Optimal Model of Web Caching and Prefetching

Lei Shi 1, Yan Zhang 2, and Wei Lin 3
1 School of Information Engineering, Zhengzhou University, Zhengzhou, China
Email: shilei@zzu.edu.cn
2 Henan Polytechnic College, Zhengzhou, China
Email: iezhangy@yahoo.com.cn
3 School of Software, Zhengzhou University
weilin@zzu.edu.cn

Abstract—Caching and prefetching play important roles in improving the quality of data access. Replacement and prefetching algorithm optimization is the core of caching and prefetching model research. First, Independent Reference Model and Markov Reference Model are analyzed and compared in this paper. And so as the Markov-based Prefetching Model. Then, based on the measurement of Relative Popularity and Byte Cost, optimal Web caching and prefetching model named PR PPM are presented and analyzed. Simulations show that the performance of optimal model.

Index Terms—caching model; Web prefetchig model; Relative popularity; optimal model PR PPM

I. INTRODUCTION

Caching and prefetching as effective approaches to explosive growth in Network users and Web service, and has been widely used in Web Proxy, P2P, Grid Computing and Wireless network. Bringing some of more popular items closer to end-users can improve the network performance and, therefore, reduce the download latency and network congestion. Web caching and prefetching are based on temporal locality of user sequence.

Independent Reference Model (IRM) and Markov Reference Model (MRM) are mostly used for Web caching Model at present. While Markov-based Prefetching Model is mostly used for prefetching. The design of replacement policy is always based on characteristic of request sequences. Therefore, to modeling on user request sequences and Web objects properties exactly and simply is so important, and we hope to find optimal policies under these factors to be pursued in systematic manner. This paper firstly analyzes and compares Web caching and prefetching models that are used nowadays, and then based on the measurement of Relative Popularity and Byte Cost, it presents an optimal Web caching and prefetching model PR PPM that satisfy different performance metrics.

The rest of this paper is organized as follows. Section 2 discusses the related work. In section 3, optimal Web caching and prefetching model are presented and analyzed. Simulations results are also analyzed in section 3. Section 4 contains the summary and conclusions.

II. RELATED WORK

Numerous studies show that [1][2][5] there are some regulations in Web environment: Web traces exhibit excellent temporal locality and spatial locality; Web object size follows a heavy-tailed distribution; The popularities of Web documents usually follow generalized Zipf’s law distribution; Viewing the Web surfing as a random walk and the probability distribution of surfing depth follows a two-parameter inverse Gaussian distribution. The notations and their presentations used in this paper are illuminated in table 1.

A. Conventional cache model

Independent Reference Model (IRM): In the context of conventional caching techniques, the underlying working assumption is Independent Reference Model (IRM) [2][4]. Under IRM the miss rate (respectively, the hit rate) is minimized (respectively, maximized) by the policy A according to which a document is evicted from the cache if it has the smallest probability of occurrence (respectively, is the least popular) among the documents in the cache. IRM can be formalized as a multi-tuple, M=(B, V, A, R, C), thereinto:

1) if \( R_i \notin S_t \), \( V_t \subseteq S_t \); else \( V_t \subseteq \emptyset \).

