Dynamic Average of Inter-reference Time as a Metric of Web Cache Replacement Policy

Agung Sediyono
Informatics Engineering Department
Universitas Trisakti, Jakarta, Indonesia
Tel: +62215663232 ext 178, Fax: +6221567001
agung@trisakti.ac.id

Abstract—Caching objects in Internet environment is aimed to reduce bandwidth consumption and increase the response time of system in term of user perception. When the size of web cache is limited, it needs to manage the objects in web cache so that the hit ratio and byte hit ratio are optimal. For solving this problem, researcher proposes many web cache replacement policies such as LRU, LFU, GDS, GDSF, GD*, and IRT. In evicting object from web cache, each policy uses recency, frequency, size, and frequency, or inter-reference time of object requested by user respectively. In web cache environment, the inter-reference time is rarely explored in previous research. Based on our research there is a correlation between average value of inter-reference time and the temporal locality, so it is reasonable to implement this value as a metric of web cache replacement policy. This research proposes to applied dynamic average of inter-reference time (DA-IRT) as a metric in evicting object from web cache. Based on the experiment using web trace log from three companies, it can be concluded that the performance of HR and BHR of DA-IRT is dependent on the behavior of the user requests.

Index Terms—inter-reference time, dynamic average of inter-reference time, web cache replacement

I. INTRODUCTION

Caching objects in Internet environment is aimed to reduce bandwidth consumption and increase the response time of system in term of user perception. The performance of web cache is measured using the hit ratio (HR) and the byte hit ratio (BHR). HR is calculated from how many user requests that can hit the cache divided by the total requests. Mean while, BHR is calculated from how many bytes that can be hit in the cache divided by total bytes requested by user. In a web cache with limited cache size, the HR and BHR can be optimized by using cache replacement algorithms. Based on the research conducted by Lindemann & Waldhorst [1], it can be concluded that there is no cache replacement algorithm that can outperform at all workloads. The performance of the cache replacement policy is dependent on the behavior or characteristic of the web cache workload. The research of workload of the web cache was conducted extensively [2][3][4]. Breaslu, et al. [2] concluded that the distribution of the web requests follow a Zipf-like distribution and this model can explain why the performance of the web cache is certain asymptotic properties. Then, Cohen&Kaplan [3] measured the regularity of the workload and use it to design the optimal cache replacement algorithm. Mean while, Benevenuto et al. [4] explores the impact of the first timer, included the one timer, on the performance of cache replacement schemes. From these studies, there are many properties of the workload such as the object size, the frequency of references, recency, one timer, first time, type of objects, and inter-reference time of object requests that can influence the performance of the web cache. Therefore, researcher proposes many web cache replacement policy such as Least Recency Used (LRU), Least Frequent Used (LFU), Greedy Dual Size (GDS), Greedy Dual Size and Frequency (GDSF), GD*, and IRT itself. The last property that is the inter-reference time of the object requested was discussed intensively in memory cache replacement [5],[6],[7] and outperforms the previous cache replacement schemes, but it was rarely discussed in web cache environment. Tanaka&Tatsukawa [8] uses the inter-reference interval called II-PO for manage the size of the web cache by using the modified 2Q, but they only use one trace log as a testbed.

Based on the IRT characteristic of eight web trace logs from three companies: GIA, Telkom, and Peti Kemas Co., it can be concluded that there is a correlation between average of IRT and the temporal popularity of the web object. Therefore instead of using IRT, this paper proposes to use average value of IRT. Because of the computation efficiency consideration, the average value of IRT for each cache object is approached using dynamic average of IRT. Therefore, this paper proposes to implement dynamic average of inter-reference time as a metric for web cache replacement policy. Based on the experiment result, it can be concluded that the performance of HR and BHR of DA-IRT is dependent on the behavior of the user requests.

