

# PART-OF-SPEECH TAGGING FOR CLIENT-SIDE LINK PREFETCHING

A. Georgakis\*

Digital Media Laboratory (DML),  
Department of Applied Physics and Electronics,  
Umeå University, SE-90187 Umeå Sweden  
apostolos.georgakis@tfe.umu.se

## ABSTRACT

In this paper a client-side algorithm that learns and predicts user requests is presented. The proposed approach is based on a user behavior profile. The profile is based on textual information extracted from visited web pages. The novelty of the paper is in the use of a part-of-speech tagger to filter the useful user-keywords. The keywords comprising the profile are employed by a transparent and speculative link weighting mechanism. The generated weights are used in estimating future web traversing. Afterwards some linked web pages are prefetched and stored locally in the browser's cache. A comparison between the proposed algorithms and four other client-side algorithms yield improved cache-hit rates given a moderate bandwidth overhead.

## KEY WORDS

Prefetching, user behavior modeling, bigrams

## 1 Introduction

Modeling user behavior on the Internet is valuable for both content providers and consumers. Consumers may appreciate better *quality of experience* due to responsiveness as a result of prefetching content into the browser's cache. Additionally, modeled user behavior can be exploited in adaptive and personalized Web sites that can make suggestions to the user. In parallel, content providers will grasp the financial benefits of a more pleased consumer that gets the desired information faster.

User behavior patterns on the Internet can be characterized by identifiable regularities among noise. Much of the patterns of activity can be identified and exploited by intelligent mechanisms for learning request patterns. The present paper offers a novel *transparent* and *speculative client-side* web page prefetching algorithm. The term "transparent" implies that there is no user participation in the selection of the pages to be prefetched. "Speculative" entails that future requests are predicted based on knowledge acquainted during past acquired experience.

An extreme variety of alternatives have been proposed and are currently implemented in an effort to decrease the user perceived latency. The solutions range from faster Internet connections, different communication technologies, faster web servers, increased bandwidth either from a single backbone provider or from multiple providers and usage of proxy servers by the Internet service providers for local caching of frequent or "popular" documents. Finally, server side proxies and distributed servers for traffic balancing are also used by popular web sites. All the above can substantially improve typical response times. But still, the latencies exist and are unfortunately amplified when the user is visiting web pages and documents that has never visited in the past.

The above techniques can be described as server-side approaches. An alternative to them is the client-side techniques where the user's PC acts independently from the servers. That is, no input is given from the servers for the access patterns of hosted files. Limited research has been carried out on recorded transactions from the client side [1]-[4]. The using of the individual client's history provides the opportunity for personalization [5, 6]. In this approach the documents that have higher probability to be accessed are *prefetched* and stored locally in the browser's cache. The succeeding requests that involve documents in the cache are served instantly thus avoiding any unnecessary connection to the Web.

Of course it is difficult to accurately determine future traversals in order to prefetch the "correct" documents. Therefore some prefetches may and will never be used. This will result in bandwidth waste and increased workload for the servers. Therefore, any prefetching scheme must operate under the constraint of keeping a balance between resource allocation and time latency reduction.

This paper describes a mechanism for the creation of personal profile that is used to assist in prefetching. The profile is based on the user's *web surfing behavior*. That is, the user traverses links based on the text anchored around them.<sup>1</sup> At this point it is presumed that there is a relationship between the textual con-

---

\*The work was supported by a research grand from the Faculty of Science, Umeå University.

---

<sup>1</sup>He or she reads the text and if it is interesting clicks on it

tent of a web pages and the user’s interests; otherwise the user would not have visited a particular page. Unfortunately, the relationship is difficult to be assessed since there does not exist any *non-invasive* approach to evaluate it. Therefore only *indirect* methods can fulfill the requirement of transparency. The previous implies a text weighting mechanism. That is, a mechanism that assigns weights to the different outbound links and then prefetches the ones with the highest weights.

