A Segment-Based Cache and Replacement Policy for Multimedia Streaming Objects

Tz-Heng Hsu[1] Yueh-Heng Li[2]
[1] Department of Computer Science and Information Engineering
Southern Taiwan University
E-mail:hsuth@mail.stut.edu.tw

摘要

最近幾年，由於行動裝置的可攜性與普及化，透過行動裝置來存取網際網路內容成為普遍現象。在異質性網路環境下，對於不同頻寬的網路環境及各種行動裝置，數位內容服務提供者需要提供不同版本的多媒體串流視訊給行動裝置客戶端做選擇。

在本篇論文裡，我們提出一種多媒體串流物件轉碼代理伺服器快取(transcoding proxy cache)取代策略。傳統的快取技術並不適用於多媒體串流代理伺服器。因此，本篇論文提出靜態權重值轉碼圖形、動態關聯樹方式，設計一個新的代理伺服器系統。在快取管理部分，當行動裝置客戶端要求影片，可改由轉碼(transcoding)後版本進行傳送，以減少傳送時間。本文就所提出的演算法和傳統 LRU、LFU 快取演算法，針對 Hit Ratio、Byte Hit Ratio、Average transcoding delay 等三部分進行模擬，結果顯示本論文的演算法明顯優於傳統 LRU、LFU 快取演算法。

關鍵詞:行動裝置、轉碼代理伺服器、快取、關聯樹

Abstract

In recent years, due to the portability and popularity of mobile devices causes access the Internet content become a common phenomenon. To support different bandwidth network environment and mobile devices in the heterogeneity network environment, digital content providers have to provide different versions of the multimedia streaming video for client to choose.

In this paper, we propose a caching replace strategy of multimedia streaming objects transcoding proxy cache. Traditional caching technology can not applicable to multimedia streaming proxy. Therefore, we proposed a static weight transcoding graph and dynamic relation tree to design a new proxy system. In the cache management, we can reduce transmission time by transcoding the original content before delivering it while the mobile device's client required. In this paper, our algorithm is compared with LRU and LFU cache algorithm in three part, "Hit Ratio"、"Byte Hit Ratio" and "Average Transcoding Delay". Experimental results show proposed algorithm outperforms to traditional LRU, LFU cache algorithm.

Keywords: Mobile Device 、 Transcoding Proxy、Cache、Relation Tree

1、前言

在現有的網際網路環境中，存在多元化的數位內容，許多影片可從網路下載觀看或線上收看，加上新興行動裝置應用，像是:攜帶式筆記型電腦、個人數位助理(Personal Digital Assistant)…等，在最近幾年快速成長，這些行動裝置客戶端可隨時隨地對網路資訊存取和要求。多媒體代理伺服器可減少網路上影片傳送時間和降低網路擁塞狀態。在行動裝置異質網路環境下，除了依照適合的頻寬來提供合適的影片品質外，影片轉碼代理伺服器(transcoding proxy)[3][4][7]設計也是一大挑戰。

轉碼(transcoding)的定義是將多媒體物件從一個物件轉換成另一個物件。以視訊串流物件為例，依照影片的 frame 特性，可從高解析度的影片轉換成低解析度的影片，或是以不同編碼格式進行轉換，如 AVI 格式轉換成 WMV 格式，如此可有效減少資料量大小。

網路串流視訊播放常會面臨一些問題，像是從伺服器端到代理伺服器之間頻寬不足、轉碼造成的延遲、同一時間客戶端到代理伺服器之間的頻寬限制…等，為了解決這些問題，影片轉碼代理伺服器將影片快取到本身的儲存空間裡，但代理伺服器終究仍有儲存上限，無法存入所有使用者要求的全部影片，所以依照

大多數使用者的存取特性來決定如何將重要影片片段放入快取裡，或將影片轉碼成較低品質影片，以減少網路傳輸時間成為快取取代策略研究上的重要課題。

現存的快取取代演算法[5][6][13]，無法應用在轉碼代理伺服器快取演算機制上，除了受到環境因素，造成轉碼延遲現象之外，不同檔案大小、參考次數、版本、片段也都會影響快取取代演算法效能。本篇論文考慮使用者網路頻寬、參考次數、檔案大小…等等因素，建構靜態權重值轉碼圖形、動態關聯樹，設計影片加入或刪除快取空間的管理方式，根據所提出的公式決定每個影片成本值，選擇傳輸成本較高的影片存入快取裡，再與現有LRU、LFU演算法進行Hit ratio、Byte Hit Ratio、Average transcoding delay等三部分的效能比較。

