Estimating Frequency of Change

Junghoo Cho
University of California, LA
and
Hector Garcia-Molina
Stanford University

Many online data sources are updated autonomously and independently. In this paper, we make the case for estimating the change frequency of data to improve Web crawlers, Web caches and to help data mining. We first identify various scenarios, where different applications have different requirements on the accuracy of the estimated frequency. Then we develop several "frequency estimators" for the identified scenarios, showing analytically and experimentally how precise they are. In many cases, our proposed estimators predict change frequencies much more accurately and improve the effectiveness of applications. For example, a Web crawler could achieve 35% improvement in "freshness" simply by adopting our proposed estimator.

Categories and Subject Descriptors: G.3 [**Mathematics of Computing**]: Probability and Statistics—*Time series analysis*; H.2.4 [**Database Management**]: Systems—*Distributed databases*

General Terms: Algorithms, Design, Measurement

Additional Key Words and Phrases: Change frequency estimation, Poisson process

1. INTRODUCTION

With the explosive growth of the Internet, many data sources are available online. Most of the data sources are autonomous and are updated independently of the clients that access the sources. For instance, popular news Web sites, such as CNN and NY Times, update their contents periodically whenever there are new developments. Also, many online stores update the price and availability of their products, depending on their inventory and on market conditions.

Since the sources are updated autonomously, the clients usually do not know exactly when and how often the sources change. However, some of the clients can significantly benefit by estimating the change frequency of the sources [Brewington and Cybenko 2000b]. For instance, the following applications can use the estimated change frequency to improve their effectiveness.

— **Improving a Web crawler:** A Web crawler is a program that automatically visits Web pages and builds a local snapshot and/or index of Web pages. In order to maintain the snapshot or index up-to-date, the crawler periodically revisits the

Authors' email addresses: Junghoo Cho (cho@cs.ucla.edu), Hector Garcia-Molina (hector@cs.stanford.edu)
Permission to make digital/hard copy of all or part of this material without fee for personal or classroom use provided that the copies are not made or distributed for profit or commercial advantage, the ACM copyright/server notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee.
© 20YY ACM 0000-0000/20YY/0000-0001 $5.00

pages and updates the pages with fresh images. A typical crawler usually revisits the entire set of pages periodically and updates them all. However, if the crawler can estimate how often an individual page changes, it may revisit only the pages that have changed (with high probability) and improve the "freshness" of the local snapshot without consuming as much bandwidth. According to [Cho and Garcia-Molina 2000b], a crawler may improve the "freshness" by orders of magnitude in certain cases if it can adjust the "revisit frequency" based on the change frequency.

— **Improving the update policy of a data warehouse:** A data warehouse maintains a local snapshot, called a *materialized view*, of underlying data sources, which are often autonomous. This materialized view is usually updated during off-peak hours, to minimize the impact on the underlying source data. As the size of the data grows, however, it becomes more difficult to update the view within the limited time window. If we can estimate how often an individual data item (e.g., a row in a table) changes, we may selectively update only the items likely to have changed, and thus incorporate more changes within the same amount of time.

— **Improving Web caching:** A Web cache saves recently accessed Web pages, so that the next access to the page can be served locally. Caching pages reduces the number of remote accesses and minimizes access delay and network bandwidth. Typically, a Web cache uses an LRU (least recently used) *page replacement policy*, but it may improve the *cache hit ratio* by estimating how often a page changes. For example, if a page was cached a day ago and if the page changes every hour on average, the system may safely discard that page, because the cached page is most probably obsolete.

— **Data mining:** In many cases, the frequency of change itself might be useful information. For instance, when a person suddenly accesses his bank account very often, it may signal fraud, and the bank may wish to take an appropriate action.

In this paper, we study how we can effectively estimate how often a data item (or an element) changes. We assume that we access an element *repeatedly* through *normal* activities, such as periodic crawling of Web pages or the users' repeated access to Web pages. From these repeated accesses, we detect changes to the element, and then we estimate its change frequency.