The novelty of the presented method is in the use of a rule-based part-of-speech (POS) tagger. Tagging is used to identify the *lexical* or *linguistic category* for individual words. Subsequently emphasis is placed on a limited set of word classes (for example *nouns*). The underlying assumption here is that not all the words comprising the visited web pages and the text achored around links are important to the user. They not only offer any clue for the actual preferences but they increase the dimensionality of the modeling space (*curse of dimensionality*). The text preprocessing step described in Subsection 2.1 is one step towards the elimination of textual noise but POS can further assist in the dimensionality reduction direction.

In what follows, the proposed algorithm is described in detail in Section 2. Section 3 reports a series of experimental results for the evaluation of the capabilities of the proposed algorithm. The experiments covered in this section are based on the cache-hit ratio, the usefulness of prediction, the fractional latency reduction and the fractional network traffic.

## 2 Proposed Algorithm

The first step before prefetching involves the creation of the user’s profile. The profile is based on a small subset of the bigramms, that is sequences of successive words, comprising the web pages that the user has visited in the past (Subsection 2.3). Subsections 2.1 - 2.3 describe the mechanism for selecting the bigrams. Finally Subsection 2.4 describes the weighting mechanism employed in assigning weight to the outgoing links be initiating the prefetching.

### 2.1 Text preprocessing

After the retrieval of each individual web page a series of preprocessing steps are carried. Those are the following:

- Identification of the outbound links and the anchored text around them.
- Elimination of links that the browser should not prefetch.
- Removal of the HTML tags and entities.

- Text cleaning or *stopping* using a stop list (articles, determiners, prepositions, pronouns, conjunctions, complementizers, abbreviations).
- Finally stemming is performed. Stemming refers to the elimination of the word suffixes so that the vocabulary shrinks, although it keeps the informative context of the text.

The above preprocessing steps can easily handle any type of web page regardless of the *charset* parameter. That is, depending on the *charset* different stemming algorithms and stop lists can be utilized.

### 2.2 Part of speech tagging

In this paper a simple *rule-based* POS tagger is employed [7]. This tagger is based on the Brown Corpus (BC) which is a tagged corpus. The corpus is divided into two training sub-corpora. The first sub-corpus comprises of the 90% of the BC whereas and the second one corresponds to the remaining 10%.

The tagger is trained initially with the first sub-corpus. The tagger at first assigns tags to each word depending on its most likely tag. This is estimated by examining the tagged sub-corpus without regard to the context. Subsequently the tagger is used to tag the second sub-corpus. This procedure will yield some tagging errors. An erroneous tag occurs when a word is tagged differently by the tagger compared to the initially assigned tag. The higher the frequency of errors for a given word the more ambiguous the word is. Following, a set of human-created rules are applied on top of the tagging errors. The rules associate the erroneous tag with the tags assigned to proceeding or succeeding words in the sentence. The rule that generates the highest reduction in errors (primary errors minus secondary errors) is applied on the second sub-corpus<sup>2</sup>.

After the training process and the creation of the list of rules, new text can be tagged. In doing so the trained tagger is used. Then, the list of rules are applied to the new text in order to reduce the erroneous tags. Moreover, in case of words not seen in the training corpus the tags assigned are the most common tags for words with the same suffix. Once again the training corpus is used to derive this type of information.

The proposed tagger performs as well as taggers based upon probabilistic models. The proposed rule-based tagger overcomes the limitations common in many other rule-based approaches: it is robust, and the rules are automatically acquired. In addition, this tagger has many advantages over stochastic taggers, including: a vast reduction in stored information required, the simplicity of a small set of meaningful rules as opposed to the large tables of statistics needed for

---

<sup>2</sup>Primary errors are the ones corrected by the rule. Secondary errors are the one generated by the rule

stochastic taggers, ease of finding and implementing improvements to the tagger, and better portability from one tag set or corpus genre to another. Finally it should be mentioned that the training process is run only once and the resulted rule list can be used multiple times afterwards.

