently distribute the stored content has become a major concern in the Internet community.

Some content delivery networks (CDN) [3][4] have emerged, and they work directly with content providers to cache and replicate the providers' content close to the end users by using geographically distributed edge servers. More recently, some other researchers have also advocated using an overlay CDN structure composed of dedicated transit nodes to distribute the large contents [19].

Although both CDN and its improved version (Overlay CDN) facilitate static file sharing, newly-developed applications, such as online auction and remote collaboration, demand that they should be able to manage dynamically-changing files. There has been some research [6,],[9],[12],[16],[18] on this problem, which is called consistency management. However, most of these studies treat different replicas of the same content to be managed for Web consistency in the same manner. Furthermore, how to optimally select a surrogate instead of an original server to update the content has not been discussed.

In this paper, we therefore propose an optimal algorithm for controlling Web consistency in Content Delivery Overlay, which includes both conventional CDN and the improved Overlay CDN. Firstly, we carry out a theoretical analysis of the Web access and the freshness time of objects. Based on the analytical result, we then propose a *consistency priority* and assign different priorities to different replicas of the same content. When a given content is changed at its original server, instead of all its replicas over the whole overlay network, only its replicas with high *consistency priorites* will be updated.

Secondly, if one replica of a given content is selected to be updated, the latest version of this content will be sent from a surrogate with the lowest *update priority*, which is proposed based on the network topology and bandwidth. Therefore, the latest version will be sent from an algorithm-decided site instead of from its original server to reduce the network traffic.

Finally, through simulations we check the performance of our proposal when the related parameters are changed, and find that our proposal can efficiently improve the hit ratio and network traffic against the previous algorithms. We also show that the necessary parameters in our proposed algorithm can be obtained from the information readily available in the local overlay.

This paper is organized as follows: in Sect.2, an overview of the CDN overlay system is provided. In Sect.3, related work with regard to consistency management algorithms is reviewed. Sect.4 presents mathematical analyses of Web access, average AS hop and user delay. And our proposed algorithm is also presented. In Sect.5, extensive simulation results are given and conclusions are presented in Sect.6.

II. CONTENT DELIVERY OVERLAYS

How to efficiently distribute the Web content has attracted

Manuscript received October 9, 2001. (Write the date on which you submitted your paper for review.) This work was supported in part by the U.S. Department of Commerce under Grant BS123456 (sponsor and financial support acknowledgment goes here). Paper titles should be written in uppercase and lowercase letters, not all uppercase. Avoid writing long formulas with subscripts in the title; short formulas that identify the elements are fine (e.g., "Nd–Fe–B"). Do not write "(Invited)" in the title. Full names of authors are preferred in the author field, but are not required. Put a space between authors' initials.

much research. Content delivery networks (CDNs) appeared recently and are deploying quite rapidly [1]~[4],[29]. Their concern is mainly placed on efficient delivery of static content, i.e. HTML files and images. Some CDN companies advocate their support for the dynamically changed content, but their technical details are not yet clarified nor verified.

In Peer to Peer (P2P) networks, users can determine from where different files can be downloaded with the help of a directory service [7], [11]. Peer-to-peer systems such as Napster [5] depend on little or no dedicated infrastructure. There is, however, the implicit assumption that the individual peers participate for a significant length of time instead.

Recently, ideas of Overlay Network, where each connection in the overlay is mapped onto a path in the underlying physical network, are being discussed to facilitate both CDN and P2P. For example, Kazza [12] organizes the clients into an overlay P2P network. But the performance of overlay changes with time as nodes dynamically join and leave the system. [19] proposed an overlay CDN architecture where a set of intermediaries act as TNs, which organize themselves into a content delivery overlay and are used to replicate and forward data

Our work can be used in both in conventional CDNs and the improved Overlay CDNs. In this paper, it focuses on how to maintain the Web consistency and reduce the network traffic caused by consistency management in content delivery overlay.

III. Related Work

In *Propagation* method, the updated version of a document is delivered to all copies whenever a change is made to the document at the origin server. Although the copies always keep the latest version of the originals by the *Propagation*, this method may generate significant levels of unnecessary traffic if documents are updated more frequently than accessed.

Some Web services employ the time to live (*TTL*) mechanism [8] to refresh their replicas. However, how to decide the proper value of *TTL* is still not resolved.