We have motivated the usefulness of estimating the frequency of change, and how we accomplish the task. However, there exist important challenges in estimating the frequency of change, including the following:

(1) **Incomplete change history:** Often, we do not have complete information on how often and when an element changed. For instance, a Web crawler can tell if a page has changed between accesses, but it cannot tell how many times the page changed.

Example 1 A Web crawler accessed a page on a daily basis for 10 days, and it detected 6 changes. From this data, the crawler may naively conclude that its change frequency is $6/10 = 0.6$ times a day. But this estimate can be smaller than the actual change frequency, because the page may have changed more than once between some accesses. Then, what would be a fair estimate for the change frequency? How can the crawler account for the missed changes? □

Previous work has mainly focused on how to estimate the change frequency given the complete change history [Taylor and Karlin 1998; Winkler 1972].

(2) **Irregular access interval:** In certain applications, such as a Web cache, we cannot control how often and when a data item is accessed. The access is entirely decided by the user's request pattern, so the access interval can be arbitrary. When we have limited change history and when the access pattern is irregular, it becomes very difficult to estimate the change frequency.

EXAMPLE 2 In a Web cache, a user accessed a Web page 4 times, at day 1, day 2, day 7 and day 10. In these accesses, the system detected changes at day 2 and day 7. Then what can the system conclude on its change frequency? Does the page change every (10 days)/2 = 5 days on average? □

(3) **Difference in available information:** Depending on the application, we may get different levels of information for different data items. For instance, certain Web sites tell us when a page was last-modified, while many Web sites do not provide this information. Depending on the scenario, we may need different "estimators" for the change frequency, to fully exploit the available information.

In this paper, we study how we can estimate the frequency of change when we have *incomplete* change history of a data item. (Traditional statistical estimators assume a complete change history.) To that end, we first identify various issues and place them into a taxonomy (Section 2). Then for each branch in the taxonomy, we propose an "estimator" and show analytically how good the proposed estimator is (Sections 4 through 5). In summary, our paper makes the following contributions:

— We identify the problem of estimating the frequency of change with *incomplete* data and we present a formal framework to study the problem.

— We develop several estimators that measure the frequency of change much more effectively than existing ones. As will be clear from our discussion, more "natural" estimators have undesirable properties, significantly impacting the effectiveness of the applications using them. By examining these estimators carefully, we propose much improved estimators. For the scenario of Example 1, for instance, our estimator will predict that the page changes 0.8 times per day (as opposed to the 0.6 we guessed earlier), which reduces the "bias" by 33% on average.

— We present analytical results that show how effective our proposed estimators are. Also, we experimentally verify our proposed estimator using *real* dataset collected from the Web. The experiments show that our estimator predicts change frequencies much more accurately than existing ones and significantly improves the effectiveness of a Web crawler.

1.1 Related work

The problem of estimating change frequency has been long studied in statistics community [Taylor and Karlin 1998; Winkler 1972; Misra and Sorenson 1975; Canavos 1972]. However, most of the previous work assumed that the complete change history is known, which is not true in many practical scenarios. In this paper, we study how to estimate the change frequency based the *incomplete* change history. As far as we know, Reference [Matloff 2002] is the only other work that studies change

frequency estimation with incomplete history. Reference [Matloff 2002] was written concurrently with our work; our estimator is similar to the one proposed in [Matloff 2002]. The main difference is that our estimator avoids the singularity problem that will be discussed later. Also, [Matloff 2002] proposes an estimator that uses the last-modification date when the access is *regular*. Our estimator proposed in Section 5 can be used when the access is irregular.

References [Cho and Garcia-Molina 2000b; Coffman, Jr. et al. 1998] study how a crawler should refresh the local copy of remote Web pages to improve the "freshness" of the local copies. Assuming that the crawler knows how often Web pages change, [Cho and Garcia-Molina 2000b] shows that the crawler can improve the freshness significantly. In this paper, we show how a crawler can *estimate* the change frequency of pages, to implement the refresh policy proposed in the reference. Reference [Edwards et al. 2001] also study how to improve the crawler's freshness using nonlinear programming.

