Designing Efficient Sampling Techniques to Detect Webpage Updates

Qingzhao Tan[†] Ziming Zhuang[‡] Prasenjit Mitra[†‡] C. Lee Giles[†‡]
qtan@cse.psu.edu zzhuang@ist.psu.edu pmitra@ist.psu.edu giles@ist.psu.edu
[†]Computer Science and Engineering [‡]Information Sciences and Technology
The Pennsylvania State University, University Park, PA 16802, USA

ABSTRACT

Due to resource constraints, Web archiving systems and search engines usually have difficulties keeping the entire local repository synchronized with the Web. We advance the state-of-art of the sampling-based synchronization techniques by answering a challenging question: *Given a sampled webpage and its change status, which other webpages are also likely to change*? We present a study of various downloading granularities and policies, and propose an adaptive model based on the update history and the popularity of the webpages. We run extensive experiments on a large dataset of approximately 300,000 webpages to demonstrate that it is most likely to find more updated webpages in the current or upper directories of the changed samples. Moreover, the adaptive strategies outperform the non-adaptive one in terms of detecting important changes.

Terms:Management, Design, Algorithms, Experimentation
Keywords:Web crawler, sampling, search engines.

1. INTRODUCTION

Web content archiving systems and search engines rely on crawlers to harvest webpages and store them in the local repositories. These local copies are later retrieved to respond to relevant queries. Although ideally the local copies of webpages are synchronized with their online counterparts, due to resource constraints it is impractical for crawlers to constantly monitor and download every single webpage. We investigate a typical scenario in which due to resource constraints, a crawler is only allowed to periodically re-visit a fixed number of webpages and update the corresponding local copies when a change has been detected. We define this fixed number of pages as the *download resources* and the periodical interval as the *download cycle*. Thus, the crawler's goal is to use the given *download resources* to *maximize* the number of updated webpages downloaded in each *download cycle*.

Several studies have been proposed to address this challenge. Some probabilistic models have been exploited to approximate the observed update history of a webpage and predict its future changes [2], in which the fast-changing pages are more likely to be crawled. The limitation of these models is that they need to gather sufficient and accurate knowledge about each webpage's change history. To address this problem, Cho, et al., [3] have prescribed to randomly sample webpages from each website, and then a crawler should crawl an entire website once it detects that the website has more changed samples than other sites. However, they do not support this choice with any theoretical or empirical evidence. It has also been shown that update patterns are different across various parts of a website [4].

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.
WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.
ACM 978-1-59593-654-7/07/0005.

In this paper, we study and improve the sampling-based update detection techniques by making the following **contributions**: 1) We propose a new sampling algorithm to detect webpage updates. In our algorithm, sampling is done at the webpage level and each webpage is equally likely to be selected as a sample; 2) We propose two downloading policies for change detection. Specifically, given a sampled webpage, we exploit its neighborhood link structure and directory structure to discover other updated webpages. 3) We propose two biased sampling algorithms based on the change history [2] and PageRank [6].

2. SAMPLING-BASED UPDATE DETECTION

Our sampling-based update detection algorithm is based on the assumption that *relevant* webpages have similar change patterns, and it works as follows. For each of the download cycles, the crawler randomly samples a set of pages from the local repository and downloads from the Web their *remote* versions. Based on the comparison of the sample's local and remote copies, the crawler decides on a probability φ whether to use these samples as seeds and crawl their *neighbors* within a distance, which is the *download granularity* d.

2.1 Downloading Policy

The download granularity d determines that, given a changed sample, which additional webpages and how many of them should the crawler choose to download. In **Link-based downloading policy (LB)**, we define d as the maximum number of hops the crawler traverses away from f the sampled seed page p_s by following its outlinks. In **Directory-based downloading policy (DB)**, we define d as the distance between the depths of two webpages' URLs. More specific, d is calculated as the difference in the number of slashes in the two URLs plus 1.