本篇論文的其他部分架構為:第二部份介紹目前相關研究，第三部份介紹本論文所提出之基於片段式多媒體串流物件轉碼代理伺服器取代策略演算法，第四部份介紹實驗模擬，第五部份對本篇論文做總結。

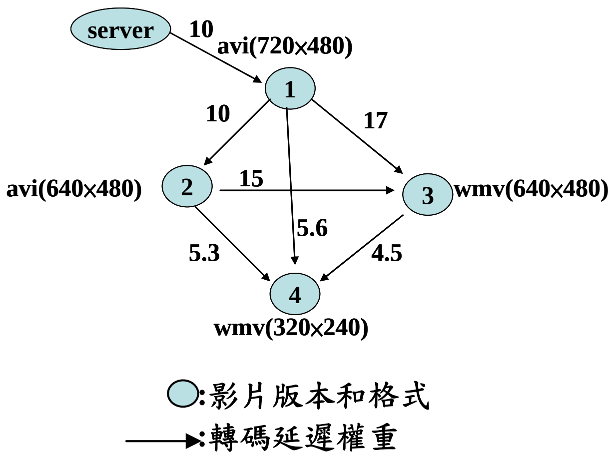
2、相關研究

為了達到高品質的影片串流服務(video streaming service)，代理伺服器的設計成為研究重點，論文[11][14][15]中提出類似web代理伺服器的架構，當使用者要求檔案時，首先會查詢代理伺服器是否擁有所要求的檔案，如果沒有，則代理伺服器與擁有檔案的伺服器進行下載，要求者再從代理伺服器下載，之後想要抓取相同檔案的使用者都可以直接從代理伺服器快取中取得到檔案。

將web代理伺服器的技術應用在多媒體串流物件上，必須考慮到多媒體物件轉碼代理伺服器快取能力。在有限快取容量中要存放哪些影片，轉碼時造成的延遲時間，不同影片大小、觀看次數及影片片段都必須要考慮到。在論文[3]中，將相關聯影片版本建立成權重值轉碼圖形，假定已將某些影片版本存放在快取中，建立成最小成本轉碼圖形，然後經由MATC (Minimum Aggregate Transcoding Cost)程序產生出最小成本值。在快取取代方式，作者根據generalized profit function提出AE (Aggregate Effect)演算法，利用鍊結串列此種資料結構進行快取管理，當快取空間足夠時將影片放入，計算每個物件在快取中成本值，當

表一、 符號說明

符號	描述
$o_{i,j}^{k}$	k 物件 i 片段 j 版本
$r_{i,j}^{k}$	k 物件 i 片段 j 版本的參考次數
$d_{i,j}$	從伺服器到 i 片段 j 版本的延遲時間
$s_{i,j}^{k}$	k 物件 i 片段 j 版本的資料量



圖一、權重值轉碼圖

空間不足時， 將成本值最小物件從快取中移除，再重新建立快取關聯圖形。在論文[5]中，作者提出MPR (Maximum Profit Replacement)演算法來達到最大快取效益。在論文[9]中，作者針對不同行動裝置和網路環境，找出使用者所需求的最適頻寬，以避免影片傳送和接收時受到影響，作者提出物件關聯圖(ORG)來管理影片版本之間的關係，建立快取物件關聯樹(CORT)做快取取代演算法的決策。

3、演算法

本節針對目前快取演算法的轉碼成本和使用者使用情形進行討論，我們根據假設圖形進行模擬和描述。

3.1 標記和描述

表一列出所使用的符號說明，假定每一個影片可切割成 i 個片段和轉碼成 j 個版本，k 物件的原始版本標示為 $o_{i,1}^{k}$；後面版本將無法轉碼成前面版本，例如:第 4 版本無法轉碼成第 3 版本。