Reference [Cho and Garcia-Molina 2000a] proposes an architecture in which a crawler can adjust page revisit frequencies based on estimated page change frequencies. To implement such an architecture, it is necessary to develop a good frequency estimation technique, which is what we study in this paper.

Various researchers have experimentally estimated the change frequency of pages [Wolman et al. 1999; Douglis et al. 1999; Wills and Mikhailov 1999]. Note that most of the work used a naive estimator, which is significantly worse than the estimators that we propose in this paper. We believe their work can substantially benefit by using our estimators. One notable exceptions are [Brewington and Cybenko 2000a; 2000b], which use last-modified dates to estimate the *distribution* of change frequencies over a set of pages. However, since their analysis predicts the distribution of change frequencies, not the change frequency of an individual page, the method is not appropriate for the scenarios in this paper.

Many researchers studied how to build a scalable and effective Web cache, to minimize the access delay, the server load and the bandwidth usage [Yu et al. 1999; Gwertzman and Seltzer 1996; Baentsch et al. 1997]. While some of the work touches on the consistency issue of cached pages, they focus on developing a *new* protocol that may reduce the inconsistency. In contrast, our work proposes a mechanism that can be used to improve the *page replacement policy* on *existing* architecture.

In data warehousing context, a lot of work has been done to efficiently maintain *materialized view*s [Hammer et al. 1995; Harinarayan et al. 1996; Zhuge et al. 1995]. However, most of the work focused on different issues, such as minimizing the size of the view while reducing the query response time [Harinarayan et al. 1996].

2. TAXONOMY OF ISSUES

Before we start discussing how to estimate the change frequency of an element, we first need to clarify what we mean by "change of an element." What do we mean by the "element" and what does the "change" mean? To make our discussion concrete, we assume that an element is a Web page and that a change is *any* modification to the page. However, note that the technique that we develop is independent of this assumption. The element can be defined as a whole Web site or a single row in a database table, etc. Also a change may be defined as more than, say, a

30% modification to the page, or as updates to more than 3 columns of the row. Regardless of the definition, we can apply our technique/analysis, as long as we have a clear notion of the element and a precise mechanism to detect changes to the element.

Given a particular definition of an element and a change, we assume that we repeatedly access an element to estimate how often the element changes. This access may be performed at a regular interval or at random intervals. Also, we may acquire different levels of information at each access. Based on how we access the element and what information is available, we develop the following taxonomy.

(1) **How do we trace the history of an element?** In this paper, we assume that we repeatedly access an element, either actively or passively.

— **Passive monitoring:** We do not have any control over when and how often we access an element. In a Web cache, for instance, Web pages are accessed only when users access the page. In this case, the challenge is how to analyze the *given* change history to best estimate its change frequency.

— **Active monitoring:** We actively monitor the changes of an element and can control the access to the element. For instance, a crawler can decide how often and when it will visit a particular page. When we can control the access, another important question is how often we need to access a particular element to best estimate its change frequency. For instance, if an element changes about once a day, it might be unnecessary to access the element every minute, while it might be insufficient to access it every month.

In addition to the access control, different applications may have different access intervals.

— **Regular interval:** In certain cases, especially for active monitoring, we may access the element at a regular interval. Obviously, estimating the frequency of change will be easier when the access interval is regular. In this case, (number of detected changes)/(monitoring period) may give us good estimation of the change frequency.

— **Random interval:** Especially for passive monitoring, the access intervals might be irregular. In this case, frequency estimation is more challenging.

(2) **What information do we have?** Depending on the application, we may have different levels of information regarding the changes of an element.

