RESP: SHORTEST-PATH-BASED CACHE REPLACEMENT IN A TRANSCODING PROXY

Hao-Ping Hung and Ming-Syan Chen

Graduate Institute of Communication Engineering
National Taiwan University
E-mail: mschen@cc.ee.ntu.edu.tw, hphung@arbor.ee.ntu.edu.tw

ABSTRACT

In this paper, we discuss the cache replacement policy in a multimedia transcoding proxy. Unlike the cache replacement for conventional web objects, to replace some elements with others in the cache of a transcoding proxy, we should further consider the inter-relationship among the cached items. To maintain the inter-relationship and to perform cache replacement, we propose in this paper the RESP framework (standing for Replacement with Shortest Path). The RESP framework contains two primary components, i.e., procedure MASP (standing for Minimum Aggregate Cost with Shortest Path) and algorithm EBR (standing for Exchange-Based Replacement). Procedure MASP maintains the inter-relationship using a shortest path table, whereas algorithm EBR performs cache replacement according to an exchanging strategy. The experimental results show that the RESP framework can approximate the optimal cache replacement with much lower execution time for processing user queries.

1. INTRODUCTION

The technology advance in network has accelerated the development of multimedia applications over the wired and wireless communication. To alleviate network congestion and to reduce latency and workload on multimedia servers, the concept of multimedia proxy has been proposed to cache popular contents. Caching the data objects can relieve the bandwidth demand on the external network, and reduce the average time to load a remote data object to local side. Since the effectiveness of a proxy server depends largely on cache replacement policy, various approaches are proposed in recent years. Subject to the application context, the cache replacement policy may differ from one situation to another, such as caching schemes in a transcoding proxy [1][3][4], multiserver cache replacement [7], and cache replacement for wireless data access [5][8].

In this paper, we focus on the cache replacement policy in a transcoding proxy. A transcoding proxy is a proxy capable of converting a multimedia object from one form to another, which trades object fidelity for size. Different from the caching strategy for conventional web objects, to replace some elements with others in the cache of a transcoding proxy, we should further consider the inter-relationship among the cached items. i.e., a multimedia object in a cache may be transcodable to several versions so as to satisfy different user requirements. In [1], the inter-relationship among the versions can be formulated as a *weighted transcoding graph*. Given some versions cached, a subgraph regarded as *minimum cost transcoding graph* is generated via procedure MATC (standing for Minimum Aggregate Transcoding Cost). To perform cache replacement, the authors in [1] also propose algorithm AE (standing for Aggregate Effect). In [3], the authors formulate the caching strategy in a transcoding proxy as the *0-1 knapsack problem* and propose algorithm MPR (standing for Maximum Profit Replacement) to achieve optimal solution. It is noted that algorithm MPR, which is based on the concept of *dynamic programming*, requires high overhead in memory and computing complexity, whereas the simple strategy adopted by algorithm AE may waste some cache space.

To remedy the drawbacks of prior works, we propose in this paper the RESP framework (standing for REplacement with Shortest Path) to maintain the transcoding graph and to perform cache replacement in a transcoding proxy. The RESP framework contains two primary components, i.e., procedure MASP (standing for Minimum Aggregate cost with Shortest Path) and algorithm EBR (standing for Exchange-Based Replacement). To maintain the transcoding graph, procedure MASP uses a table to record the shortest path information and determines the transcoding sources according to the table. To perform cache replacement, algorithm EBR uses a more profitable *caching candidate* to exchange less profitable elements in the cache so as to maximize the profit of cached elements. The experimental results show that the proposed RESP framework outperforms algorithm AE in *cache hit ratio*. Under many circumstances, the RESP framework can approximate the optimal solution very effectively. Moreover, the RESP framework costs much less computing complexity in processing user queries. The experimental study will justify the practicability of the RESP framework over the prior research in cache replacement of the transcoding proxy.