定義 1:假定權重值轉碼圖，G_i，具有方向性權重值w_i，用來描述物件i 的轉碼關聯性；圖一為權重值轉碼圖，主要表示不同版本之間

的轉碼延遲時間,根據這個圖可以知道所要求的影片會得到的成本值大小。

假定有 4 個影片版本,第 1 個影片版本為 avi(720×480)格式、第 2 個影片版本為 avi(640×480)格式、第 3 個影片版本為 wmv(640×480)格式、第 4 個影片版本為 wmv(320×240) 格式，伺服器到第 1 個影片版本傳送時間為 10 秒，其它版本轉碼權重值以圖一表示，例如:描述節點 1 轉碼到節點 2 的延遲權重表示:

$e_g(V(avi(720\times480)),V(avi(640\times480)))=10$

定義 2:快取關聯樹,T_i,建立成樹狀結構，其樹根節點定義為N,它的子節點n_1、n_2、n_3…視為很多不同影片,每個影片分支出很多片段[8]**[9]**，每個片段再分支 4 個版本；圖二為快取關聯樹，主要表示影片放入到快取時，依照樹狀結構進行管理，我們假設每個影片分為 4 個片段i_1、i_2、i_3、i_4 ，每個片段可轉碼 4 個版本，建立樹狀快取架構，依照使用需求加入、刪除、檢索影片，隨時進行樹狀結構更新。

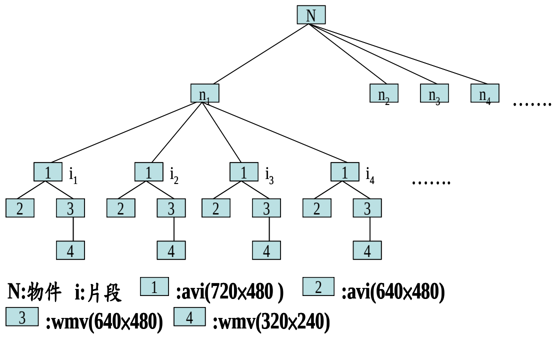
我們根據圖一定義影片轉碼延遲權重,當使用者要求影片時,計算該影片轉碼延遲的成本值,將所要求過的影片放入快取空間,根據圖二快取關聯樹,管理快取結構,決定影片的加入或刪除情形。

3.2 取代策略

當代理伺服器空間不足時,必須選擇物件來做為犧牲者，釋放空間。這部份我們提出快取取代策略，r 代表受歡迎的程度，數值越高優先權也越高。我們主要的考慮因素是影片轉碼延遲和受歡迎情況,計算出當影片在快取中成本為多少，如果當影片在快取內，使用者要求影片時，可將影片轉碼成適合客戶端的版本，就不需要從伺服器要求，只需要跟快取伺服器要求，便可減少延遲時間。所以我們定義公式，計算每個影片在快取的成本值:

定義 3:讓 $o_{i,j}^k$ 表示第 k 個影片的 i 片段的 j 版本， $r_{i,j}^k$ 表示參考 $o_{i,j}^k$ 次數， $d_{i,j}$ 表示從伺服器到代理伺服器延遲,我們定義快取利益公式:

$$CacheProfit(O_{i,j}^k) = r_{i,j}^k \times (d_{i,j} + e_g(n,P_n)) + \sum r_{i,j}^k \times (d_{i,j} + e_g(M,P_n) - e_g(M,n)) - \sum r_{i,j}^k \times (d_{i,j} + e_g(C_n,P_n) - e_g(C_n,n)) \qquad (1)$$



圖二、快取關聯樹

k表示物件，i表示片段，j表示版本，n表示要求節點，e_g表示每個版本的權重值，M表示連結到n節點的相關節點,P_n表示n節點的父親節點，C_n表示n節點的子節點。

公式(1)說明:快取中第 k 物件 i 片段 j 版本總成本利益=X-Y。X:其它物件透過 i 片段 j 版本所得到的利益=其它片段物件的利益- i 片段 j 版本物件的利益。Y:已存在快取中物件透過此物件所得到的利益。