<table>
<thead>
<tr>
<th>Notations</th>
<th>Presentations</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Objects available in Web server</td>
</tr>
<tr>
<td>R</td>
<td>User request sequences</td>
</tr>
<tr>
<td>( R_i )</td>
<td>The ( i^{th} ) request in cache</td>
</tr>
<tr>
<td>S</td>
<td>Objects in cache</td>
</tr>
<tr>
<td>( S_t )</td>
<td>Objects in cache before ( R_t ) arrived</td>
</tr>
<tr>
<td>X</td>
<td>Cache states volume</td>
</tr>
<tr>
<td>( X_t )</td>
<td>Cache state at time ( t )</td>
</tr>
<tr>
<td>V</td>
<td>Operation volume</td>
</tr>
<tr>
<td>( V_t )</td>
<td>Objects evicted from cache if ( R_t ) is not in cache</td>
</tr>
<tr>
<td>C</td>
<td>Restriction</td>
</tr>
<tr>
<td>A</td>
<td>Replacement algorithm</td>
</tr>
<tr>
<td>B</td>
<td>Cache size</td>
</tr>
<tr>
<td>( s_i )</td>
<td>The size of ( i )</td>
</tr>
<tr>
<td>( c_i )</td>
<td>Replaced cost if ( i ) is not in cache</td>
</tr>
<tr>
<td>( l_i )</td>
<td>Download latency of ( i )</td>
</tr>
<tr>
<td>P_i</td>
<td>Popularity of ( i )</td>
</tr>
<tr>
<td>( \text{RP}_i )</td>
<td>Relatively popularity of ( i )</td>
</tr>
<tr>
<td>( U_i )</td>
<td>Byte cost of ( i )</td>
</tr>
</tbody>
</table>

Supported by the National Natural Science Foundation of China under Grant No.60472044
(2) R is assumed to form i.i.d.
(3) A is mandatory.
(4) restriction C: ∀ i and j, s_i = s_j, c_i = c_j.
IRM doesn’t consider the statistical information contained in the stream of requests and fails to capture temporary locality. However, it is simple enough to find effective policy.

Markov Reference Model (MRM): Another reference model often encountered in caching applications is the Markov Reference Model (MRM) [3][4] according to which requests are modeled by a stationary and ergodic Markov chain. MRM can also be formalized as a multi-tuple, M= (B, V, A, R, C), thereinto:

(1) if R_t ∉ S_t, V_t ∈ S_t; else V_t ∈ φ.
(2) R is prescribed by a Markov chain.
(3) Restriction C: Markov.
(4) The MRM specializes to the IRM.

The key property of MRM is Markov. MRM is good for independent distribution and request streams with less contact.

B. Existing prefetching model

A Markov model is a finite-state machine where the next state depends only on the current state. Associated with each arc of the finite-state machine network is the probability of making the given transition. When applied to the prediction of user accesses, each state represents the context of the user. This model can also be formalized as a multi-tuple, P= (X, P, O, C), thereinto:

(1) X denotes the states space.
(2) P is the transition probability from X_i to X_j.
(3) O is the length of context.
(4) Other restraints like threshold, pruning and so on.

The mostly used PPM Models are Standard PPM, LRS PPM Model and PB PPM Model. The comparisons of these typical PPM Models are given in table 2.

C. Web caching model

A total of N distinct cacheable objects {1, 2, ..., N} are available over all servers. For each t= 0, 1, 2, ..., n, the n-value r_vt represents the i^th request presented at the cache. The stream of successive requests arriving at the cache is then captured by the sequence of requests R= {R_t, t=0, 1, 2, ..., n}. The popularity of requests in the sequence {R_t, t=0, 1, 2, ..., n} is defined as the pmf P, P_i is the probability of making the given transition. When applied with each arc of the finite-state machine network is the next state depends only on the current state. Associated with the content of the cache evolves after the request R_t is handled, we have

\[ S_t = \begin{cases} S_t + R_t & \text{if } R_t \in S_t \\ S_t + R_t - V_t & \text{if } R_t \not\in S_t \end{cases} \]

where |S_t| denotes the cardinality of the set S_t and S_t + R_t - V_t is a subset of N obtained from S_t by adding R_t and removing V_t. The eviction action V_t at time t= 0, 1, 2, ..., n is dictated by a cache replacement policy. Variety of cache state is decided by:

(1) Cache states collection, X= {X_t, t=0, 1};
(2) User request sequences, R= {R_t, t=0, 1};
(3) Eviction actions produced by policy A, V= {V_t, t=0, 1}.

We select X_t as the pair (S_t, V_t) for time t, therefore we have X_t+1 = (S_t+1, V_t+1) at time t+1.

Web caching model can be defined as a multi-tuple, M= (B, V, A, R, C), thereinto:

(1) if R_t ∉ S_t, V_t ∈ S_t; else V_t ∈ φ;
(2) R forms Zipf-like distribution;
(3) A is randomized;
(4) Restriction C: whenever i≠j, s_i≠s_j, c_i≠c_j.

