An Improved DG Model for Prediction

Lei Shi¹, Haizhen Shi², Lin Wei³
¹School of Information Engineering, Zhengzhou University, Zhengzhou, China 450001
Email: shilei@zzu.edu.cn
²School of Information Engineering, Zhengzhou University, Zhengzhou, China 450001
Email: shz418@163.com
³School of Software, Zhengzhou University, Zhengzhou, China 450002
Email: weilin@zzu.edu.cn

Abstract—The key of prefetching techniques is to establish an effective prediction model, which is mainly classified into two main categories according to the data structure namely PPM model and DG model. However, both models have their own shortcomings. To overcome their shortcomings, we will propose an improved algorithm based on exponential descendent double dependency graph algorithm. And we will also optimize the model in breadth which obeys Zipf’s distribution and in depth which obeys Gaussian distribution. Experimental results show that the model achieves certain effectiveness in improving prediction accuracy and reducing space complexity.

Index Terms—Web prefetching, Zipf, Gaussian, DG, PPM

I. INTRODUCTION

In recent years, the web has become the primary means for information dissemination. Since the Internet capacity is not keeping pace, the net effect of this growth was a significant increase in the global traffic, damaging the quality of service (availability, reliability, security) and specially the user perceived latency, that is, the time between when a client issues a request for a document and the time the response arrives.

Web prefetching techniques, known as active caching techniques, by analyzing the user’s access history, preprocess the user’s requests before the user demands them explicitly and get the objects into the cache. For users, the prefetching techniques mask the server processing time and network transfer delaying time. The key of prefetching techniques is to establish an effective prediction model and then make an accurate prediction based on the prediction model.

Pandey [1] and Nanopoulos [2], based on previous work, find that the majority of prediction model can be classified into two categories: PPM (Prediction by Partial Match) prediction model which is based on tree structure; DG (Dependency Graph) prediction model which is based on graph structure. In recent years, there have been many researches working on the Web prefetching techniques. Prediction by Partial Match (PPM) is a commonly used technique in web prefetching, where prefetching decisions are made based on historical URLs in a dynamically maintained Markov prediction tree [3]. Guaranteed the accuracy, most of the researches focus on the reducing of the space occupation of the PPM prediction model because of its large space complexity.

The DG prediction algorithm constructs a dependency graph that depicts the pattern of accesses to the objects. As defined by Griffioen [5] in file systems prediction, and implemented by Padmanabhan [4] in web prefetching, there is an arc from one object to another object. When the client request arrives, the server dynamically updates the model, and predicts the next request.

PPM prediction algorithm has better prediction accuracy, but it takes enormous space complexity. Although DG prediction algorithm can overcome PPM’s shortcomings, it can’t distinguish the initial weight of different order and do not take into account how web pages are structured [6], so its prediction accuracy is lower.

Based on the above discussion, we propose an improved algorithm based on exponential descendent double dependency graph algorithm.

II. RELATED WORK

Users go surfing usually among web documents through a hyperlink. Such browsing behavior implies common access patterns and laws. These laws not only determine the popularity of visited pages, but also determine the probability distribution of user browsing depth (the number of pages which user accessed in a web site). By studying the web surfing features and user access behavior, we will optimize the DG to control the scale of model in breadth and in depth. In breadth we discuss about the Zipf’s law, and in depth we take into account Gaussian distribution.

In some papers, they show that web requests follow Zipf’s law. By using Zipf’s law, it can help us to effectively choose which object need to cache. Caching high-frequency objects will greatly enhance the hit ratio.

However, some researches don’t support Zipf’s law thesis such as Guo [8]. In recent researches, from the model point of view, actual streaming data is generally consistent with Zipf-like model.

From the supported Zipf’s law thesis or not, we can clearly recognize: low-frequency objects which account for a higher proportion can be removed without affecting the overall situation; high-frequency objects which account for a lower proportion will affect the final results. Therefore, we can get a conclusion: high-frequency objects can affect the final result, however, the low-frequency objects which is few contributions or no contributions for the hit ratio can be used as a basis for pruning. Caching high-frequency objects will greatly enhance the hit ratio.
Next, the number of links a user follows before the page value first reaches the stopping L threshold is a random variable. The probability distribution of first passage times to a threshold is given asymptotically by the two parameter inverse Gaussian distribution [7].

\[ P(L) = \frac{\sqrt{\lambda}}{2\pi L^3} \exp\left( \frac{-\lambda(L - \mu)^2}{2\mu^2L} \right) \]  

(1)

With mean \( E(L) = \mu \) and variance \( \text{Var}[L] = \mu^3 / \lambda \).

L (user browsing depth) is the number of links that a user accessed in a Web site.