In *Invalidation* [6], an invalidation message is sent to all copies when a document is changed at the origin server. This method does not make full use of the delivery network for content delivery and each replica needs to fetch an updated version individually at a later time. Therefore, the user-delay may get worse if a frequently accessed document can not be updated on time.

[9] addressed a set of models that capture the characteristic of dynamic content at the sub-document level in terms of independent parameters such as the distribution of objects size, their refresh times and reusability across time.

Cluster Lease [16] was designed to maintain data consistency by propagating server notifications to cluster of proxies in the content delivery networks. However, how to reduce the network traffic caused by the propagation between server and proxies is not mentioned.

[18] proposed a hybrid approach that can generate less traffic than the propagation approach and the invalidation approach. The origin server makes the decision of using either propagation or invalidation method for each document, based on the statistics about the update frequency at the origin server and the request rates collected by replicas. However, the algorithm only takes the request frequency into consideration. More discussion should be continued.

MONARCH [12] divided Web objects into several different groups based on object relationships and object change characteristics. Furthermore, it identifies the relationships among objects composing a page and used relationships to keep all objects consistent. However, how to cooperatively keep the consistency among replicas stored in different sites has not been resolved.

We ourselves proposed an integrated pre-fetching and replacing algorithm for the hierarchical image based on a cooperative proxy-server model, in which the metadata of the hierarchical image was used to keep the data consistency with user-satisfaction [23]. We also presented a scheme for stream caching by using hierarchically distributed proxies with adaptive segments assignment, in which "segment" meant a group of pictures [27]. This method clarified effectiveness of "local-scope" server cooperation (in the overlay network) with per-segment management and discussed how to reduce the overhead in overlay network.

IV. Theory analysis

In this section, we give an analysis of Web consistency over the Content Delivery Overlay. Firstly, in Sect.4.1 we introduce the notations used in the network model. Secondly, theoretical analyses of Web access distribution and average hop count are presented from Sect.4.2 to Sect.4.3, respectively. Then, based on the analytical results, how to reduce user perceived latency is discussed in Sect.4.4. The mathematically optimized consistency algorithm is proposed in Sect.4.5. Finally, How to reduce the computation complexity and how to get the necessary parameters are introduced in Sect.4.6.

A. Notations

We assume that each surrogate is located in a different administrative domain, such as an autonomous system (AS). Let S_i (bytes) denote storage capacity of a server in domain i (i=1,…, I), and λ_i (bytes/second) denote an aggregate request rate from clients to the server.

As for the contents, we assume that there are J different contents in our CDN. A parameter P_j defines the request probability for content j (i.e., content popularity). Here, B_j denotes the data size of content j. In this paper we look on content j as an update object specified by (j). And its origin server is defined as $o(j)$.

Let $X_{i,j,}$ be a parameter which takes a binary value of

$X_{i,j} = 1$ (if object j is stored in server i)

$X_{i,j} = 0$ (otherwise) (1)

Then, we can get a matrix $\boldsymbol{X}$ of which one element is $X_{i,j}$, which represents a placement pattern of contents. As for the link between two servers, $D_{m,n}(\boldsymbol{X})$ means the shortest distance (hop

count) from server *m* to server *n* under the placement ***X***. And $C_{m,n}$ denotes the average bandwidth (per hop) along the above path from server *m* to server *n*.

B. Definition of Web Access Distribution

Let $\Lambda = \sum_i \lambda_i$ be the total request rate from all the domains. Then, for a given surrogate *i*, its surrogate popularity can be given by λ_i / Λ.

According to the Zipf distribution which the distribution of Web access follows, the probability that the content *j* is requested can be obtained as follows:

$$P(j) = \frac{\Omega}{r_j^{\alpha}} \qquad (2)$$

where Ω , α are parameters of the Zipf distribution, and r_j is the ranking of request times. Therefore, we can get the probability that a request happens for the *j*-th content from surrogate *i* by:

$$P(i, j) = \frac{\Omega}{r_j^{\alpha}} \cdot (\lambda_i / \Lambda) = \frac{\Omega \cdot \lambda_i}{r_j^{\alpha} \cdot \Lambda} \quad (3)$$

Recent studies [9] show that the freshness time of objects follws a Weibull distribution with a CDF:

$$F(x) = 1 - e^{1-(ax)^b} \qquad (4)$$

Furthermore, for content *j*, the mean *E(x)* (called MTTF or MTBF) of this distribution is given by:

$$E(X_j) = a_j^{-1/b_j} \cdot \Gamma(1 + \frac{1}{b_j}) \quad (5)$$

where a_j, b_j are parameters of the Weibull distribution.