— **Complete history of changes:** We know exactly when and how many times the element changed. In this case, estimating the change frequency is relatively straightforward; It is well known that (number of changes)/(monitoring period) gives "good" estimation of the frequency of change [Taylor and Karlin 1998; Winkler 1972]. In this paper, we do not study this case.

— **Last date of change:** We know when the element was last modified, but not the complete change history. For instance, when we monitor a bank account which records the last transaction date and its current balance, we can tell when the account was last modified by looking at the transaction date.

— **Existence of change:** The element that we monitor may not provide any history information and only give us its current status. In this case, we can compute the "signature" of the element at each access and compare these signatures between accesses. By comparing signatures, we can tell whether the element changed or

not. However, we cannot tell how many times or when the element changed by this method.
In Section 4, we study how we can estimate the frequency of change, when we only know whether the element changed or not. Then in Section 5, we study how we can exploit the "last-modified date" to better estimate the frequency of change.

(3) **How do we use estimated frequency?** Different applications may use the frequency of change for different purposes.
— **Estimation of frequency:** In data mining, for instance, we may want to study the correlation between how often a person uses his credit card and how likely is a default. In this case, it might be important to *estimate* the frequency accurately. In Sections 4 and 5, we study the problem of *estimating* the frequency of change.
— **Categorization of frequency:** We may only want to classify the elements into several frequency categories. For example, a Web crawler may perform a "small-scale" crawl every week, crawling only the pages that are updated very often. Also, the crawler may perform a "complete" crawl every three months to completely refresh all pages. In this case, the crawler may not be interested in exactly how often a page changes. It may only want to *classify* pages into two categories, the pages to visit every week and the pages to visit every three months.
In the extended version of this paper [Cho and Garcia-Molina 2002], we discuss how we may use the Bayesian inference method in this scenario. Since our goal is categorize pages into different frequency classes, say, pages that change every week (class C_W) and pages that change every month (class C_M), we store the probability that page p belongs to each frequency class ($P\{p \in C_W\}$ and $P\{p \in C_M\}$) under the Bayesian method and update these probabilities based on detected changes. For instance, if we learn that page p has not changed for one month, we increase $P\{p \in C_M\}$ and decrease $P\{p \in C_W\}$.

3. PRELIMINARIES

In this section, we will review some of the basic concepts for frequency estimation and a Poisson-process model. A reader familiar with a Poisson process and estimation theory may skip this section.

In Section 3.1, we first explain how we model the changes of an element. A model for the change is essential to compare various "estimators." Then in Section 3.2, we explain the concept of "quality" of an estimator. Even with the same experimental data, different estimators give different values for the change frequency. Thus we need a well-defined metric that measures the effectiveness of different estimators.

3.1 Poisson process: the model for the changes of an element

In this paper, we assume that an element changes by a *Poisson process*. A Poisson process is often used to model a sequence of random events that happen independently with a fixed rate over time. For instance, occurrences of fatal auto accidents, arrivals of customers at a service center, etc., are usually modeled by Poisson processes.

To describe the Poisson-process model we use $X(t)$ to refer to the number of occurrences of a change in the interval $(0, t]$. Then a Poisson process of *rate* or *frequency* λ has the following property:

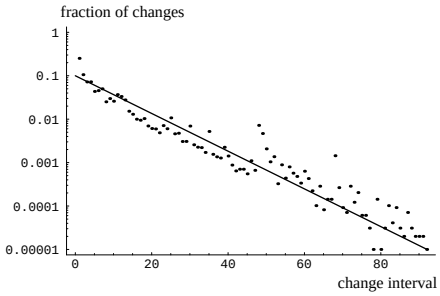

Fig. 1. The actual change intervals of pages and the prediction of the Poisson process model