User browsing characteristics provide a theoretical basis for cutting down the vertical scale of the model, that is, user’s browsing depth is not messy but laws and relatively concentrated. Experimental results show that the expectation of user browsing depth is located in between 3 and 6. In order to maintain higher prediction accuracy and control the scale of model, improved DG model dynamically calculates the E (L) as the value of the greatest lookahead window.

III. DG ALGORITHM


First, we consider a web log as a relation table \( T \) that is defined by a set of attributes \( A = \{ A_1, A_2, ..., A_m \} \). Usual attributes include Host, Ident, Authuser, Time, Request, Status, Bytes, Referrer, Agent and Cookie. Assume that transactions generated by different users are identified by a subset of attributes \( S \subseteq A \). Let \( u \) be a set of user ids and \( F : S \rightarrow U \) a function that maps each unique combination of values of \( S \) to a user id of \( U \). Let \( A \in S \) be the Time attribute.

Definition 1. A session \( s \) is an ordered set of transactions in \( T \) which satisfy \( A_{t}(s+1) = A_{t}(s) \) and \( A_{t}(t-1) - A_{t}(t) < \tau \) where \( t \ldots \), \( t \in s \) and \( \tau \) is a given time threshold (30 minutes).

Conceptually, a session defined in Definition 1 is a set of ordered request objects viewed in one visit by the same visitor. In a session, these objects have the order before and after. What’s more, the types of these objects may be different.

Given the above considerations, we define a DG model as follows:

Definition 2. A DG model is a \( \langle S, O, T, W \rangle \) where \( S \) indexes all identified sessions \( s_1, s_2, ..., s_n \), \( O \) indexes all identified objects \( o_1, o_2, ..., o_n \), \( T \) is a type of the object where we can divide the object into two types which are html object and image object, and \( W \) is the lookahead window size.

Definition 3. Let \( F \) be a mapping from \( \langle S, O, T, W \rangle \) to \( M \) that performs that we enter a session, a request object sequence, the type of the object and the size of \( w \), so we can get a matrix.

\[
M(s, t, w) = \begin{cases} 
1 & \text{if } s \in S, o \in O, t \in T, w \in W \\
0 & \text{otherwise}
\end{cases}
\]

Assume the sequence of the session \( s1 \) is \( H1, Ob1, H2, Ob2, H3 \) and \( Ob3 \); the size of \( w \) is 2. Therefore we have

\[
M(s1) = \begin{bmatrix} 
0 & 1 & 0.5 & 0 & 0 \\
0 & 0 & 1 & 0.5 & 0 \\
0 & 0 & 0 & 1 & 0.5 \\
0 & 0 & 0 & 0 & 1 
\end{bmatrix}
\]

Where 1 in the first row and second column stands for the weight from \( H1 \) to \( Ob1 \); 0.5 in the first row and third column stands for the weight from \( H1 \) to \( H2 \); 0 stands for two objects not unreachable. If we enter another session, so we need to update the matrix. It must be explained that in the matrix the size of \( w \) is fixed and we don’t consider the type of the object in order to describe questions from mathematical model.

From the intuitive point of view, we can construct a dependency graph by the rules as follows:

1) The graph has a node for each URL that has ever been accessed. It uses a structure which stores the number of visited nodes. For example, \( A(2) \) means that node \( A \) is accessed for twice.

2) In the graph, there is an arc from node \( A \) to \( B \) if and only if at some point in time a client accessed to \( B \) within \( w \) accesses after \( A \), where \( w \) is the lookahead window size.

3) The weight of the arc is the ratio of the number of accesses to \( B \) within a window after \( A \) to the number of accesses before \( A \) itself. Within \( w \), for \( i \leq w \) the initial weight of arc set \( 1 \); the confidence of each arc is calculated by dividing the counter of the arc by the amount of appearances of the node.

Figure 1 shows the state of the DG algorithm after a training example with the following accesses: \( H1, Ob1, H2, Ob2, H3, Ob3 \) by one user; and \( H1, Ob1, H4, Ob2, H5, Ob3 \) by other user.

Figure 1. DG.
IV. IMPROVED DG ALGORITHM

On the one hand, according to characteristics of web object popularity and Zipf’s second law, improved model removes the noise objects whose weight is less than the threshold $\theta$. On the other hand, according to inverse Gaussian distribution of user browsing depth, improved model sets the value of lookahead window $w$ by dynamically computing the expectation of inverse Gaussian distribution. And improved DG model distinguishes the initial weight of different order by using the exponential descent algorithm which set the weight of arc to be $1/2^i$. Therefore, improved DG model can further improve the prediction accuracy.