Assume the time when content *j* was updated last time is $t_{0,j}$, if in the period from $t_{0,j}$ to ($t_{0,j}$ + $E(x_j)$) there are $W_{E(xj)}$ total request happened in the whole CDN system, then the number of request times for the *j*-th content happened from surrogate *i* within this period can be obtained by:

$$R_{i,j} = W_{E(X_j)} \cdot \frac{\Omega}{r_j^{\alpha}} \cdot (\lambda_i / \Lambda) = \frac{\Omega \cdot \lambda_i \cdot W_{E(X_j)}}{r_j^{\alpha} \cdot \Lambda} \qquad (6)$$

If the $R_{i,j}$ is greater than one, it means at least one request happens for this object since it has been changed last time. To avoid sending the invalidation version of the data, the replica of content *j* on surrogate *i* should be updated when the original changes next time.

C. Minimization of User Perceived Latency

In this subsection, we made the analysis of network traffic cause by sending the modified version of a document. Our goal is to fetch the latest version of the modified document from an alternative surrogate instead of the original sever to minimize the user perceived latency.

When a request happens for object *(j)* (content *j*) from a given surrogate *t* where the latest version of object *(j)* is not available, if we assume that there are *K* surrogates where the latest version of object *(j)* (content *j*) is available except the original server *o(j)*, for *k*={1,...,*K*}, we can get:

$$K <= I \;\&\; k \neq t$$
$$k \neq o(j) \;\&\; X_{k,j} = 1 \qquad (10)$$

Assume that content *j* is originally stored in server *o*(*j*) and $C_{k,t}$ is the average bandwidth (per hop) during the path from surrogate *k* to surrogate *t*. Then, if a client sends a request for obeject (*j*) to surrogate *t* and surrogate *k* sends the latest version to this client, the user delay during the delivery from server *k* to server *t* is given by

$$T_{k,t,j}(\mathrm{X}) = \frac{1}{\Lambda} \lambda_t \cdot B_j \cdot P_j \cdot D_{k,t}(\mathrm{X}) / C_{k,t}(\mathrm{X}) \quad (11)$$

$D_{k,t}(\boldsymbol{X}_0)$ is the shortest distance from server k to server t under the initial placement pattern **X** and $\Lambda = \sum_i \lambda_i$ is the total request rate from all the domains.

If we continue to define:

$$G_j = \frac{1}{\Lambda} \cdot B_j \cdot P_j \qquad (12)$$

$$U_{k,t}(\mathrm{X}) = D_{k,t}(\mathrm{X}) / C_{k,t}(\mathrm{X}) \quad (13)$$

it can be obtained:

$$T_{k,t,j}(\mathrm{X}) = \lambda_t \cdot G_j \cdot U_{k,t}(\mathrm{X}) \quad (14)$$

Similarly, if a client sends a request for object (*j*) to surrogate *t* and original server *o*(*j*) sends the latest version to this client, the user delay during the delivery from server *k* to server *o*(*j*) is given by

$$T_{o(j),t,j}(\mathrm{X}) = \lambda_t \cdot G_j \cdot U_{o(j),t}(\mathrm{X}) \qquad (15)$$

For a given surrogate *k*, we can calculate the reduced user delay $\Delta T_{k,j}$ by taking a difference of Eq.(14) and Eq.(15)

$$\Delta T_{k,j} = G_j \cdot \lambda_t \cdot (U_{o(j),t}(\mathrm{X}) - U_{k,t}(\mathrm{X})) \quad (16)$$

D. Computational Complexity and Related Parameters

Since the scale of CDN is being increased recently, to manage all surrogates' all replicas' update will cause a great amount of computational complexity. How to reduce the Computational Complexity should be considered. Fortunately, previous researches [4] showed that the distribution of web requests from a fixed group of users follows a Zipf-like distribution, where most of web requests to surrogates are just for a very small set of objects, for example top 10 %. Furthermore, in [5] it had been found that 80% of the requests to the Web contents is served by only top 4% most popular surrogates. Therefore, to reduce computation complexity of our algorithm, it is suggested that we only need to calculate the priorities for popular surrogates and popular contents.