在視訊串流媒體中,使用者通常由影片開頭片段開始播放,故影片開頭片段擁有較高的重要性，因此，我們針對快取檔案大小跟片段的重要性做處理，公式(2)如下:

$$PF(O_{i,j}^k) = \frac{CacheProfit(O_{i,j}^k)}{S_{i,j} \times i} \qquad (2)$$

我們根據提出的公式(1)和公式(2)，舉例計算，如圖三，假設目前有兩部影片，將影片分成 4 個片段 4 個版本,每個影片參考數值如表二，假設目前快取中有 $O_{1,1}^{k1}$ 、$O_{1,2}^{k1}$ 、$O_{4,1}^{k1}$ 、$O_{1,1}^{k2}$ ，分別計算快取物件成本值。

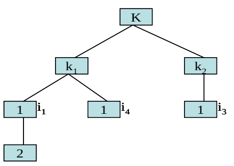
在公式(2)中，我們針對快取物件的大小跟片段的重要性做處理,這樣造成越後面存入的片段重要性越低,為了使片段重要性不要差距太大，新增加權重值 *w*，公式(3)如下:

$$PF(O_{i,j}^k) = \frac{CacheProfit(O_{i,j}^k)}{S_{i,j} \times (1 + w \times i)}, 0 < w < 1 \qquad (3)$$

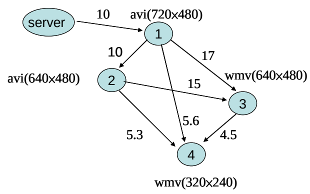
w 值介於 0~1 間,使後面片段成本不會差距太大。圖四為我們提出的快取取代演算法。

表二、影片參考值

物件k_1的i片段	d_i	r_{i1}	r_{i2}	r_{i3}	r_{i4}	s_{i1}	s_{i2}	s_{i3}	s_{i4}
1	10	5	1	1	1	11	10	8	4
2	10	4	3	2	1	11	8	5	3
3	10	2	1	1	0	10	7	5	4
4	10	3	1	1	1	12	11	8	4
物件k_2的i片段	d_i	r_{i1}	r_{i2}	r_{i3}	r_{i4}	s_{i1}	s_{i2}	s_{i3}	s_{i4}
1	10	5	3	2	1	14	12	8	6
2	10	2	2	1	1	10	9	6	3
3	10	2	2	1	1	10	9	7	4
4	10	2	1	1	1	12	10	5	4



目前快取關聯樹



權重值轉碼圖

$CacheProfit(O_{1,1}^{k1}) = 5\times(10)+1\times(10+10-10)$
$+1\times(10+17-17)+1\times(10+5.6-5.6)-1\times$
$(10+10-10)=70$

$PF(O_{1,1}^{k1}) = \frac{70}{11} = 6.36$

$CacheProfit(O_{1,2}^{k1}) = 1\times(20)+1\times(10+17-15)$
$+1\times(10+5.6-5.3)-0 = 42.3$

$PF(O_{1,2}^{k1}) = \frac{42.3}{10} = 4.23$

$CacheProfit(O_{4,1}^{k1}) = 3\times(10)+1\times(10+10-10)$
$+1\times(10+17-17)+1\times(10+5.6-5.6) = 60$

$PF(O_{4,1}^{k1}) = \frac{60}{12\times 4} = 1.25$

$CacheProfit(O_{1,1}^{k2}) = 5\times(10)+3\times(10+10-10)$
$+2\times(10+17-17)+1\times(10+5.6-5.6) = 110$

$PF(O_{1,1}^{k2}) = \frac{110}{14\times 1} = 7.86$

圖三、計算快取成本例子

4、模擬結果與討論

此部份我們將模擬及評估所提出之快取取代演算法的效能，在 4.1 描述模擬環境，4.2 顯示實驗結果。

4.1 模擬環境

我們設計出轉碼代理伺服器系統模擬環境，利用 GISMO(Generator of Internet Streaming Media Objects and workloads)[16]工具，模擬串流媒體的存取特性，包含物件受歡迎度、選擇需要影片時間、週期存取特性…等。

在本模擬實驗中多媒體串流物件受歡迎程度使用Zipf-like[10] [17]分佈來呈現，機率密度分佈函式為: $f(x) = 1/x^{\alpha}, x = 1,2,...,N$，N:假設有 N 部影片可供使用者選擇要求，$\alpha = 0.73$。