This paper is organized as follows. Notation and definitions are given in Section 2. In Section 3, we describe the RESP framework. The experimental results are shown in

Symbol	Description
G_i	the weighted transcode graph of object i
$V[G_i]$	the set of all vertices in the G_i
$E[G_i]$	the set of all edges in the G_i
$o_{i,j}$	version j of object i
$r_{i,j}$	the mean reference rate of $o_{i,j}$
d_i	the delay of fetching object i from server to proxy
$s_{i,j}$	the size of $o_{i,j}$

Table 1. Description of the symbols

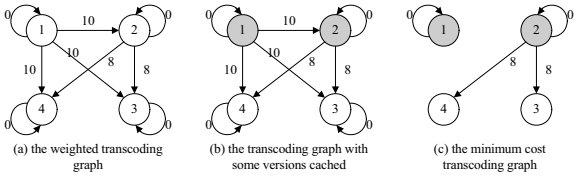


Fig. 1. Illustration of minimum cost transcoding graph

Section 4. Finally, this paper concludes in Section 5.

2. NOTATION AND DEFINITIONS

The symbols used throughout this paper are listed in Table 1. Without loss of generality, we adopt the same analytical model used in [1][3]. Assume that each object can be represented in n versions. The original version of object i is denoted as $o_{i,1}$, whereas the least detailed version which cannot be transcoded any more is denoted as $o_{i,n}$.

Definition 1: The $weighted\ transcoding\ graph$, G_i, is a directed graph with weight function ω_i. G_i depicts the transcoding relationship among transcodable versions of object i. Each vertex $v \in V[G_i]$ represents a transcodable version of object i. Version u of object i, i.e., $o_{i,u}$ is transcodable to version v, i.e., $o_{i,v}$ iff there is a directed edge $(u, v) \in E[G_i]$. The transcoding cost from version u to v is given by $\omega_i(u, v)$, regarded as the weight of the edge from u to v. Figure 1(a) illustrates an example of a weighted transcoding graph.

Definition 2: $PF(o_{i,j})$ is defined as the *singular profit* of caching $o_{i,j}$ while no other version of object i is cached. $PF(o_{i,j}) = \sum_{(j,x)\in E[G_i]} r_{i,x} * (d_i + \omega_i(1, x) - \omega_i(j, x))$.

Definition 3: The $minimum\ cost\ transcoding\ graph$, G'_i, is a subgraph in which the aggregate transcoding cost is minimized when the proxy caches multiple versions of an object simultaneously. For example, considering that $o_{i,1}$ and $o_{i,2}$ are cached, as shown in Figure 1(b), we can obtain the *minimum cost transcoding graph* (i.e., Figure 1(c)) by retaining the edges with the *cheapest* weight from the cached versions to all the other versions.

Definition 4: $PF(o_{i,j1}, o_{i,j2}, ..., o_{i,jk})$ is defined as the *aggregate profit* of caching multiple objects $o_{i,j1}, o_{i,j2}, ..., o_{i,jk}$ simultaneously.

$$PF(o_{i,j1}, o_{i,j2}, ..., o_{i,jk}) = \sum_{v\in V[G'_i]} \sum_{(v,x)\in E[G'_i]} r_{i,x} * (d_i + \omega_i(1, x) - \omega_i(v, x)).$$

Definition 5: $PF(o_{i,j}|o_{i,j1}, o_{i,j2}, ..., o_{i,jk})$ is defined as the *marginal profit* of caching $o_{i,j}$, given that $o_{i,j1}, o_{i,j2}, ..., o_{i,jk}$ are already cached where $j \neq j1, j2, ..., jk$.

$$PF(o_{i,j}|o_{i,j1,...}, o_{i,jk}) = PF(o_{i,j}, o_{i,j1,...}, o_{i,jk}) - PF(o_{i,j1,...}, o_{i,jk}). \quad (1)$$

When some versions already exist in the proxy. If a new version is cached, the amount of additional profit can be derived according to Eq. (1).