As for how to get the necessary parameters to carry out the proposal: In our algorithm, $W_{E(xj)}$ need to be calculated to obtain *consistency priority* in Eq.6. The Web log of CDN keeps the records of the request times when time goes on. If the fresh time $E(x_j)$) can be predicted, the total request times $W_{E(xj)}$ during the period of $E(x_j)$) can be obtained. In order to grasp the current update period of the object, an auto-regressive (AR) model can be used to predict its current update period from its past records which are available as local information in CDN. Besides, the bandwidth $C_{k,t}$ needs to be measured to calculate

update priority in Eq.11. There are lots of methods to measure the available bandwidth including some algorithms such as *Pathchar*, *Packet Pair* and *SloPS* with a guaranteed accuracy.

E. Proposed Algorithm

1. Step1: Scalable Update Selection

When a given content *j* changes at server *o(j)*, a *consistency priority* $R_{i,j}$ will be calculated according to Eq.6. For content *j* 's each replica ($X_{i,j}$ =1) over the whole overlay network, only when its priority $R_{i,j}$ is beyond the threshold *Th*, the replica of content *j* at surrogate *i* will be updated.

Otherwise, this replica will not be updated until a new request for content *j* happens at the site *i* next time.

Therefore, when a given content *j* is changed at its original server, not all its replicas ($X_{i,j}$ = 1) over all overlay network will be updated according to the analysis of Web access distribution.

2. Step2: Scalable Lowest Delay Update

Assume that there are *K* ($(X_{k,j}=1,\ k=\{1,\ldots,K\}\ \&\ K<=I)$) replicas of content *j* selected to be updated, for a replica at surrogate *k* , an *update priority* $\Delta T_{i,j}$ will be calculated according to Eq.16. The latest version of content *j* will be sent to surrogate *k* from surrogate *i* with the lowest $\Delta T_{i,j}$.

Therefore, the latest version will be sent from an algorithm-decided site instead of its original server resulting the reduction of network traffic and user delay.

F. Computational Complexity and Related Parameters

Since the scale of CDN is being increased recently, to manage all surrogates' all replicas' update will cause a great amount of computational complexity. How to reduce the Computational Complexity should be considered. Fortunately, previous researches [20] showed that the distribution of web requests from a fixed group of users follows a Zipf-like distribution, where most of web requests to surrogates are just for a very small set of objects, for example top 10 %. Furthermore, in [28] it had been found that 80% of the requests to the Web contents is served by only top 4% most popular surrogates. Therefore, to reduce computation complexity of our algorithm, it is suggested that we only need to calculate the priorities for popular surrogates and popular contents.

As for how to get the necessary parameters to carry out the proposal: In our algorithm, $W_{E(xj)}$ need to be calculated to obtain *consistency priority* in Eq.6. The Web log of CDN keeps the records of the request times when time goes on. If the fresh time $E(x_j)$) can be predicted, the total request times $W_{E(xj)}$ during the period of $E(x_j)$) can be obtained. In order to grasp the current update period of the object, an auto-regressive (AR) model can be used to predict its current update period from its past records which are available as local information in CDN. This method has been used for image caching in [23], and the efficiency of predicting Web access by AR model or other similar models in the real network has also been proved in [32][33].

Besides, the bandwidth $C_{k,t}$ needs to be measured to calculate *update priority* in Eq.11. There are lots of methods to measure the available bandwidth including some algorithms such as *Pathchar* [34], *Packet Pair* [30] and *SloPS* [31] with a guaranteed accuracy.

V. Evaluation of Algorithms

In this section numerical results will be presented by simulation experiments to validate the proposed algorithm.

A. Simulation Conditions

In simulation experiments, we assume the following conditions:

There are 21 nodes (servers) in our network simulator. Among these nodes, there are 15 original servers and 6 content servers (surrogates), respectively. As a Recent study showed that most communication networks have Power-Law link distributions [28][15], where the *i*'th most connected node has Ω/r_i^{β} neighbors, as for the network topology, we carry out our proposal under the Power-Law link distribution.

Because the distribution of web request has already proved to follow a Zipf distribution, which states that the relative probability of requests for the *i*'th most popular page is proportional to Ω/r_i^{α}, the access frequency is decided by this Zipf distribution with a Zipf parameter 0.8 [20][21].

About the contents, there are 1000 different objects. We assume that the average size of objects is 10 KBytes and the size of an invalidation message for each object is 100 Bytes [18].

Client requests arrive according to a Poisson process. All clients are always redirected to the closest server without failure of request routing. The update rate of a given object is decided at random. The total request times in the simulations are 100000.