Different from conventional caching model, under Web caching model, the objects have non-uniform costs (as we assimilate cost to size and variable retrieval latency), there exist correlations in the request streams and request streams form Zipf-like distribution, and object placement and replacement are optional upon a cache-miss.

II. OPTIMAL WEB CACHING MODEL AND CORRESPONDING POLICIES

A. Modeling based on caching performance metrics

Perfect cache performance is the main motivation of replacement policies. When estimating performance, hit ratio, byte hit ratio and download latency are used commonly to measure performance of policies.

For the caching model above, we can estimate expected cost under policy A.

Let \( \delta_i = \begin{cases} 1, & \text{if } i \text{ is in cache}; \\ 0, & \text{else} \end{cases} \)

M. The first user request was arrived at time 0. Thus the cumulative expected cost over the horizon [0, T] becomes

\[ E_c(A) = E\left[ \sum_{t=0}^{T} (1 - \delta_t) \cdot c_t \right] \]

The expected average cost (over the infinite horizon) under policy A defined by

\[ \overline{E}_c(A) = \lim_{T \to \infty} \frac{1}{T+1} E\left[ \sum_{t=0}^{T} (1 - \delta_t) \cdot c_t \right] \]

A number of situations can be handled by adequately
specializing the cost-per-step $c_i$, i.e., if $c_i=1$, good hit rate can be achieved; on the other hand, if $c_i$ is taken to be the byte size $s_i$, then byte hit ratio is denoted. Therefore we have performance metrics as follows.

(1) Hit ratio (HR): Let $c_i=1$, $i=1,...,N$. Hit ratio under policy A is defined by

$$HR(A) = \lim_{T \to \infty} \frac{1}{1+E} \left( \sum_{i=1}^{N} \delta_{i} \right)$$

(2) Byte hit ratio (BHR): Let $c_i$ denote size of object $i$, $c_i = s_i$. Byte hit ratio under policy A can be defined by

$$BHR(A) = \lim_{T \to \infty} \frac{1}{1+E} \left( \sum_{i=1}^{N} \delta_{i} \cdot s_{i} \right)$$

(3) Download latency (LR): Another performance metric of great interest is the user-perceived download latency. Let $c_i$ denote the delay fetching document $i$ from Web server, is $l_i$, then download latency under policy A is defined by

$$LR(A) = \lim_{T \to \infty} \frac{1}{1+E} \left( \sum_{i=1}^{N} (1-\delta_{i}) \cdot l_{i} \right)$$

In actual Web environment, there is conflict among different performance metrics. For example, hit ratio emphasizes particularly on reducing respond time, whereas byte hit ratio pays more attention to bandwidth spending. Since objects in cache have variable sizes, keeping more documents with small size can improve hit ratio, however, preferable byte hit ratio is not always obtained. On the other hand, saving large size documents can improve byte hit ratio. Network users always prefer reducing bandwidth, thus maximizing byte hit ratio is more important for them. In a word, we hope to find optimal policies under these factors to be pursued in systematic manner.

B. Modeling based on prefetching performance metrics

This subsection surveys the web performance indexes appeared in the open literature focusing on prefetch aspects. To the better understanding of the meaning of those indexes, we classify them into three main categories, according to the system feature they evaluate: 1) prediction related indexes. 2) resource usage indexes. 3) end-to-end perceived latency indexes. Surveys show that, indexes between caching and prefetching have some relations like:

$$L_{prec} = L_{cache} - \sum_{i=1}^{N} \frac{p_{i}T}{a_{i}u_{i} + 1}$$

$$H_{prec} = H_{cache} + \sum_{i=1}^{N} \frac{p_{i}}{a_{i}u_{i} + 1}$$

$$B_{prec} = B_{cache} + \sum_{i=1}^{N} \frac{s_{i}}{u_{i} + 1}$$

Where $L_{cache}$, $H_{cache}$, $B_{cache}$ and $L_{pre}$, $H_{pre}$, $B_{pre}$ denote the latency, hit ratio and byte hit ratio of caching and prefetching, respectively. These formulae show that caching and prefetching have the common optimizing aim that the one with lower cost, smaller size and more accessing frequency.