Given the database D which represents the collection of all possible queried data objects and the corresponding versions, the element $o_{i,j}$ contains two attributes: *profit* $p_{i,j}$ and item size $s_{i,j}$. Note that $p_{i,j} = PF(o_{i,j})$ if no other version of object i is cached, and $p_{i,j} = PF(o_{i,j}|o_{i,j1,...}, o_{i,jk})$ if there are some other versions $o_{i,j1,...}, o_{i,jk}$ cached.

Definition 6: The *generalized profit*, $g_{i,j}$, of the element $o_{i,j}$ is defined as $g_{i,j} = p_{i,j}/s_{i,j}$.

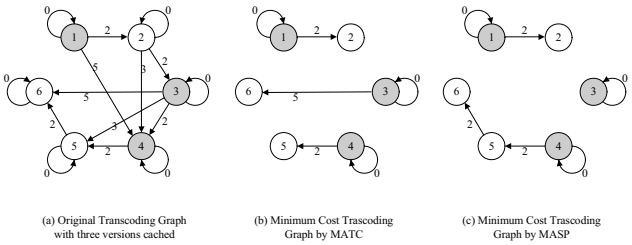
3. DESIGN OF THE RESP FRAMEWORK

3.1. Overview

The RESP framework (standing for REplacement with Shortest Path) will be triggered when a transcoding proxy needs to perform cache replacement. Two primary components are contained, i.e., procedure MASP (standing for Minimum Aggregate cost with Shortest Path) and algorithm EBR (standing for Exchange-Based Replacement). A user query for a specific element identified by the object number and version number corresponds to a *caching candidate*. With the caching candidate and the current cache state as the input, algorithm EBR will determine which elements in the cache should be replaced with the caching candidate. On the other hand, during the execution of algorithm EBR, procedure MASP will update the minimum cost transcoding graph dynamically so as to maintain the inter-relationship among the cached elements.

3.2. Generating Transcoding Graph with Shortest Path

Procedure MASP generates the minimum cost transcoding graph for cache replacement. Unlike the conventional approach MATC [1] (standing for Minimum Aggregate Transcoding Cost) which only performs well on simple transcoding graphs, procedure MASP is capable of maintaining the complex inter-relationship very effectively. The algorithmic form of procedure MASP is outlined as follows.



(a) Original Transcoding Graph with three versions cached

(b) Minimum Cost Trascoding Graph by MATC

(c) Minimum Cost Trascoding Graph by MASP

Fig. 2. Comparison between MASP and MATC

Procedure MASP
Input: weighted transcoding graph G,
the wet of cached versions C
Output: minimum cost transcoding graph G'
01. Create a table T with $|V[G]| \times |V[G]|$ entries
02. Set the value of $T(i,j)$ to be ∞ if $i > j$
03. **for** each vertex $v(i)$
04. **if** $v(i) \in C$
05. **for** each vertex $v(j)$, $j > i$
06. Set $T(i,j)$ to be the length of the shortest path from $v(i)$ to $v(j)$
07. **else**
08. Set $T(i,j)$ to be ∞
09. Create an array Src with $|V[G]|$ elements.
10. **for** each column j in T
11. $Src[j] = \arg\min\{T(i,j)|$ for each $i \leq j\}$
12. Generate G' according to Src
13. **return** G'

The comparison between the conventional MATC procedure and the proposed MASP procedure is illustrated in Figure 2. Consider the weighted transcoding graph G shown in Figure 2(a). Suppose that three versions, $v(1)$, $v(3)$, and $v(4)$ are cached. In procedure MATC, each vertex is only aware of its adjacent nodes. Therefore, the minimum cost weighted transcoding graph generated by MATC can be shown in Figure 2(b). On the other hand, in procedure MASP, the vertex with the minimum shortest path will be selected as the transcoding source, whether it is adjacent or not. The result of procedure is shown in Figure 2(c). Compare Figure 2(b) with Figure 2(c), we can find that version 6 should be transcoded from version 4 instead of version 3.