There are five replication algorithms we will study:

- *Propagation Policy*
- *Invalidation Policy*
- *Hybrid Policy*
- *Monarch Policy*
- *Proposal*

To evaluate different algorithms, we use two performance measures. One is traffic generated during the process of sending a latest version of a given content. The other is *Old Hit*, which is the percentage of objects are invalid (not the latest version) when a request arrives at the replica.

B. Simulation Results

Figure 1 shows the result of *Old Hit*, which means that the requested data is not of the new version. Because the updated version of the requested document is delivered to all copies when a change is made at its origin server in the *Propagation*, its *Old Hit* is zero. However, the network traffic generated by the *Propagation* is very serious in Fig.2, where the traffic caused by sending the new version is shown. Here, the network traffic is calculated by multiple the data size with the traversed AS Hops.

As for the *Invalidation* [6], when a change is made at its origin server, this method only sends an invalidation message instead of the content itself. Then, the client need to wait for the

new version sent from the original server after the request. As a result, the *Old Hit* is the worst. Furthermore, sending the invalidation messages to all surrogates where the copies are stored also causes the additional network traffic. As a result, the total traffic can not be decreased efficiently.

The *Hybrid Policy* [18] sets a threshold based on the request frequency, if the request is beyond the threshold, the *Propagation Policy* is carried out, otherwise, *Invalidation Policy* will be used. In *Monarch Policy* [12], objects are divided into different groups based on the object change frequency. Different groups manage their objects separately according to the predicted update-period.

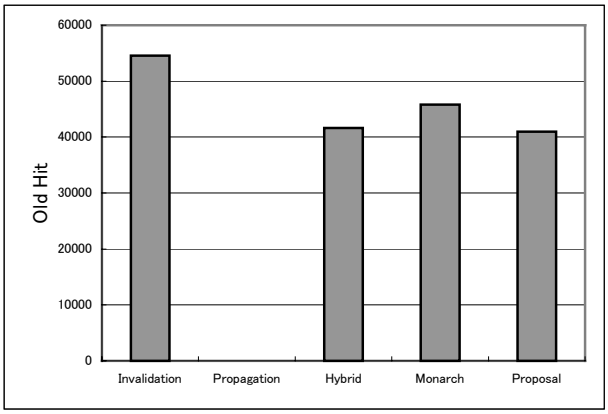

Fig.1: Comparison of Old Hit with Different Replication Algorithms.

From the results in Fig.1 and Fig.2, we can obtain the following conclusions: The two conventional methods (*Propagation Policy* and *Invalidation Policy*) have their own drawbacks respectively. The *Propagation Policy* generates extremely huge network traffic and the *Invalidation Policy* causes the highest *Old Hit* and more network traffic compared with the other 3 methods (*Hybrid Policy, Monarch Policy and Proposal*). As for the above 3 methods, they can get the balance of two conventional ones: Compared with the *Invalidation*, their traffics are closed to it but theirs *Old Hits* are much better. Compared with the *Propagation*, although the *Old Hit*s are more than the *Propagation*, their traffic can be greatly reduced.

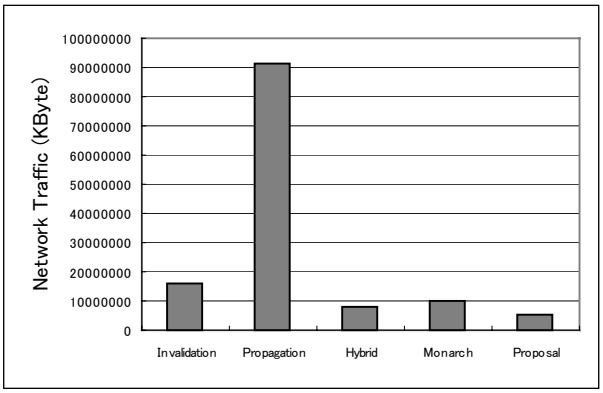

Fig. 2: Comparison of Network Traffic with Different Replication Algorithms.

Since these 3 methods (*Hybrid Policy, Monarch Policy and Proposal*)' performances seem quite close compared with the other two methods (*Propagation Policy* and *Invalidation Policy*) in Fig1 and Fig2, we continue to test the performances of these 3 algorithms when the related parameters are changed.

We firstly tested the *Old Hit* with respect to the simulation times. From Fig.3, it shows that the proposed algorithm obtains stable performance when the simulation times are increased. It always outperforms the other two algorithms (*Hybrid Policy and Monarch Policy*). The reason is because the proposal decides whether an object should be updated in a surrogate by considering not only contents request frequency, but also server popularity and fresh distribution.