The rest of this paper will be arranged as follows. Related work will be discussed in Section II. The DA-IRT is discussed in section III, and data preparation and simulation processes are presented in section IV. Experimental result and analysis is presented in section V, and then the conclusion and future work are presented.
II. RELATED WORK

Inter-reference time of the successive object requests was extensively discussed and implemented in memory cache replacement [5][6][7]. Phalke&Gopinath [5] explored the behavior of inter-reference gap (IRG) that is the time interval between successive references to the same address. They concluded that the IRG has, in general, a repetitive behavior. Therefore, they applied a $k$ order Markov chain to predict the next reference in the future. Based on the experiment, this method can improve the cache replacement until 37% over the Least Recently Usage (LRU).

Jiang&Song [6] introduce the LIRS cache replacement based on Inter-reference Recency (IRR) Set. IRR uses the number of references of the other objects that is in the inter-reference time of certain object. On the other hand, they use spatial locality instead of temporal locality. They argue that the age of the object in the cache can be measured by counting the number of references of the other object after the object measured is entered into the cache. LIRS uses two blocks of cache: LIR for low inter reference and HIR for high inter reference. By using this approach that is not depending on the detectable predefined regularities in the reference of the workloads, LIRS can improve the LRU performance.

Mean while, Takagi&Hiraki [7] argue that each memory address has own IRG distribution, so that they suggest to make individual probability distribution of each memory block and use the distribution to estimate the next reference in the future. This approach depends on the historical data so that it can introduce the complexity in both memory and computation.

Even though IRT has extensively discussed and implemented successfully in memory cache replacement, the research on the inter-reference time for web cache replacement was rarely conducted. Tanaka & Tatsukawa [8] adopted IRT to be a metric in web cache replacement algorithm. They modified the definition of IRT as an interval time between the time of purge object and time of the miss access of that object. For example, if an object $x$ is referenced at time $t_1$, $t_2$, and $t_3$, and the reference at $t_2$ is not in the cache, then the original inter-reference interval for object $x$ are $t_2-t_1$ and $t_3-t_2$. Instead of taking all inter-reference time, they take only $t_2-t_1$ as a metric called II-PO for cache replacement. A small II-PO implies that if the cache had additional it could have kept the object.

They implemented the II-PO metric in the modified 2Q (2Q-Opt). 2Q-Opt uses two caching areas, Q1 and Q2. Q1 is a FIFO queue that keeps objects which are referenced for the first time, and Q2 is a LRU queue that keeps objects whose references counts are more than one. Caching management is conducted by decreasing or increasing the length of Q1 or Q2 vice versa based on the II-PO value so that the total cache size is not change. The drawbacks of this approach are requiring unlimited space for recording purged objects and testing only in one trace log. Therefore, it stills not confident whether is not the result also valid for the other web trace logs. This question is reasonable because based on research conducted by Lindemann & Waldhorst [1], it can be concluded that there are no cache replacement algorithm that can fit at all situations. The performance of the cache replacement policy is dependent on the behavior or characteristic of the web cache workload, especially the composition of the object type in the web cache.

III. DYNAMIC AVERAGE OF IRT

In this section it will be presented why the dynamic average of IRT can be used as a metric in evicting object in the web cache and how this metric will be implemented so that the computation time will be efficient.

A. Rationale

Based on the characteristic of IRT average, it can be shown that there is a strong correlation between IRT average and the temporal locality (see Table 2). Therefore, it is reasonable to implement the average of IRT as a metric for web cache replacement policy. However, if this metric is implemented in web cache replacement policy, it has to be calculated from previous set of log trace. If this approach will be taken, it is not only takes time to compute the average of IRT but also

<table>
<thead>
<tr>
<th>Table 1. The properties of the web caches under investigation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GIA #1</strong></td>
</tr>
<tr>
<td># of Request</td>
</tr>
<tr>
<td># of Cachable Request</td>
</tr>
<tr>
<td>Request rate daily</td>
</tr>
<tr>
<td>% of Cachable Request</td>
</tr>
<tr>
<td>Total Size of Cachable Object (MB)</td>
</tr>
<tr>
<td>One Timer</td>
</tr>
<tr>
<td>% of One Timer</td>
</tr>
<tr>
<td># of Distinct Request</td>
</tr>
</tbody>
</table>

72
there is difficult to determine how long the previous log trace will be taken. Therefore, the IRT average will be approximated by the dynamic average of IRT (DA-IRT).