### 2.3 Content and ngrams

Future actions are estimated using the text anchored around outgoing links. When the user visits a web page and spends some time either reading or looking around for valuable information then the textual content of the page might reveal the so-called *region of interest*. Furthermore, the path of followed links also reveals interests. Hence, when modeling the behavior one can rely on the terms forming the visited pages, the visited and also non-visited links.<sup>3</sup>

In any one of the above three case and under some general *strong* suppositions it can be assumed that the more frequent a term is, the more important it should be for the user. This assumption is plausible since text-preprocessing step has removed a good deal of textual noise. The assumption that the frequency of individual terms is informative can be rather weak though. Studies have shown that the frequency of occurrence for isolated terms compared to sequences of terms or ngrams is an inferior medium for capturing the content of a document [8]. Therefore, an ngrams model, specifically a bigram model will be employed instead of the unigram model.

Let  $\mathcal{S}_t = \{s_i : i \in \{1, 2, \dots, |\mathcal{S}_t|\}\}$  denote the vocabulary found seen by the system until the time moment, where  $|\cdot|$  denotes the cardinality of a set. In the bigram model the unigrams  $s_i$  are substituted by  $b_{ij} = \langle s_i, s_j \rangle$ .

Due to the previous step (POS tagging) each bigram is accompanied with tags. By assuming that certain linguistic categories are more informative than other one can filter out a portion of the bigrams. In the case under consideration nouns and proper nouns combined with adjectives, verbs and other nouns were kept included in the user profile. The previous step can also be combined with various term selection techniques based on document frequency, mutual information, a  $\chi^2$ -test, term strength and dice coefficient [9] can be applied as well in post-processing step after the text preprocessing.

### 2.4 Weighting mechanism

Let  $\mathcal{B}_t$  denote the set of bigrams seen by the system until the time instant  $t$ , that is,  $\mathcal{B}_t = \{b_{\kappa\lambda} : \kappa \neq \lambda \wedge \kappa, \lambda \in \{1, 2, \dots, |\mathcal{S}_t|\}\}$ .<sup>4</sup> Let also  $\mathcal{L}_s$  denote the set

<sup>3</sup>The non-visited links imply repulsive textual content for the user.

<sup>4</sup> $|\mathcal{B}_t| = |\mathcal{S}_t|(|\mathcal{S}_t| - 1)$

of bigrams,  $\mathcal{L}_s = \{ {}_s b^1, {}_s b^2, \dots, {}_s b^{n_s} \}$ , associated with the  $s$ th outbound link in the page under consideration and  ${}_s b^i \in \mathcal{B}_t, \forall i$  where  $n_s = |\mathcal{L}_s|$ . In order to compute the total weight for  $\mathcal{L}_s$  the proposed algorithm takes into consideration the fact that the frequency of appearance for a given bigram denotes the bigram's importance for the user. By adding the frequencies of the bigrams comprising a particular link the algorithm can get a rough approximation of its importance. The previous is transcribed into:

$$w_1^s = \frac{\sum_{i=1}^{n_s} f_{b2d}({}_s b^i)}{n_s}, \quad (1)$$

where  $f_{b2d}({}_s b^i)$  denotes the bigram-to-document frequency of the  $b^i$  bigram in the collection of visited web pages.

Unfortunately, the  $f_{b2d}({}_s b^i)$  might not reflect the true trends of the user's interest when surfing the net; rather the method of presenting the information contained in the web pages. On the other hand the user's interest is reflected by the path of followed links. In that case the algorithm must exploit this information as well.