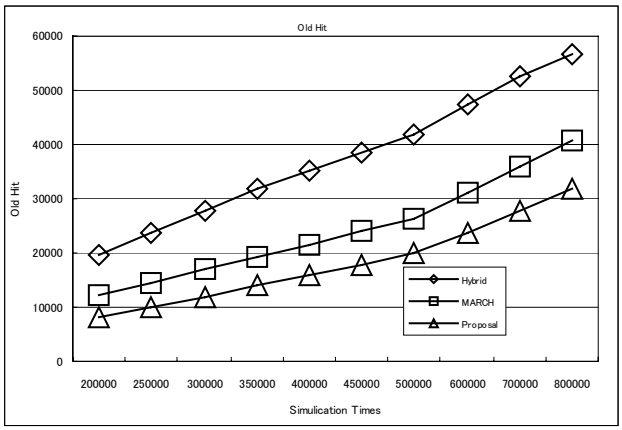

Fig. 3: Old Hit under Different Simulation Times

We then do the simulation about network traffic when the different simulation times are carried out. Similar result is shown in Fig. 4, where the proposed one reduced network traffic most. In our proposal, the client gets the latest version of the request content from an algorithm-decided surrogate, network traffic can be reduced because the traversed Hops can be decreased at the same time.

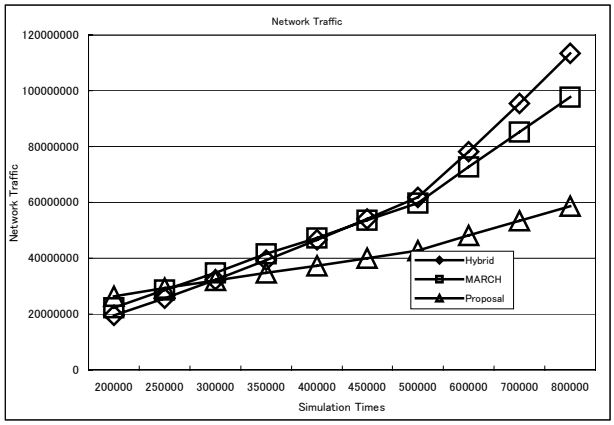

Fig. 4: Network Traffic under Different Simulation Times

Several researchers have observed that the distribution of web request from a fixed group of users follows a Zipf distribution. Besides, the value of α , a parameter of Zipf distribution, varies from trace to trace, ranging from 0.64 to 0.83 [20][21]. We then varied the Zipf parameter and get the results in Fig.5 and Fig.6.

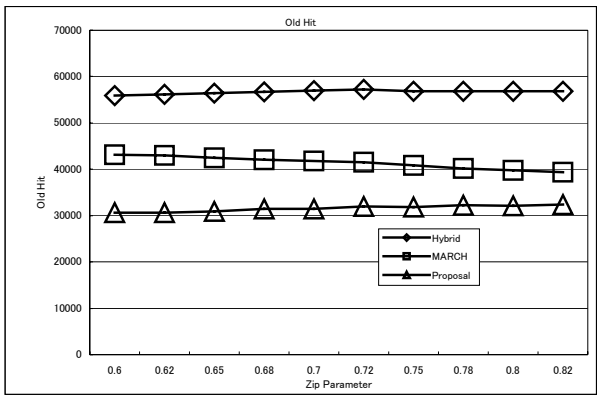

Fig. 5: Old Hit under Different Zipf Parameters

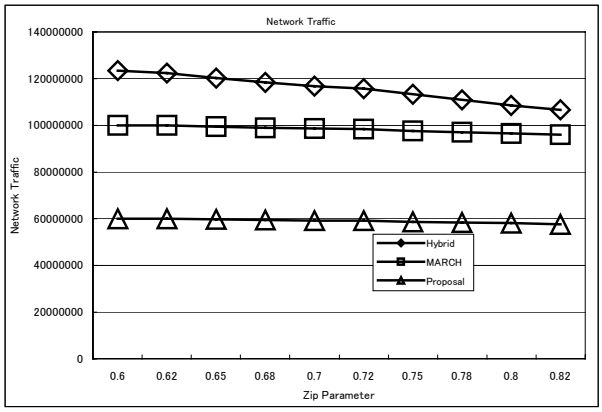

Fig. 6: Network Traffic under Different Zipf Parameters

Finally, we change the relative cache size to test the performance of each algorithm. The relative cache size means the percentage of all contents that a surrogate can store in.