In this approach, the average of IRT is calculated on every object in web cache by dividing the cumulative of the IRT by the frequency of reference (see Equation 1)

\[
IRT_{ni} = \frac{IRT_{ni-1} + (t_{ni} - t_{ni-1})}{f_{ni}}
\]

where \(IRT_{ni}\) is a dynamic average of inter-reference time of the reference \(n\) at time \(i\), \(t_{ni} - t_{ni-1}\) is inter-reference time of reference \(n\) at time \(i\), \(f_{ni}\) is frequency of reference \(n\). Notable, for the first timer object, the average of IRT is assumed equal to the first reference time and placed into the web cache based on LRU policy among the first timer object.

Object with largest IRT average will be evicted first from the web cache. This approach assumes that object with smaller IRT average will be accessed sooner, so that it has to be kept in web cache in order to be hit.

### Table 2. The correlation between the average of IRT and temporal locality

<table>
<thead>
<tr>
<th>Web Cache</th>
<th>Correlation of Average of IRT</th>
<th>(a)</th>
<th>(b)</th>
<th>(r^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GIA#1</td>
<td>0.792747808</td>
<td>5.399</td>
<td>152.14</td>
<td>0.6284</td>
</tr>
<tr>
<td>GIA#2</td>
<td>0.612937861</td>
<td>12.624</td>
<td>290.18</td>
<td>0.3757</td>
</tr>
<tr>
<td>GIA#3</td>
<td>0.774937775</td>
<td>2.8128</td>
<td>65.35</td>
<td>0.6005</td>
</tr>
<tr>
<td>GIA#4</td>
<td>0.792337503</td>
<td>11.946</td>
<td>144.23</td>
<td>0.6278</td>
</tr>
<tr>
<td>Telkom#1</td>
<td>0.648806551</td>
<td>1.8602</td>
<td>261.43</td>
<td>0.4209</td>
</tr>
<tr>
<td>Telkom#2</td>
<td>0.546831085</td>
<td>1.9141</td>
<td>519.91</td>
<td>0.2990</td>
</tr>
<tr>
<td>Telkom#3</td>
<td>0.74491156</td>
<td>3.2217</td>
<td>100.42</td>
<td>0.5543</td>
</tr>
<tr>
<td>PetiKemas</td>
<td>0.731387095</td>
<td>15.49</td>
<td>1194.9</td>
<td>0.5349</td>
</tr>
</tbody>
</table>

### B. Implementation

The DA-IRT is implemented using linked list data structure. The data attribute saved in the web cache are object size, lastly referred time, cumulative IRT, and frequency of reference. The cumulative IRT and frequency of reference of the object will be updated if the object is referred. Every updating the average of IRT, the linked list has to be sorted descendently by the value of average IRT. The algorithms of the DA-IRT as follows:

**Input:** X the object requested by user

**Process:**

- If Object X is in Cache
  - Add cumulative IRT by current IRT
  - Increase the frequency by one
  - Calculate the DA-IRT
  - Sort descendently by DA-IRT value
  - Calculate HR, BHR
- Else
  - While there is no the space of cache for X
    - Evict the object with largest DA-IRT value
    - Add object X into the cache using LRU policy among the first timer

**Output:** HR, BHR

### IV. METHODOLOGY

The methodology used in this research is experimental-based methodology. The experiment is conducted by simulating the web cache replacement policy and using the web trace log as an input. This section describes and discusses about the evaluation criteria and the data preparation for the web cache simulation.

#### A. Evaluation Criteria

The criteria of evaluation is determined to assess the performance among web cache replacement policies. Based on the previous research, the criteria of evaluation for the web cache replacement are Hit Ratio (HR) and Byte Hit Ratio (BHR). The HR is ratio between the number of references and number of requests. Meanwhile BHR is ratio between the number of byte of the references and the number of byte of the requests. The formulation of HR and BHR as follows:

\[
HR = \frac{\sum \text{hit}}{\sum \text{request}}
\]

\[
BHR = \frac{\sum \text{byte_hit}}{\sum \text{byte_requested}}
\]

#### B. Data Preparation

This section discusses about data testbed for simulation beginning from the raw data, data processing, and data properties.