Since the selection of a link is governed by the text anchored around it, this information must be exploited. In doing so the text around the visited as well as the non-visited links is taken into consideration. For that the algorithm keeps a record of the frequency of appearance for each bigram in the anchored text in both the visited and the non-visited links. That is, for each bigram  ${}_s b^i$  there are two different counters. The first counter which will be denoted by  $v_{b2l}({}_s b^i)$  records the frequency of appearance of the bigram in the visited links. The second counter is denoted by  $nv_{b2l}({}_s b^i)$  and corresponds to the frequency of the links that were not visited by the user. Then the importance of that bigram is:

$$\frac{v_{b2l}({}_s b^i)}{p} - \frac{nv_{b2l}({}_s b^i)}{q}, \quad (2)$$

where  $p$  is the total number of visited links and  $q$  the number of non-visited links. So, the weight of the link according to the contribution of the bigrams found around the links is:

$$w_2^s = \sum_{i=1}^{n_s} \left[ \frac{v_{b2l}({}_s b^i)}{p} - \frac{nv_{b2l}({}_s b^i)}{q} \right]. \quad (3)$$

And the weight of the  $s$ th link is determined by the following linear combination:

$$T_s = aw_1^s + (1 - a)w_2^s \quad (4)$$

where  $a \in \{0, 1\}$  is a scaling parameter. This parameter is adjusted by the user and manages the aggressiveness of the prediction. What this means is that although the content of the visited pages might reveal

partially the preferences it is the traversal path that reflects the user’s attention. Therefore, if the user feels confident that the visited links are relevant to the preferences (not just a simple random walk) then the parameter  $a$  can be set close to unity other wise it can be set to any other value. By setting the value equal or close to one any random walking in the past has a very little effect in the final performance.

### 3 Prefetching evaluation

The evaluation procedure can be described as an online approach [10]. In the online approach the collected data are separated into distinct training and test sets. The algorithm attempts to determine the appropriate model by learning from the training set. Afterwards the user profile is updated with each new user action.

The performance of the proposed algorithm is tested against four different prefetching algorithms which are:

- a variant of the proposed algorithm without the POS tagging mechanism
- using the text inside the `<a href=...> </a>`
- an enhancement of the above technique based on the adaboost technique [11]
- random selection of links.

For the evaluation of the performance two traces consisting of 500.000 and 175.000 pages are used. The first one is used for training and the second for testing all five algorithms. It must be noted here that the contextual statistics generated during the training phase are the starting point for the proposed prefetching algorithms for each volunteer individually, during the testing phase. That means that during the training phase for each individual user a profile is created and in the testing phase the corresponding profiles are used. Moreover, the profiles are constantly updated during the testing phase for each user separately.

The metrics employed are the *cache-hit ratio*, the *usefulness of prediction*, the *fractional latency reduction* and the *fractional network traffic* [12]. Cache-hit ratio is the ratio of prefetched pages that a user requested to all the pages the prefetch agent retrieved and represents the accuracy of the prediction. The above metric constitute the accuracy of the prediction. Usefulness of predictions is the quantitative relation between the useful prefetched pages (that the user requested) to all the requested pages. Literally, it corresponds to the coverage of the prediction. Fractional latency reduction is the ratio between the decrease due to prefetching of the observed UPL without a caching system ( $UPL_{np}$ ) from the UPL with a caching system ( $UPL_p$ ) to the observed UPL without caching

( $UPL_{np}$ ). Finally, fractional network traffic is the ratio between the amount of bytes transmitted from a web server to the user’s client to the total number of bytes requested. It represents the bandwidth overhead added to the network traffic of the non prefetched case, when prefetching is engaged. It is obvious that since some of the future behavior will not be predicted precisely some of the prefetched documents will never be requested. These redundant documents add undesired network traffic and should not have been prefetched.

Figure 1(a) depicts the cache-hit ratio curves for each of the five prefetching algorithms. The in question figure depicts the average cache-hit curves for all the volunteers in predefined steps of the outbound links volume. From the figure it is evident that all curves will converge to the same cache-hit point when all the outbound links will be fetched. This is obvious since the number of useful outbound links is not a function of the prefetching mechanism employed, rather than the personal preferences of each user.

It is important to note that interpolation has been applied during the formation of these curves when it was necessary. Interpolation is justified in the cases when the percentage of outbound links in a page is not an integer number. That is, when a page has for example nine outbound links, then prefetching 35% of the links implies 2.15 prefetches. In that case, all the ratios are calculated by interpolating the closest values available.