2.2 Adaptive Download Probability

Basic adaptive strategy The *download probability* φ can be adapted to the sampled webpage's *latest* change status as follows:

$$\varphi = \begin{cases} 1 & \text{if the sampled webpage has changed,} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

History-adaptive strategy (HA) and weighted-history–adaptive strategy (WHA) We model the change of a webpage p following a Poisson process [5] with its own change rate λ_p. We set the *download probability* φ to be the probability that p changes in the interval $(o, t]$ where $t = 1$, i.e., one download cycle.

$$\varphi = Pr\{T \leq t\} = \int_0^t \lambda_p e^{-\lambda_p t} dt = 1 - e^{-\lambda_p t} = 1 - e^{-\lambda_p}. \quad (2)$$

We compute λ_p based on the change history of the webpage p within n download cycles:

$$\lambda_p = \frac{\sum_{i=1}^{n} w_i \mathbf{I_1}(p_i)}{n}. \quad (3)$$

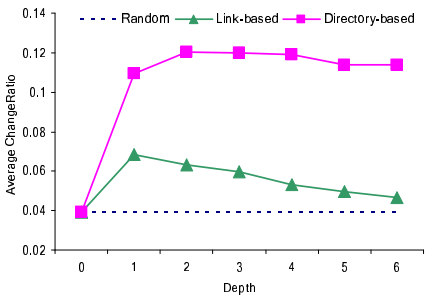

Figure 1: Comparison of two downloading policies in $\overline{C}$. DB has an explicit percentage of improvement over LB.

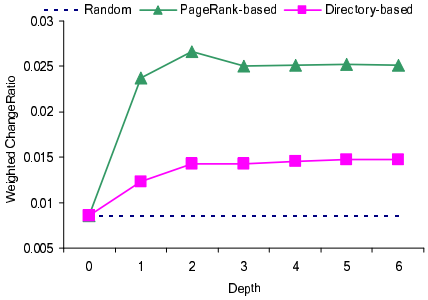

Figure 2: Comparison of the PageRank-adaptive strategy and the non-adaptive DB in weighted $\overline{C}$.

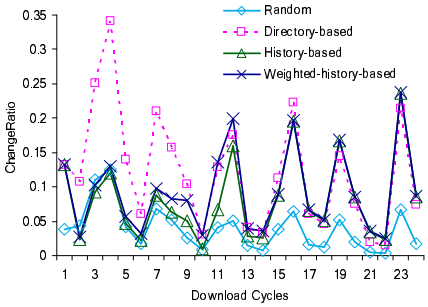

Figure 3: Comparison of the long-term performance over 24 cycles. HA and WHA eventually outperformed DB.

where $\mathbf{I_1}(p_i)$ is an indicator function. It returns 1 when p is changed in the i^{th} cycle, 0 otherwise. And w_i is the weight to differentiate changes occurred in various download cycles. Typically, changes occurred in the more recent download cycles are more important, thus having higher weights. We assign the weight w_i for the i^{th} download cycle as $w_i = i/\sum_1^n$ where n denotes the total number of cycles the crawler has processed thus far.

PageRank-adaptive strategy(PRA) We set φ for each sampled webpage in proportion to its *popularity* as reflected in its PageRank [6]. The PageRank of a webpage, p_i, is calculated as follows:

$$\varphi \propto PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in IL(p_i)} \frac{PR(p_j)}{OL(p_j)}. \quad (4)$$

where $p_1, p_2, ..., p_N$ are the webpages on the Web, $IL(p_i)$ is the set of pages that link to p_i, $OL(p_j)$ is the number of out-going links on page p_j, and N is the total number of webpages in the repository.

3. EVALUATION AND DICUSSION

We conducted extensive experiments on a large collection of real webpages from the Internet Archive's WayBack Machine [1]. Our dataset contains approximately 300,000 distinct webpages from 210 websites, with their historical versions dated between 1996 and 2005. We set the download cycle to be two weeks and the download resource to be 25,000 webpages. We used the *ChangeRatio* as the evaluation metric, which is the fraction of *downloaded and changed* webpages over the total number of *downloaded* webpages in one download cycle. We measured the per-download-cycle *ChangeRatio* C_i, and the *average ChangeRatio* $\overline{C}$ which is the mean C_i over all download cycles. *Weighted* $\overline{C}$ improves the definition of $\overline{C}$ by assigning different weights to the changed webpages. In our evaluation, we set the weights proportional to the webpage's PageRank. The Random Node Sampling (RNS) method, in which the crawler uniformly re-download at random webpages in each download cycle, was used as the baseline algorithm.