The raw data for the experiment are collected from three companies: Garuda Indonesia Airways (GIA), PT Telkom (Telkom), and PT Peti Kemas (PetiKemas). The GIA web caches have been collected as long as three weeks from November, 1st till 18th 2008, and the Telcom web caches have been collected for one week from November, 2nd till 8th 2008. Meanwhile, PetiKemas web cache have been collected for five weeks from June, 26th 2008 till July, 31st 2008.

Before the web caches workload is explored, the web caches are filtered so that only the cacheable object that will be explored. To filter the cacheable objects, this paper adopt the rule that was also used by [9]. The rule is the web request that contain the ‘?’ , ‘cgi’, or ‘cgi-bin’ will be discarded from the web cache log, and only those request with a cacheable response code, that is, 200 (OK), 203 (Partial), 206 (Partial Content), 300 (Multiple Choices), 301 (Move), 302 (Redirect), and 304 (Not Modified) will be used in the experiment.

The properties of the web caches workload are presented in Table 1. From the Table 1, it can be described that the cacheable requests are below 53 % of total web requests. The percentage of one timer is different among three companies, but for the cache in the same company the one timer is nearly equal. In all web caches, the object type is dominated by application, image, and text. More over, the composition of object type contained in the cache in the same company is nearly
equal. The important property that is related to IRT is the web request rate that shows the density of web request. From Table 1, it can be described that all web cache have different web request rate.

C. Simulation

The simulation is conducted using computer program in C# language. The web cache replacement policies that are compared in the experiment are LRU, LFU, GDS(1), and DA-IRT itself. The size of web cache is varied in range 20, 40, 60, 80 percent of the total size of distinct requests in testbed. These policies are implemented in same testbed and then HR and BHR of each web cache replacement policy and web cache size are calculated.

V. EXPERIMENT RESULT AND ANALYSIS

In general, increasing web cache size tends to increasing either hit ratio or byte hit ratio. This characteristic confirms to the previous research. GDS(1) outperforms all web cache replacement policies in terms of HR, but it has a poor performance in terms of BHR. This result is also conformity with previous research. The HR of DA-IRT outperforms LRU and LFU for Petikemas and GIA testbed, while for Telkom testbed DA-IRT don’t outperform all of web replacement policies observed in this research. In general, it can be concluded that the performance of DA-IRT is dependent on the behavior of user request. In general, it can be concluded that the performance of DA-IRT is dependent on the behavior of user request. Based on this finding, it is important to explore the characteristic of web trace in order to determine the properties of workload so that the DA-IRT will be outperform the LFU. If this properties can be found, it could be decided whether or not DA-IRT will be implemented.

Based on the correlation between the average of IRT and temporal locality (see Table 2), the Telkom testbed have lesser correlation than the other testbeds. But, eventhough the correlation is weak for GIA#2, the DA-IRT of GIA#2 outperforms either LRU or LFU in terms of HR. Based on this fact, it can be concluded that it must be other parameters, beside the correlation value, that can be used to guarantee the DA-IRT will outperform. The other parameter could be the gradient value of the linier regression, the maintaining the value of average IRT after purging the object from web cache, the sequence of user request, or spatial locality parameter.

VI. CONCLUSION AND FUTURE WORK

The performance of DA-IRT is dependent on the characteristic of workload. The strong correlation between average IRT and temporal locality can be used as one of parameter that can guarantee the DA-IRT will outperform, but it is not enough. Other parameter like gradient value of the linier regression, maintaining the value of average IRT after purging the object from web cache, the sequence of user request, or spatial locality parameter, have to be explore in depth to support the correlation value in order to recognize the other properties of workload that can be used to guarantee the DA-IRT better than other policies.
VIII. ACKNOWLEDGMENT

Special thank to Garuda Indonesia Airways, PT Telkom, and PT Peti Kemas that have prepared and given the trace log of the web cache for this research.

REFERENCES