Furthermore, from Fig. 1(a) is evident that the proposed algorithm exhibits better performance than its prime candidate, which is the algorithm entitled “Bigrams in link & adaboosting,” in each prefetch volume, with a maximum difference of 2.8% higher cache-hit rate which is achieved at a 35% prefetch volume.

Figure 1(b) depicts the usefulness of the prediction. From this figure is apparent that the significant increase of the hit ratio is induced with less significant cost in the usefulness (maximum observed overhead between the proposed algorithm and its prime candidate is 3% when there is an 35% prefetching).

Finally, as it can be seen in Fig. 1(c), prefetching does reduce network latency in all prefetch volumes whereas the bandwidth overhead is more or less the same for all the algorithms (Fig. 1(d)). The latency is reduced from between 6.6% to 12.3% when only 5% of the links are retrieved to nearly 45% when 35% of the links have been prefetched. At the same time the proposed algorithm achieves better results with a difference in the reduction of the latency in the range from 0.3% to 2.8% when again compared with its prime candidate.

## 4 Conclusions

A prefetching algorithm for hyperlinks was proposed in this paper. The algorithm aimed towards the creation

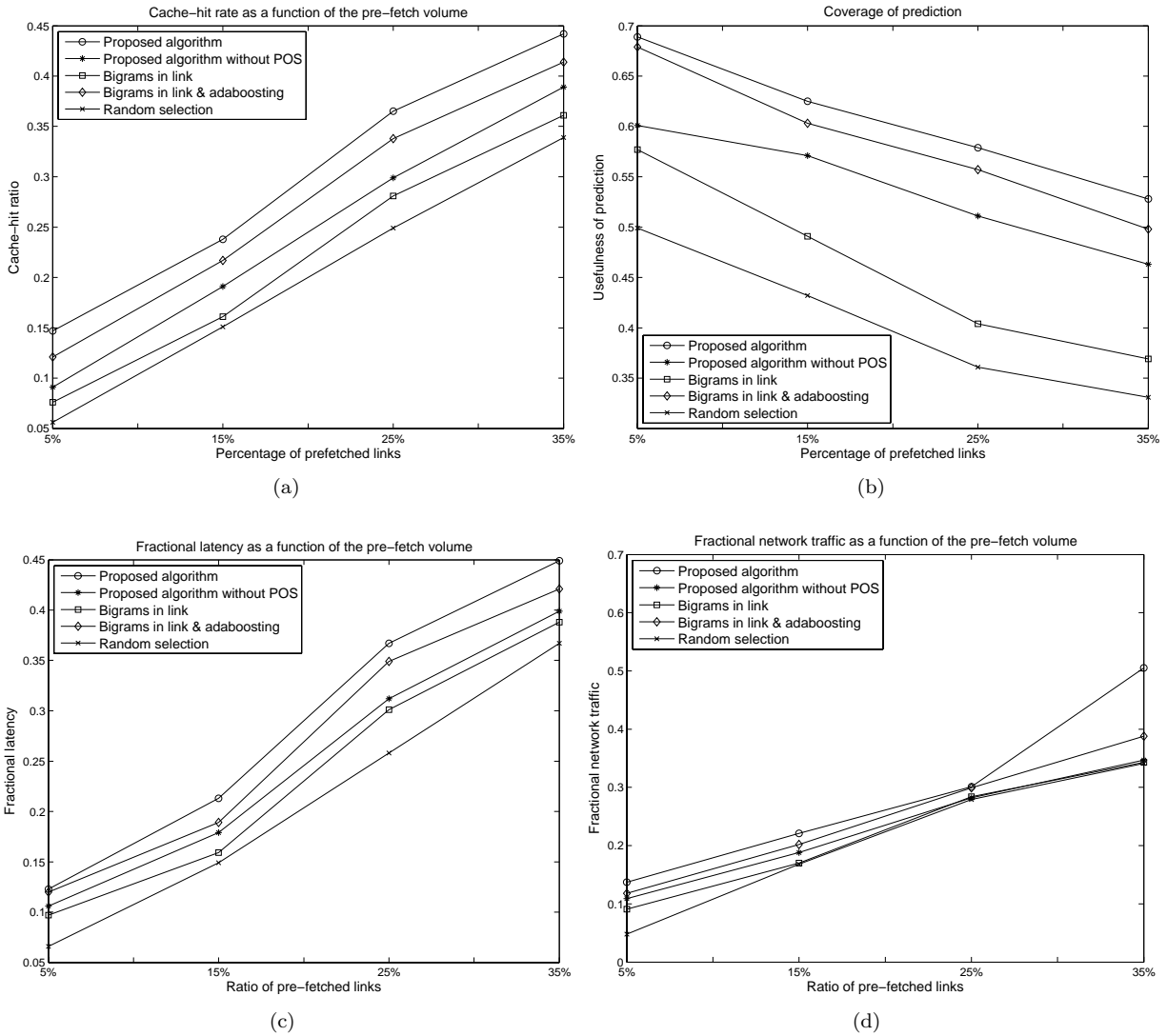


Figure 1. The average performance for each of the five prefetching algorithms for all the volunteers and for: (a) cache-hit ratio, (b) usefulness of prediction, (c) fractional latency reduction ratio and (d) fractional network traffic ratio.

of a Internet behavior profile for the end user. The proposed algorithm relies on POS tagging for bigram selection that will populate the profile. The proposed algorithm has been tested against four other algorithms and demonstrated superior performance, even when compared to an adaboosting classifier.