串流物件的大小根據 Lognormal 分佈亂數產生，在分佈中有 σ:ln(x)的偏差量、μ:ln(x)的平均值兩參數，首先從標準正規分布產生 x，然後傳回$e^{\mu+\sigma_X}$ 值到 Lognormal 分佈以表示物件大小，假設μ=10，σ=0.5。每個物件的 frames 大小根據 VBR(variable bit-rate)Marginal Distribution 產生。GISMO 可

```
CacheProfit constrained Replacement Algorithm (v)
V: requested version object
if( requested version object in cache)
{
  hit++;
  calculation  CacheProfit(v);
  push v into cache;
}else
{
 Miss++;
 while( not enough free space for v)
 { look for object u in the cache with the lowest
   CacheProfit(u);
   remove u;
 }
   calculation CacheProfit(v);
   push v into cache;
 }
```

圖四、CacheProfit 取代演算法

表三、依不同機率分佈模擬產生的模擬數據

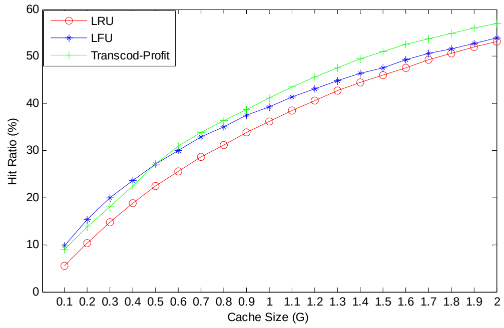
Component	Model	PDF	Parameters
Popularity	Zipf-like	$f(x)=1/x^{\alpha}, x=1,2,...,N$	$\alpha=0.73, N=250$
Temporal Correlation	Pareto	$f(x)=\alpha\frac{k^{\alpha}}{1-k^{\alpha}}x^{-\alpha-1}, k<x<1$	$\alpha=1, k=0.001$
Object size	Lognormal	$f(x)=\frac{e^{-(\ln x-\mu)^2/2\sigma^2}}{x\sigma\sqrt{2\pi}}, x>0$	$\mu=10, \sigma=0.5$
User Inter-activities	Pareto	$f(x)=\alpha\frac{k^{\alpha}}{1-k^{\alpha}}x^{-\alpha-1}, k<x<1$	$\alpha=1, k=0.001$
VBR Marginal Distribution	Lognormal	$f(x)=\frac{e^{-(\ln x-\mu)^2/2\sigma^2}}{x\sigma\sqrt{2\pi}}, 0<x<C$	$\mu=6, \sigma=0.3$

模擬使用者行為，包含使用者什麼時間觀看影片，從影片第幾秒開始看，總共看幾秒；本實驗中，基於表三，根據不同分佈來模擬產生數據。本實驗假設有 250 部影片、2000 個 session，一個 session 可包含多個要求，每個影片以一分鐘為一個片段，例如: OBJ#0 影片時間 12 分 57 秒共可分為 13 個片段。每個片段轉碼四個版本:avi(720×480)、avi(640×480)、wmv(640×480)、wmv(320×240)，快取容量最初 100M，每次再增加 100M，最大達到 2G，伺服器到代理伺服器傳送時間為 10 秒。

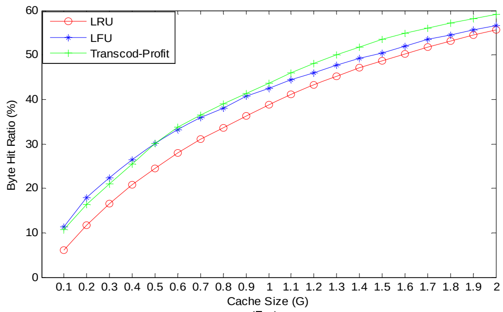
4.2 實驗結果

本節根據 Hit Ratio、Byte Hit Ratio、Average transcoding delay 等條件，計算代理伺服器快取效率。

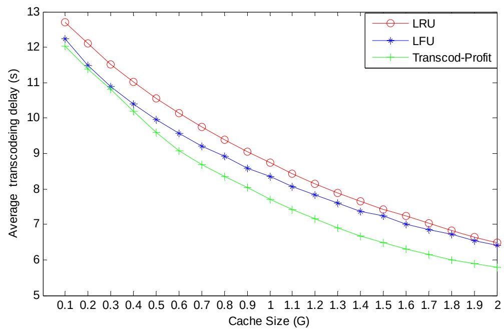
Hit Ratio:當使用者要求影片時，若快取內包含此使用者所要求之影片，我們稱之為 cache hit；反之，則稱為 cache miss。Hit rate 就