3.3. Exchange-Based Cache Replacement Algorithm

As described in [3], the problem of cache replacement in the transcoding proxy can be formulated as a *0-1 knapsack problem*. In [3], although algorithm MPR (standing for Maximum Profit Replacement) can achieve an optimal solution, it requires high memory and CPU overhead. On the other hand, algorithm AE (standing for Aggregate Effect) proposed in [1] can be executed very efficiently. However, the simple strategy adopted by AE (i.e., caching the requested element and its original version simultaneously and evicting the elements with smaller *generalized profit*) may result in the waste of cache space. To remedy the drawbacks of AE and MPR, we propose algorithm EBR, which can be executed efficiently to approximate the optimal cache replacement.

To design algorithm EBR, we are inspired by the concept proposed in [6], where the optimal cache replacement for web objects is approximated. Compared to [6], we further take the characteristics of the transcoding proxy system into consideration. The execution of algorithm EBR is described as follows. We first regard the cache as a $heap$ H, in which the elements are sorted in ascending order according to the *generalized profit*. Next, given the caching candidate element $o_{i,j}$, if the elements before the running pointer pos have larger aggregate size and smaller aggregate profit, algorithm EBR will replace these elements with $o_{i,j}$. Note that algorithm EBR is only executed when the cache has insufficient space to retain $o_{i,j}$. Otherwise, the transcoding proxy will cache $o_{i,j}$ without removing any elements. Moreover, since there is inter-relationship among the elements in the transcoding proxy, procedure MASP is employed to dynamically update the profit information in H. Unlike algorithm AE [1] which may waste some cache space, algorithm EBR utilizes the cache space very thoroughly. The insight of algorithm EBR is that we use a more profitable element to *exchange* the less profitable elements in the cache and guarantee that the total amount of profit of the cached elements is increased monotonically. Therefore, algorithm EBR can effectively maximize the profit of cached elements with much lower computation cost compared to algorithm MPR [3].

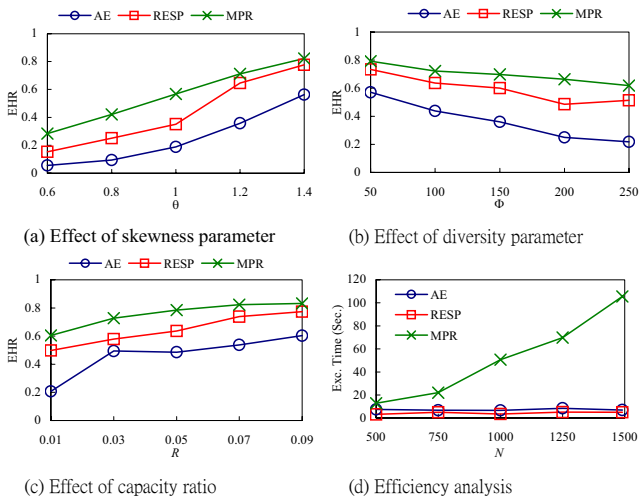
Parameters	Values
Number of objects (N)	1500
Number of versions	7
Skewness parameter of (θ)	1.2
Diversity parameter (Φ)	200
Cache capacity ratio (R)	0.05

Table 2. Parameters during the simulation

4. EXPERIMENTAL STUDY

4.1. Simulating Environment

The simulation model is designed to reflect the system environment of the transcoding proxy in Section 2. Table 2 summaries the settings of some primary simulation parameters. In the client model, user devices can be partitioned into seven classes. That is, each object can be transcoded to seven different versions by the transcoding proxy to satisfy the users' requirements. We generate the weighted transcoding graph of each object by removing several edges randomly from a complete graph. As for the transcoding proxy model, we assume that there are 1500 different objects, and choose the cache capacity to be $R*(\sum$object size$)$, where R is regarded as the *capacity ratio*. The reference rate of each object is generated by Zipf distribution $r_i = (\frac{1}{i})^\theta \sum_{j=1}^{N}(\frac{1}{j})$, where θ is viewed as a *skewness parameter*. The size of each ob-



(a) Effect of skewness parameter (b) Effect of diversity parameter

(c) Effect of capacity ratio (d) Efficiency analysis

Fig. 3. Performance evaluation of the RESP framework

ject is represented by the uniform distribution between $(0, \Phi]$ units, where we define Φ as the *diversity parameter*. As for the setting of other minor parameters, interested readers are referred to [1].

