# Supporting Consistency Management in Dynamic Content Distribution Overlays

Zhou Su, Jiro Katto, Yasuhiko Yasuda
*Faculty of Science and engineering, Waseda University*
Contact E-mail: suzhou@waseda.jp

## Abstract

*Content Distribution 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 time, web access distribution and network topology, we derive a novel algorithm as follows: (1) For a given content which has been changed at 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 update overhead and network traffic.*

## 1. 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 distribution networks (CDN) [1]~[5],[7],11] 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 content [12]]~[14], [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 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 Distribution 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 this 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 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 overhead 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.

## 2. Related Work

In the *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

Proceedings of the Joint International Conference on Autonomic and Autonomous Systems
and International Conference on Networking and Services (ICAS/ICNS 2005)
0-7695-2450-8/05 $20.00 © 2005 IEEE

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 the *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 distribution 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't 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 a cluster of proxies in the content distribution 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 the *Propagation* or the *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. 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 [24]. This method clarified the effectiveness of "local-scope" server cooperation (in the overlay network) with per-segment management and discussed how to reduce the overhead in overlay network.

## 3. Theoretical Analysis

### 3.1 Notations

| Parameters | Definition |
|---|---|
| *(j)* | content *j* |
| *(i,j)* | Replica of content *j* at surrogate *i* |
| *o(j)* | Server originally stores *(j)* |
| $r_i$ | Ranking of request times of *(j)* |
| $\lambda_i$ | Aggregate request rate to server *i* |
| $R_{i,j}$ | Consistency priority of *(i,j)* |
| $\Delta T_{i,j}$ | Update priority of *(i,j)* |
| $P_j$ | Request probability of object *(j)* |
| $X_{i,j}$ | Element of placement matrix $\boldsymbol{X}$ |
| $D_{m,n}(\boldsymbol{X})$ | Shortest distance (hop count) from server *m* to server *n* under the placement $\boldsymbol{X}$ |
| $C_{m,n}(\boldsymbol{X})$ | Average bandwidth (per hop) during the path from server *m* to server *n* under the placement $\boldsymbol{X}$ |

Table 1: Client Workload and Model Parameters

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 $\lambda_i$ (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). In this paper we look on content *j* as one 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 $\boldsymbol{X}$. And $C_{m,n}$ denotes the average bandwidth (per hop) along the above path from server *m* to server *n*. All of the above defined parameters are listed in Table 1.

### 3.2 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 $\lambda_i / \Lambda$.

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

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

where $\Omega$, $\alpha$ 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 follows a Weibull distribution with a CDF:

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

Furthermore, for content *j*, the mean $E(x_j)$ (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 that the time when content *j* was last updated is $t_{0,j}$, and in the period from $t_{0,j}$ to $(t_{0,j} + E(x_j))$ there are $W_{E(X_j)}$ total requests happned 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:

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

If $R_{i,j}$ is greater than 1, it means at least one request has been made 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 timely updated when the original changes.

### 3.3 Definition of Average Hop Count

In a conventional method, if a surrogate decides to fetch the latest version of a modified document, it will contact the modified document's original server to fetch its latest version.

If we define an average hop count $H_{k,j,o(j)}(\boldsymbol{X})$, which represents the traverse of the latest version of content *j* from surrograte *k* (where a stale copy of content *j* is sotored ) to the original server o (*j*). It can be obtained by:

$$H_{k,j,o(j)} = D_{k,o(j)}(X) \cdot (1 / \sum_i \lambda_i) \cdot \lambda_k \cdot P_j = D_{k,o(j)}(X) \cdot S_{k,j} \quad (7)$$

$$S_{k,j} = (1 / \sum_i \lambda_i) \cdot \lambda_k \cdot P_j \quad (8)$$

Eq.(8) represents a request probability weighted by aggregation ratio to server *i* (i.e. joint probability of stream popularity and server popularity). The parameter $D_{k,o(j)}(\boldsymbol{X})$ denotes the shortest distance (hop count) from domain *k* to server $o(j)$ storing the object (*j*) under the placement $\boldsymbol{X}$ with

$$X_{k,j} = 1 \quad (9)$$

In the above method, as all update requests are sent to the same original server $o(j)$ to fetch the latest version of content *j*, it causes poor user-delay and load balance. Some researchers also advocate fetching the latest version from another surrogate instead of the original server. However, how to select a proper surrogate is still a problem to be dealt with.