(a)

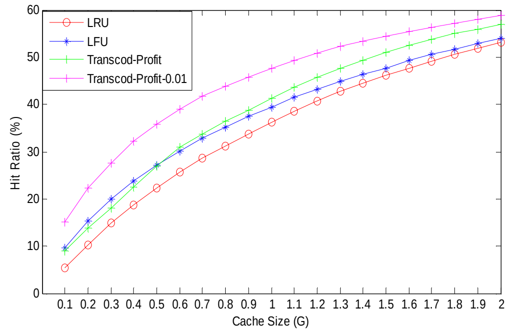


(b)

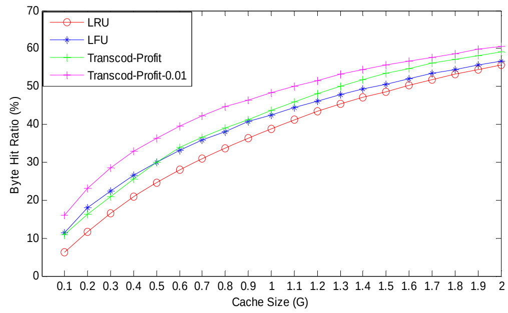


(c)

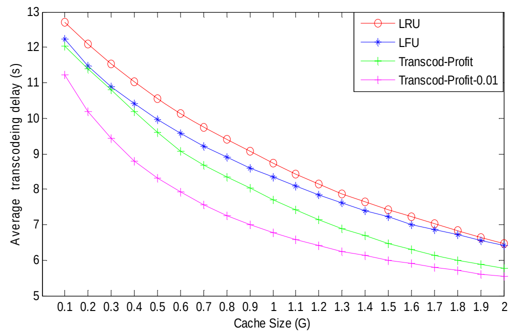
圖五、(a) Hit Ratio (b) Byte Hit Ratio (c) Average transcoding delay



(a)



(b)



(c)

圖六、 (a) Hit Ratio (b) Byte Hit Ratio (c) Average transcoding delay

是 cache hit 的發生百分比。Hit rate 高，表示 cache 能有效地降低使用者直接存取伺服器的次數。

Byte Hit Ratio[1][2]:使用者從快取內取得的檔案大小佔所有取得的檔案大小的比例即為byte hit rate。Byte hit rate 高，表示快取能有效地降低伺服器資料傳輸量。

Average transcoding delay:我們定義直接存取伺服器時間及存取經過轉碼後時間值稱為 transcoding delay，所有使用者要求行為的平均轉碼延遲時間，稱之 Average transcoding delay。

根據第三部份提出的演算法與 LRU 和 LFU[12]快取進行比較:LRU(Least Recently Used policy):是以影片上次的存取時間為根據，將最近沒有被存取的影片取代掉。LFU (Least Frequently Used):考慮影片的存取次數，將影片存取次數最低者淘汰取代，留在快取中的影片為最常被存取到的影片。比較後結果顯示在圖五，明顯的可以看出公式(2)提出的演算法整體會比 LRU 和 LFU 好。但在快取容量低於 500M 以下時 LFU 表現較為優異，主要是因為快取容量小，無法存放多個影片時 LFU 演算法，捨去目前最少存取次數的物件，而我們提出的演算法，則是捨去根據公式(2)所計算出的最小成本，所以形成 LFU 演算法的效率較佳。

接著我們改以公式(3)，進行實驗模擬，將 w 值假設為 0.01，每個片段重要性從原本假設為 1、2、3…等，改為 1.01、1.02、1.03…

等，使影片內片段成本差距不會很大。結果如圖六，整體效率提升 5%，延遲(delay)時間也減少 1 秒左右，主要因為後面片段在多人要求時能保留在快取以增加效率，不會因為片段成本差距太大而容易的被移除及取代。