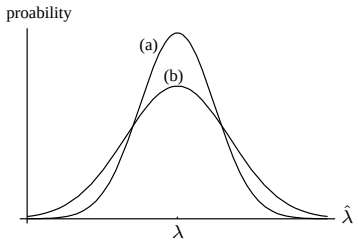

Fig. 2. Two possible distributions of the estimator $\hat{\lambda}$

For $s \geq 0$ and $t > 0$, the random variable $X(s+t)-X(s)$ has the Poisson probability distribution $\Pr\{X(s+t)-X(s) = k\} = \frac{(\lambda t)^k}{k!}e^{-\lambda t}$ for $k = 0, 1, \ldots$

The parameter λ of a Poisson process is the *average frequency* or *rate* that a change occurs. We can verify this fact by calculating how many events are expected to occur in a unit interval:

$$E[X(t+1) - X(t)] = \sum_{k=0}^{\infty} k\Pr\{X(t+1) - X(t) = k\} = \sum_{k=1}^{\infty} k\frac{\lambda^k e^{-\lambda}}{k!} = \lambda.$$

Experiments reported in the literature [Brewington and Cybenko 2000a; Cho and Garcia-Molina 2000a] strongly indicate that the changes to many Web pages follow a Poisson process. For instance, Reference [Cho and Garcia-Molina 2000a] traces the change history of half million Web pages and compares the observations with the prediction of a Poisson model. For example, Figure 1 is one of the graphs in the paper showing the distribution of successive change intervals of real Web pages. In the graph, the horizontal axis represents the change intervals of pages, and the vertical axis shows the fraction of the changes occurred at the given interval. Under a Poisson model, this distribution should be exponential [Taylor and Karlin 1998; Snyder 1975; Wackerly et al. 1997], which is exactly what we observe from the graph: The distribution of real Web page changes (dots in the graph) is very close to the exponential straight line (note that the vertical axis is logarithmic). Reference [Brewington and Cybenko 2000b] also analyzes real Web data and reports that a Poisson process is a good approximation to model Web data.

While the results reported in the existing literature strongly indicate that changes can be modeled by a Poisson process, it is still possible that some pages may not follow the Poisson model. To address this concern, in Section 6.1 we also examine our estimators when the elements do not follow the Poisson model.

3.2 Quality of estimator

The goal of this paper is to estimate the frequency of change λ, from the repeated accesses to an element. To estimate the frequency, we need to summarize the observed change history, or *samples*, as a single number that corresponds to the

frequency of change. In Example 1, for instance, we summarized the six changes in ten visits as the change frequency of $6/10 = 0.6$/day. We call this summarization procedure the *estimator* of the change frequency. Clearly, there exist multiple ways to summarize the same observed data, which can lead to different change frequencies. In this subsection, we will study how we can compare the effectiveness of various estimators.

An estimator is often expressed as a function of the observed variables. For instance, let X be the number of changes that we detected and T be the total access period. Then, we may use $\hat{\lambda} = X/T$ as the estimator of the change frequency λ as we did in Example 1. (We use the notation "hat" to show that we want to measure the parameter underneath it.) Here, note that X is a random variable, which is measured by sampling (or repeated accesses). Therefore, the estimator $\hat{\lambda}$ is also a random variable that follows a certain probability distribution. In Figure 2, we show two possible distributions of $\hat{\lambda}$. As we will see, the distribution of $\hat{\lambda}$ determines how effective the estimator $\hat{\lambda}$ is.

(1) **Bias:** Let us assume that the element changes at the average frequency λ, which is shown at the bottom center of Figure 2. Intuitively we would like the distribution of $\hat{\lambda}$ to be centered around the value λ. Mathematically, $\hat{\lambda}$ is said to be *unbiased*, when the expected value of $\hat{\lambda}$, $E[\hat{\lambda}]$, is equal to λ.

(2) **Efficiency:** In Figure 2, it is clear that $\hat{\lambda}$ may take a value other than λ, even if $E[\hat{\lambda}] = \lambda$. For any estimator, the estimated value might be different from the real value λ, due to some statistical variation. Clearly, we want to keep the variation as small as possible. We say that the estimator $\hat{\lambda}_1$ is more *efficient* than the estimator $\hat{\lambda}_2$, if the distribution of $\hat{\lambda}_1$ has smaller variance than that of $\hat{\lambda}_2$. In Figure 2, for instance, the estimator with the distribution (a) is more efficient than the estimator of (b).