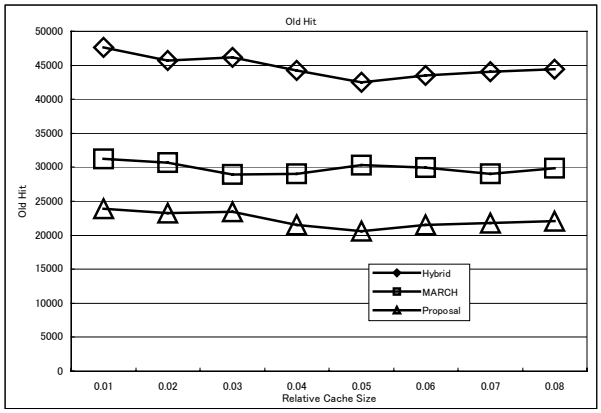

Fig. 7: Old Hit under Different Relative Cache Size

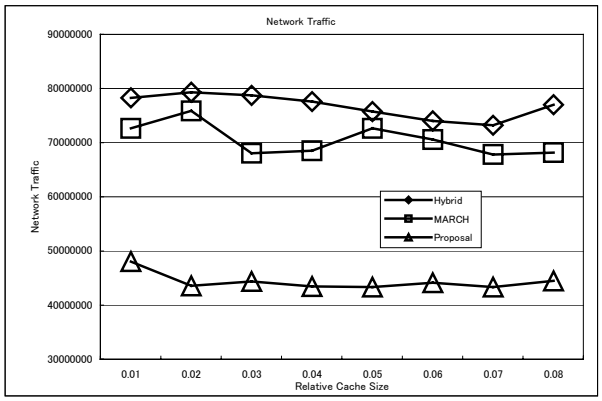

Fig. 8: Network Traffic under Different Relative Cache Size

VI. CONCLUSION

This paper discussed how to optimally manage Web consistency among replicas in content distribution overlay networks and presented an efficient scheme to update them without wasting surrogates' resources. Based on mathematical analysis, we proposed a novel algorithm to minimize average user delay over traversed domains where the scalable content consistency is obtained. Our proposal dealt with not only popularities of contents and servers but also server load. We then compared our proposal with other conventional methods using computer simulations.

There are a number of works to be done as further researches. First, we plan to do more simulation results when other parameter and network topology are also considered. Secondly, theoretical analysis should be expanded to be applicable to general cases. Finally, implementation is to be carried out.

REFERENCES

[1] J.Kangasharju and K.W. Ross," Performance Evaluation of Redirection Schemes in Content Distribution Networks", The 5th International Web Caching and Content Delivery Workshop, May 2000.

[2] A.Beck and M. Hofmann, "Enabling the Internet to Delivery Content-Oriented Services" Proc. The 6th International Web Caching and Content Distribution, Boston, USA Jun 2001.

[3] Adero, <URL: http: www. adero. com>

[4] Akamai, <URL: http: www. akamai. com>

[5] Napster, < URL: http://www.napster.com>

[6] P. Cao and C. Liu, "Maintaining Strong Cache Consistency in theWorld-Wide Web," Proc. 17th Int'l Conf. Distributed ComputingSystems, May 1997.

[7] H M. Radha, M.V.D. Schaar, and Y. Chen, "The MPEG-4 Fine Grained Scalable Video Coding Method for Multimedia Streaming Over IP ", IEEE Trans. Multimedia Vol.3, No.1, pp.53-67, March 2001.

[8] V. Cate, "Alex: A Global File System," Proc. 1992 USENIX FileSystem Workshop, pp. 1-12, May 1992.

[9] W.Shi, E.Collins, and V,Karamcheti, **"**Modeling object characteristics of dynamic Web content", IEEE Globecom 2002,Taiwan,2002 Nov.

[10] M. Chesire, A. Wolman, G.M.Voelker, and H.M.Levy, "Measurement and Analysis of a Stream Media Workload", USITIS'01, San Francisco, CA, Mar.2001.

[11] S. Acharya and B. Smith and P. Parnes, "Characterizing User Access To Videos on the Videos On the World Wide Web'' SPIE/ACM MMCN 2000, San Jose, CA, Jan 2000.