Improved DG prediction algorithm constructs a dependency graph by the rules as follows:

1) The graph has a node for each URL that has ever been accessed. It uses a structure which stores the number of visited node. For example, $A (2)$ means that node $A$ is accessed for twice.

2) In the graph, there is an arc from node $A$ to $B$ if and only if at some point in time a client accessed to $B$ within $w$ accesses after $A$, where $w$ is the lookahead window size. But unlike DG, it distinguishes two classes of dependences: dependences to an object of the same page and dependences to an object of another page. The arc is a primary arc if $A$ and $B$ which are objects of different pages. On the contrary, the arc is secondary arc.

3) The weight of the arc is the ratio of the number of accesses to $B$ within a window after $A$ to the number of accesses to $A$ itself. Within $w$, for $i=w$ the initial weight of arc set $1/2^i$, the confidence of each arc is calculated by dividing the counter of the arc by the amount of appearances of the node, both for primary and for secondary arcs.

Figure 2 shows the state of the DG algorithm after a training example with the following accesses: $H1, Ob1, H2, Ob2, H3, Ob3$ by one user; and $H1, Ob1, H4, Ob2, H5, Ob3$ by other user. Arrows with continuous lines represent primary arcs while dashed lines represent secondary arcs. Primary and secondary arcs represent the relation between object accesses of different pages and between object accesses of the same page, respectively.

**Figure 2. Improved DG model.**

For example, the confidence of the arc of node $Ob1$ to node $Ob2$ when using DG prediction algorithms:

$$P (Ob1, Ob2) = (1 + 1) / 2 = 1$$  \hspace{1cm} (2) \hspace{1cm}

The confidence of the arc of object $Ob1$ to object $Ob2$ when using improved DG prediction algorithms:

$$P (Ob1, Ob2) = (1/2 + 1/2) / 2 = 0.5$$  \hspace{1cm} (3) \hspace{1cm}

The pseudo code for building the improved DG prediction model is shown in Figure 3:

**Figure 3. Pseudo code for building the improved DG prediction model.**

V. EXPERIMENTAL EVALUATION

This section presents the experimental results. In order to test the performance of improved DG algorithm, we use a real web log to predict the performance. Experiment data derives from Inventive Information Technology Co. E-commerce Platform log files [9] and Henan University log files [10]. For DG model and improved DG model, the experiment uses 1/2 of log files as a training set and the remaining serve as a test set.

Three performance key metrics are used to evaluate web prefetching benefits. Precision (Pc): measures the ratio of good prediction to the number of prediction. Recall (Rc): measures the percentage of user requested objects that were previously prefetched. From the figure 4, in prediction we can see that when $w$ is 1, DG model and improved DG model have the same prediction accuracy. With the increasing of $w$, prediction accuracy of improved DG model significantly has been enhanced. When $w$ is 3, prediction accuracy of improved DG is steady. And when $w$ continues to increase, the accuracy is not obviously enhanced. But for the DG model, when $w$ is 2, the prediction accuracy reaches its maximum. Later with the increasing of $w$, the accuracy is worse. The reasons for this phenomenon are that DG model can’t distinguish the initial weight of different orders. When $w$ is too large, it actually reduces the weight of first order. So when $w$ is increasing, it will appear that the prediction accuracy declines. What’s more, figure 5 shows that the precision and recall of improved DG has been enhanced higher than one in the figure 4. Because the former has higher amount of embedded objects per page, for example, in e-commerce platform there are a large amount of images. In recall,
improved DG model also has a relatively better performance.

![Figure 4. Henan University log](image)

![Figure 5. Inventive e-commerce platform log](image)

Figure 6 shows that improved DG and PPM are compared in the precision and recall, when the lookahead w is 3. From the picture, we can find that improved DG obtains the similar performance in precision and recall as the PPM.

In storage space, figure 2 shows improved DG model only stores 7 nodes information where PPM model need to store 38 nodes information in figure 7. Therefore, improved DG algorithm can greatly reduce the space complexity of the model.

![Figure 7. PPM model](image)

VI. CONCLUSIONS AND FUTURE WORK

For the two categories of web prediction model, in recent years, the problem of modeling and predicting a user’s accesses on a web-site by PPM has attracted a lot of research interest; however, the research about the DG is relatively less.

In this paper, according to web surfing features and user browsing depth, we improved and optimized the DG prediction model. What we have done is to distinguish the initial weight of different order by using the exponential descent algorithm which set the initial weight of arc to be 1/2i. In contrast to existing algorithm, experimental results show that the model achieves certain effectiveness in improving prediction accuracy and reducing space complexity.

In the future, we will test and analyze the performance of our approach by establishing a unified evaluation model. And we will combine our algorithm with practical application in order to adapt actual work better.

REFERENCES