4.2. Performance Evaluation

In Figure 3, the performance of the RESP framework is compared to the algorithm MPR [3] and algorithm AE [1]. It is noted that algorithm MPR performs optimal cache replacement whereas algorithm AE can be regarded as a greedy algorithm. From Figure 3(a) to Figure 3(c), we evaluate the effectiveness of the RESP framework by varying several parameters such as *skewness parameter*, *diversity parameter* and *capacity ratio*. Unlike the various metrics used in [1], we choose the tightest one, the *exact hit ratio*, denoted by EHR, to verify the performance. The *exact hit ratio* is defined as the fraction of requests which are satisfied by the exact versions of the objects cached. This metric is also motivated by the fact that we usually intend to provide an exact version to users (rather than an overqualified one) for effective bandwidth use. In Figure 3(d), we discuss the efficiency of the related algorithms. To examine the efficiency of each algorithm, we measure the execution time for simulators to process 100000 client requests. Since we implement these approaches by Java language and execute the programs on the same system platform, the execution time will reflect the efficiency.

According to the experimental results, we observe that although algorithm MPR has the most outstanding performances in EHR, it requires more execution time and memory space to process user queries. When the number of the objects increases, algorithm MPR will suffer from the efficiency issues. On the other hand, algorithm AE is very scalable and insensitive to the variation of the efficiency parameters. However, it only results in fair quality of the effectiveness. As for the proposed RESP framework, it is as scalable as algorithm AE. Moreover, from the viewpoints of effectiveness, RESP outperforms AE prominently. Under certain circumstances such as high skewness, low diversity and low cache capacity, the quality of the RESP framework is very close to that of algorithm MPR. Therefore, compared to AE and MPR, the proposed RESP framework will be suitable and practical for a transcoding proxy to perform the cache replacement.

5. CONCLUSION

In this paper, we propose the RESP framework to perform cache replacement in a multimedia transcoding proxy. In RESP, two primary components, i.e., procedure MASP and algorithm EBR, are designed to maintain the inter-relationship and to perform cache replacement, respectively. The experimental results show that the RESP framework is effective in approximating the optimal solution with much lower complexity. Therefore, compared to the algorithms proposed in prior works, the proposed RESP framework will be suitable and practical for a transcoding proxy to perform the cache replacement.

Acknowledgements

The work was supported in part by the National Science Council of Taiwan, R.O.C., under Contracts NSC93-2752-E-002-006-PAE.

6. REFERENCES

[1] C.-Y. Chang and M.-S. Chen. On Exploring Aggregate Effect for Efficient Cache Replacement in Transcoding Proxies. *IEEE Transaction on Parallel and Distributed Systems*, 14(6), 2003.

[2] E. W. Dijkstra. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[3] H.-P. Hung and M.-S. Chen. Maximizing the profit for Cache Replacement in a Transcoding Proxy. In *Proceedings of IEEE ICME*, 2005.

[4] K. Li, K. Tajima, and H. Shen. Cache Replacement for Transcoding Proxy Caching. In *Proceedings of WI'05*, 2005.

[5] J. Xu, Q. Hu, W.-C. Lee, and D. L. Lee. Performance Evaluation of an Optimal Cache Replacement Policy for Wireless Data Dissemination. *IEEE TKDE*, 16(1), 2004.

[6] K. Yeung and K. Ng. An Optimal Cache Replacement Algorithm for Internet Systems. In *22nd Annual Conference on Local Computer Networks*, 1997.

[7] Q. Zhang, Z. Xiang, W. Zhu, and L. Gao. Cost-based Cache Replacement and Server Selection For Multimedia Proxy Across Wireless Internet. *IEEE Tran. Multimedia*, 6(4), 2004.

[8] B. Zheng, J. Xu, and D. Lee. Cache Invalidation and Replacement Strategies for Location-Dependent Data in Mobile Environments. *IEEE Transaction on Computers*, 51(10), 2002.