[12] Mikhail Mikhailov, Craig E. Wills, "Evaluating a New Approach to Strong Web Cache Consistency with Snapshots of Collected Content", The Twelfth International World Wide Web Conference, 20-24 May 2003, Budapest, HUNGARY

[13] I. Cidon, S. Kutten, and R. Soffer. “Optimal allocation of electronic content”, IEEE Infocom, Anchorage, AK, April, 2001.

[14] B. Li, M. J. Golin, G. F. Italiano, and X. Deng. “On the optimal placement of web proxies in the internet”. IEEE Infocom, New York, NY, pp 21-25, March 1999.

[15] L. Qiu, V. N. Padmanabhan, and G. M. Voelker. “On the placement of web server replicas”. IEEE Infocom, Anchorage, AK, pp 22-26, April 2001.

[16] A.Ninan, P.Kulkarni, P.Shenoy, K.Ramamritham, R.Tewari “Scalable Consistency Maintenance in Content Distribution Networks Using Cooperative Leases, IEEE Transactions on Knowledge and Data Engineering, July/August 2003

[17] Sung-Ju Lee, Wei-Ying Ma, and Bo Shen ”An Interactive Video Delivery and Caching System Using Video Summarization”, WCW*2001,* Boston, MA, June 2001

[18] Zongming Fei, ``A novel approach to managing consistency in content distribution networks", Proceedings of the 6th International Web Caching and Content Distribtution Workshop (WCW'01), Boston, MA, June 2001, pp.71-86

[19] S.Ganguly, A. Saxena, S.Bhatnagar, S.Banerjee, R.Izmailov “Fast Replication in Content Distribution Overlays”, IEEE Infocom, Miami FL, 2005

[20] L. Breslao, P. Cao, L. Fan, G. Phillips, and S. Shenker “Web Caching and Zip-like Distributions: Evidence and Implications” Proc. IEEE INFOCOM’99, New York, April, 1999.

[21] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "On the Implications of Zipf's Law for Web Caching", In 3rd International WWW Caching Workshop, June 1998.

[22] Kazza, < URL: http://www.kazza.com>

[23] Z.Su, T.Washizawa, J.Katto, and Y.Yasuda, “Integrated Pre-fetching and Caching Algorithm for Graceful Image Caching”, IEICE Trans on Commun, Vol.E86-B, No.9, Sep. 2003, pp 2753-2763

[24] Q.Lv, P.Cao, E.Cohen, K.Li, S.Shenker, “Search and Replication in Unstructured Peer-to-Peer Networks”. ICS 2002

[25] G. Antoniol, G. Casazza, G. Di Lucca, M. Di Penta, and E. Merlo, “Predicting Web Site Access: An Application of Time Series”, Proceedings of IEEE the third International Workshop on Web Site Evolution, Florence, Nov. 2001

[26] J.E.Pitkow, and M.R.Recker, "A simple yet robust caching algorithm-based on dynamic access patterns", Proceedings of the Second World-Wide Web Conference, Amsterdam, 1994

[27] Z.Su, J.Katto, T.Nishikawa, M.Murakami and Y.Yasuda, “Stream Caching Using Hierarchically Distributed Proxies with Adaptive Segments Assignment”, IEICE Trans on Commun, Vol.E86-B, No.6, Jun. 2003, pp 1859-1869

[28] L. A. Adamic, B. Humberman, R. Lukose, and A. Puniyani. “Search in Power Law Networks” Phys. Rev. E, Vol. 64, 2001

[29] M.Sasabe, N.Wakamiya, M.Murata, and H.Miyahara, "Proxy caching mechanisms with video quality adjustment", SPIE ITCom Feb. 2001

[30] S. Keshav, “A Control-theoretic Approach to Flow Control”. Proceedings of SIGCOMM91, Zurich, Sep, 1991

[31] M. Jain and C. Dovrolis, “End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput,” Proceedings of SIGCOMM02, Pittsburgh, PA, Aug. 2002

[32] G. Antoniol, G. Casazza, G. Di Lucca, M. Di Penta, and E. Merlo, “Predicting Web Site Access: An Application of Time Series”, Proceedings of IEEE the third International Workshop on Web Site Evolution, Florence, Nov. 2001

[33] J.E.Pitkow, and M.R.Recker, "A simple yet robust caching algorithm-based on dynamic access patterns", Proceedings of the Second World-Wide Web Conference, Amsterdam, 1994

[34] V.Jacobson. “Pathchar” ftp://ftp.ee.lbl.gov/pathchar/, 1997