First, we implemented LB and DB to compare the performance of the two definitions on the *download granularity* d. The download probability φ was set to 1 for changed samples and 0 for the unchanged ones. We tuned d from 1 to 6. As illustrated in Figure 1, DB clearly outperforms LB in terms of $\overline{C}$, which indicates that following the directory structure is a good strategy for the crawler to discover newly updated webpages. We then fixed the downloading policy as DB for the following experiments. Figure 1 also shows that the $\overline{C}$ is strongly influenced by the download granularity d. For both download policies, $\overline{C}$ is the lowest when $d = 0$, which is equivalent to the RNS baseline. The $\overline{C}$ goes up when d increases, and after it reaches its peak at $d = d_{opt}$ it gradually levels off. Empirically, $d_{opt} = [2, 3]$ for DB, and $d_{opt} = 1$ for LB. This indicates that given a changed sample, the most-likely-to-change webpages are usually located in the same or the upper directory of the sample.

To investigate the impact of the download probability φ, we tune φ based on three adaptive strategies: PRA, HA, and WHA. For PRA, we get the PageRank as integers in [0,10] and set φ to be $1/11$, $2/11$, ..., and $11/11$ for webpages with *PageRank* 0, 1, ..., and 10, respectively. For HA and WHA, we assign φ according to Equation (2). Figure 2 shows the comparison results measured in terms of *Weighted* $\overline{C}$. We see that PRA clearly outperforms DB by almost 100% in the *Weighted* $\overline{C}$ metric. It could imply the fact that webpages with high PageRank tend to link to other webpages also with high PageRank. From Figure 3, we see that at the beginning, the two *history-adaptive* strategies performed worse than DB. However, as time went by and more knowledge about the change history was obtained, the C_is for the two history-adaptive strategies gradually caught up in the following download cycles, and eventually outperformed DB. This reconfirms that in the long-run, algorithms that take into account the past changes of the webpages will have a more accurate prediction about their future changes. Between the two history-adaptive strategies, WHA performed better than HA. Moreover, WHA outperformed DB in download cycle 16, earlier than HA did in cycle 18. This is a strong implication that a crawler takes into account the change history of the webpages should pay more attention to the more recent changes than the older ones.

4. CONCLUSION

We proposed a sampling-based algorithm to study a challenging problem in Web crawling: Given a sampled subset of the webpages in a local repository, how can a crawler exploit a fixed amount of available download resources to make the local repository as up-to-date as possible? We investigated in detail the parameter space in our algorithm, especially the download granularities and adaptive download probabilities, to achieve the optimal performance in terms of *ChangeRatio*.

5. REFERENCES

[1] The wayback machine. http://www.archive.org/web/web.php.
[2] CHO, J., AND GARCIA-MOLINA, H. Effective page refresh policies for web crawlers. *ACM TODS 28*, 4 (2003), 390–426.
[3] CHO, J., AND NTOULAS, A. Effective change detection using sampling. In *VLDB '02* (2002).
[4] FETTERLY, D., MANASSE, M., NAJORK, M., AND WIENER, J. L. A large-scale study of the evolution of web pages. In *Proceedings of WWW '04* (Budapest, Hungary, 2004).
[5] GRIMMETT, G., AND STIRZAKER, D. *Probability and Random Processes, 2nd ed*. Oxford University Press, 1992.
[6] PAGE, L., BRIN, S., MOTWANI, R., AND WINOGRAD, T. The pagerank citation ranking: Bringing order to the web. Tech. rep., Stanford Digital Library Technologies Project, 1998.