### 3.4 Minimization of User Perceived Latency

In a conventional method, if a surrogate decides to fetch the latest version of a modified document, it will contact the modified document's original server to fetch its latest version. Because all update requests are sent to the same original server $o(j)$ to fetch the latest version of content *j*, it always causes poor user-delay and load balance.

In this subsection, we made the analysis of network traffic cause by sending the modified version of a document. Our goal is to take 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 expect the original server *o( j)*, for $k=\{1,...,K\}$, we can get:

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

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

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

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

If we continue to define:

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

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

it can be obtained:

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

Similarly, if a client sends a request for obeject ($j$) 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

$$T_{k,o(j),j}(\mathrm{X}) = \lambda_k \cdot G_j \cdot U_{k,o(j)}(\mathrm{X}) \quad (15)$$

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)

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

### 3.5 Proposed Algorithm

We present our algorithm as follows:

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 every replica of content $j$ ($X_{i,j}$=1, $i$={1,...,$I$}) over the whole overlay network, only 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 the entire overlay network will be updated to avoid the unneccsary waste of update.

Step2: Lowest Delay Updating

Assume that there are $K$ ( $(X_{k,j}$ =1, $k$={1,...,$K$} & $K$<=$I$) ) surrogates which keep the latest version of content $j$, whne we decide to select which sorrogate to provide the latest version to the client, an update priority $\Delta T_{k,j}$ will be calculated according to Eq.16. The latest version of content $j$ will be sent to the client from surrogate $k$ with the lowest $\Delta T_{k,j}$.

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

## 4. Evaluation of Algorithms

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

### 4.1 Simulation Conditions

Recent studies show that most communication networks have Power-Law link distributions [22], where the $i$'th most connected node has $\Omega/r_i^{\beta}$ neighbors. As for the network topology, to show the performances of our proposal under different network topologies, we carry out our proposal under the Power-Law link distribution as the second step and the parameter $\beta$ in the Power-Law link distribution is set to be $0.8$ [15].

Because the distribution of web requests 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 $\Omega/r_i^{\alpha}$, the access frequency is decided by this Zipf distribution with $\alpha$ $=0.8$ [20][21].

About the contents, there are 1000 different contents. Client requests arrive according to a Poisson process [10][17]. All clients are always redirected to the closest server without failure of request routing. The total request times in the simulations are 10000.

There are three replication algorithms we will study:

*Propagation Policy*
*Invalidation Policy*
*Proposal*

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

### 4.2 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 caused by the *Propagation* is very serious as in Fig.2, where the traffic caused by sending the new version is shown. As for the *Invalidation*, although its traffic is the least, its *Old Hit* is the worst.

Our proposal can balance the two conventional ones: Compared with the *Invalidation*, their traffics are closed to each other, but the *Old Hit* of our proposal is much better. Compared with the *Propagation*, although the *Old Hit* of our proposal is more than that of the *Propagation*, its traffic can be greatly reduced. The network traffic is calculated by multiplying the data size by the traversed AS Hops.

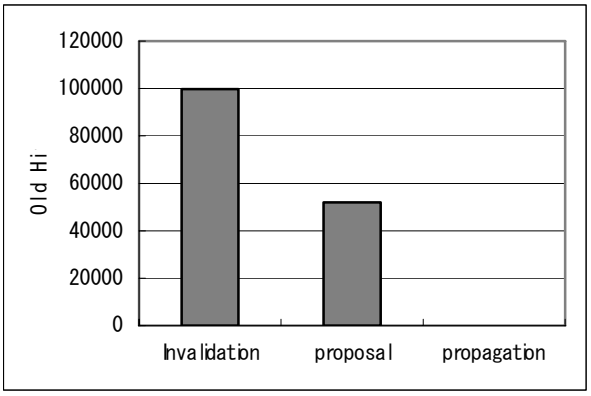


Fig.1: Comparison of Old Hit with Different Replication Algorithms.