## References

- [1] Catledge, L.D., Pitkow, J.E.: Characterizing browsing strategies in the world wide web. *Computer Networks and ISDN Systems* **26** (1995) 1065–1073
- [2] Crovella, M.E., Bestavros, A.: Self-similarity in world wide web traffic: Evidence and possible causes. *IEEE/ACM Transactions on Networking* **5** (1997) 835–846
- [3] Cunha, C.R., Bestavros, A., Crovella, M.E.: Characteristics of www client-based traces. Technical Report TR-95-010, Computer Science Department, Boston University (1995)
- [4] Tauscher, L., Greenberg, S.: How people revisit web pages: Empirical findings and implications for the design of history systems. *International Journal of Human Computer Studies* **47** (1997) 97–138
- [5] Aggarwal, C.C., Wolf, J., Yu, P.S.: Caching on the World Wide Web. *IEEE Trans. on Knowledge and Data Eng.* **11** (1999) 95–107

- [6] Shim, J., Scheuermann, P., Vingralek, R.: Proxy Cache Algorithms: Design, Implementation, and Performance. *IEEE Trans. on Knowledge and Data Eng.* **11** (1999) 549–562
- [7] Brill, E.: A simple rule-based part-of-speech tagger. In: *Proc. of 3rd Conf. on Applied Natural Language Processing (ANLP'92)*. (1992) 152–155
- [8] Manning, D., Schütze, H.: *Foundations of Statistical Natural Language Processing*. Cambridge, MA: MIT Press (1999)
- [9] Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: *Proc. of 14th Int. Conf. on Machine Learning (ICML'97)*. (1997) 412–420
- [10] Davison, B.D.: Learning web request patterns. In Pouloussis, A., Levene, M., eds.: *Web Dynamics: Adapting to Change in Content, Size, Topology and Use*. Springer (2004) 435–460
- [11] Georgakis, A., Li, H.: Web documents prefetching based on an ensemble of classifiers. In: *Proc. of IASTED Int. Conf. on Internet and Multimedia Systems and Applications (EuroIMSA'05)*, Grindelwald, Switzerland (2005)
- [12] Davison, B.D.: The design and evaluation of web prefetching and caching techniques. PhD thesis, State University of New Jersey (2002)