5、結論

本篇論文主要提出多媒體串流物件轉碼代理伺服器快取取代策略，建構靜態權重值轉碼圖形、動態關聯樹，設計影片加入或刪除快取空間的管理方式。我們針對行動裝置網路環境下，使用者要求行為進行模擬，與傳統的 LRU、LFU 快取演算法進行模擬實驗，結果顯示所提出演算法效能比較好，代表我們的演算法取代策略可有效淘汰替換較少存取的影片。未來我們將針對行動裝置之間的資料傳輸協同合作方式做研究，以期減少轉碼所造成延遲的時間。

誌謝

本研究計劃執行所須之研究經費係由行政院國家科學委員會所提供。計劃編號:NSC 96-2221-E-218 -012。

參考文獻

[1]Arlitt M., Cherkasova L., Dilley J., Friedrich R., Jin T. (2000) “ Evaluating Content Management Techniques for Web Proxy Caches. ” *ACM SIGMETRICS Performance Evaluation Review,27(4)*:pp. 3-11.

[2]Cherkasova Ludmila (1998), “ Improving WWW Proxies Performance with Greedy -Dual-Size-Frequency Caching Policy. ” *HP Computer Systems Laboratory*.

[3]C.-Y. Chang and M.-S. Chen(2003) “ On Exploring Aggregate Effect for Efficient Cache Replacement in Transcoding Proxies. ” *IEEE Transaction on Parallel and Distributed Systems,14(6)*:pp. 611-624.

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

[5]Hao-Ping Hung, Ming-Syan Chen(2005) “ Maximizing the profit for cache replacement in a transcoding proxy. ” *ICME*:pp. 1162-1165.

[6]Hao-Ping Hung, Ming-Syan Chen(2006) “ RESP:Shortest-Path-Based Cache Replacement in a Transcoding Proxy. ” *ICME* :pp. 1145-1148.

[7]K. Li, K. Tajima, and H. Shen.(2005) “ Cache Replacement for Transcoding Proxy Caching. ” *In Proceedings of WI’05*:pp. 500-507.

[8]Kun-Lung Wu, Philip S. Yu, and Joel L. Wolf.(2001) “ Segment-based proxy caching of multimedia streams, ” *In Proceedings of the 10th International Conference World Wide Web*:pp. 36 - 44.

[9]Kuei-Chung Chang, Ren-Yo Wu, Tien-Fu Chen(2005) “ Efficient Segment-Based Video Transcoding Proxy for Mobile Multimedia Services. ” *ICME 2005* :pp. 755-758.

[10] L. Breslau, P. Cao, L. Fan, G. Phillips, S. Shenker(1999) “ Web caching and zipf-like distributions: Evidence and implications. ” *INFOCOM*: pp. 126–134.

[11] Li Xiao , Xiaodong Zhang , Zhichen Xu(2002) “ On reliable and scalable peer-to-peer Web document sharing. ” *In Proc. International, IPDPS 2002*:pp. 224 – 231.

[12] P. J. Denning. (1968) “ The working set model for program behavior. ” *Communications of the ACM 11(5)*:pp. 323–333.

[13] Q. Zhang , Z. Xiang , W. Zhu and L. Gao(2004) “ Cost-based Cache Replacement and Server Selection For Multimedia Proxy Across Wireless Internet. ” *IEEE Tran. Multimedia, 6(4)*:pp. 587-598.

[14] R. Floyd and B. Housel.(1998) “Mobile Web Access Using Network Web Express. ” *IEEE Personal Comm 5(5)*:pp. 47-52.

[15] R. Han , P. Bhagwat , R. LaMaire , T. Mummert, V. Perret, J. Rubas.(1998) “ Dynamic Adaption in an Image Transcoding Proxy for Mobile Web Browsing. ” *IEEE Personal Comm. 5(6)*:pp. 8-17.

[16] Shudong Jin, Azer Bestavros(2001) “ GISMO:A generator of internet streaming media objects and workloads. ” *SIGMETRICS Performance Evaluation Review 29(3)*:pp. 2-10.

[17] Zipf–Mandelbrot law. Available from: http://en.wikipedia.org/wiki/Zipf-Mandelbrot_law.