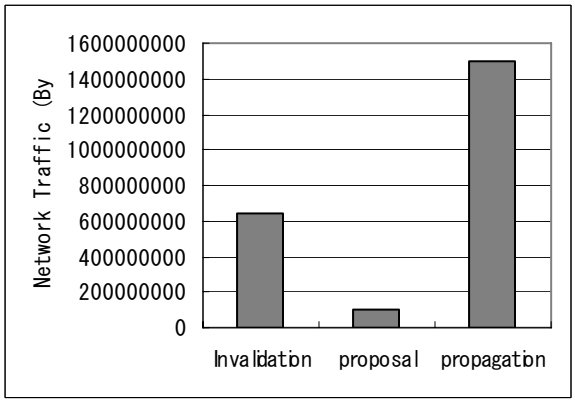


Fig. 2: Comparison of Network Traffic with Different Replication Algorithms.

We continue to test the algorithm performance when the web access parameters are changed. Several researchers have observed that the distribution of web request from a fixed group of users follows a Zipf distribution. Besides, the value of $\alpha$ , 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 shown in Fig.4 and Fig.5. We firstly tested *Old Hit* with respect to the varied parameter $\alpha$ . Figure 3 shows that the proposed algorithm obtains stable performance when the Zipf parameter is changed from 0.6 to 0.85. It always outperforms the conventional ones (*Invalidation* or *Propagation*).

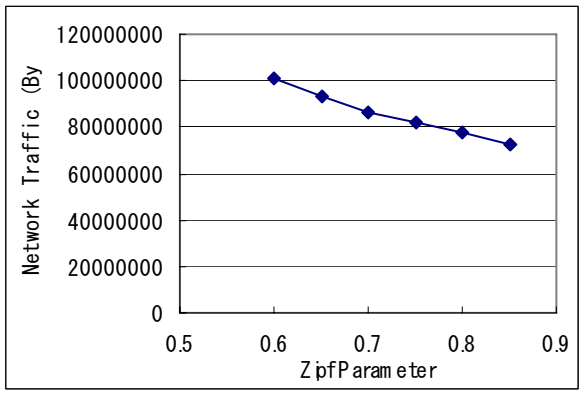


Fig. 3: Network Traffic under Different Zipf Parameters

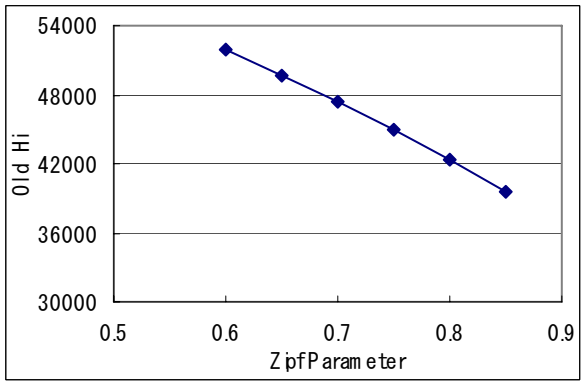


Fig. 4: Old Hit under Different Zipf Parameters

Note that the performance of our proposal gets better with the Zip'f parameter increased. When the Zip'f parameter is increased, the popular objects are requested more often compared with the situation under the low Zip'f parameter. It becomes easy for our algorithm to manage the client's always-requested objects. As our proposal is intended to keep these objects' consistencies well, it results in the higher performance.

We then simulate network traffic under the different Zipf parameters are carried out. Similar results are shown in Fig. 45, where the proposed one reduced network traffic most. Since our proposed algorithm is designed to reduce the user delay, the client can get the latest version of the request content from the nearer surrogate, so network traffic can also be reduced at the same time. We conclude that the proposal has better performance and is robust in varying situations when the web access pattern is changed.

## 5. Conclusions

This paper 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.

As for 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", Journal of Parallel and Distributed Computing**,** Volume 63 , Issue 10,Oct,2003

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

[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] 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.