C. Optimal model

Analysis in chapter 2 shows that, conventional caching model fails to capture temporary locality and statistical information contained in the stream of requests. To describe user request sequences more exactly and improve performance of Web cache and prefetching, Relative Popularity and Byte Cost are presented to optimize Web caching and prefetching model.

**Definition1:** Relative Popularity (RP): rate of popularity of each document and the highest popularity.

$$RP_i = \frac{P_i}{D}$$

$D$ is a parameter means the most popular Web objects. We can obtain its value through normalized computing.

**Definition2:** Byte Cost ($U$): cost of unit byte.

$$U_i = \frac{c_i}{s_i}$$

When cache is full, objects with low cost (i.e., less popular, low latency, large size) are evicted out from cache.

Let optimal cost at time $k$ be $c_k$. After $R_k$ arrived, the state of cache becomes $S_{k+1} = S_k - V_k + R_k$. Thus optimal cost at time $k+1$ is

$$c_{k+1} = c_k + \sum_{i \in V_k} RP_i \cdot U_i$$

Since we can’t control $c_k$, to optimize the cost, minimal $\sum_{i \in V_k} RP_i \cdot U_i$ should be selected. We can achieve a Web caching and prefetching optimal model called PR PPM based on $RP_i$ and $U_i$ as follows.

**Input:** $R = \{R_1, R_2, ..., R_n\}$

**Output:** Caching objects

**Method:**
Step1. if $R_t$ is in cache
Step2. for each $R_t$ in request sessions, create $T$ which contains URL and other accessing record
Step3. return $T$

**Input:** user sequence LF

**Output:** prediction model $T$

**Method:**
Step1. initialize, $T$=Null
Step2. for each $R_t$ in request sessions, create $T$ which contains URL and other accessing record
Step3. return $T$

D. Simulations

Request trace forms Zipf’s law, and we select parameter $a=0.75$ in this experiment. Simulation is based on actual log: USASK-HTTP[6]. Representative PPM models like Standard PPM, LPS PPM and PB PPM are selected to compare with PR PPM.
Consider hit ratio, byte hit ratio, precision and recall as indexes for performance evaluation given by:

\[ HR = \frac{H}{R} \]  \hspace{1cm} (13)

\[ BHR = \frac{H_b}{R_b} \]  \hspace{1cm} (14)

\[ \text{Precision} = \frac{p^+}{p} \]  \hspace{1cm} (15)

\[ \text{Recall} = \frac{p^-}{R} \]  \hspace{1cm} (16)

We define below some basic concepts that have been used in above formulae.

H: amount of requests hit.
R: amount of user requests.
H_b: amount of hit requests bytes.
R_b: amount of requests bytes.
p: amount of objects predicted by the prediction engine.
p+: amount of prefetched objects that are subsequently demanded by the user.
p-: amount of objects prefetched by the prefetching engine.

We define below some basic concepts that have been used in above formulae.

H: amount of requests hit.
R: amount of user requests.
H_b: amount of hit requests bytes.
R_b: amount of requests bytes.
p: amount of objects predicted by the prediction engine.
p+: amount of prefetched objects that are subsequently demanded by the user.
p-: amount of objects prefetched by the prefetching engine.

III. CONCLUSIONS

PR PPM is studied in this paper. By analyzing advantages and weaknesses of conventional caching model and prefetching model, this paper presents optimal Web caching and prefetching model named PR PPM based on the measurement of Relative Popularity and Byte Cost. PR PPM order Web objects by size, value and download latency in systematic manner. And also PR PPM has better hit ratio, byte hit ratio because it can reflect the transfer of users’ interests. Therefore, good hit ratio and byte hit ratio are achieved and total precision and recall is optimal at the same time.

REFERENCES