(3) **Consistency:** Intuitively, we expect that the value of $\hat{\lambda}$ approaches λ, as we increase the sample size. This convergence of $\hat{\lambda}$ to λ can be expressed as follows: Let $\hat{\lambda}_n$ be the estimator with sample size n. Then $\hat{\lambda}_n$ is said to be a *consistent* estimator of λ if

$$lim_{n\to\infty}\Pr\{|\hat{\lambda}_n - \lambda| \leq \epsilon\} = 1 \quad \text{for any positive } \epsilon.$$

4. ESTIMATION OF FREQUENCY: EXISTENCE OF CHANGE

How can we estimate how often an element changes, when we only know whether the element changed or not between our accesses? Intuitively, we may use X/T (X: the number of detected changes, T: monitoring period) as the estimated frequency of change, as we did in Example 1. This estimator has been used in the existing literature that estimates the change frequency of Web pages [Douglis et al. 1999; Wills and Mikhailov 1999; Wolman et al. 1999].

In Section 4.1 we first study how effective this naive estimator X/T is, by analyzing its *bias*, *consistency* and *efficiency*. Then in Section 4.2, we will propose a new estimator, which is less intuitive than X/T, but is much more effective.

4.1 Intuitive frequency estimator: X/T

To help our discussion, we first define some notation. We assume that we access the element n times at a regular interval I. (Estimating the change frequency for irregular accesses is discussed in Section 4.3.) Assuming that T is the *total* time elapsed during our n accesses, $T = nI = n/f$, where $f(= 1/I)$ is the frequency at which we access the element. We use X to refer to the total number of changes that we detected during n accesses. We also assume that the changes of the element follow a Poisson process with rate λ. Then, we can define the frequency ratio $r = \lambda/f$, the ratio of the change frequency to the access frequency. When r is larger than 1 ($\lambda > f$), the element changes more often than we access it, and when r is smaller ($\lambda < f$), we access the element more often than it changes.

Note that our goal is to estimate λ, given X and $T(= n/f)$. However, we may estimate the frequency ratio $r(= \lambda/f)$ first and estimate λ indirectly from r (by multiplying r by f). In the rest of this subsection, we will assume that our estimator is the frequency ratio $\hat{r}$, where

$$\hat{r} = \frac{\hat{\lambda}}{f} = \frac{1}{f}\left(\frac{X}{T}\right) = \frac{X}{n}.$$

Note that we need to measure X repeated accesses to the element and use the X to estimate r.

(1) **Is the estimator $\hat{r}$ biased?** As we argued in Example 1, the estimated $\hat{r}$ will be smaller than the actual r, because the *detected* number of changes, X, will be smaller than the *actual* number of changes. Furthermore, this bias will grow larger as the element changes more often than we access it (i.e, as $r = \lambda/f$ grows larger), because we miss more changes when the element changes more often. The following theorem formally proves this intuition.

THEOREM 4.1 *The expected value of the estimator $\hat{r}$ is*

$$E[\hat{r}] = 1 - e^{-r}.$$ □

PROOF. To compute $E[\hat{r}]$, we first compute the probability that the element does not change between accesses. We use X_i to indicate whether the element changed or not in the ith access. More precisely,

$$X_i = \begin{cases} 1 & \text{if the element changed in } i\text{th access,} \\ 0 & \text{otherwise.} \end{cases}$$

Then, X is the sum of all X_i's, $X = \sum_{i=1}^{n} X_i$.
Assuming q is the probability that the element does not change during time interval I $(= 1/f)$, $q = \Pr\{X(t+I) - X(t) = 0\} = e^{-\lambda I} = e^{-r}$. By definition, X_i is equal to zero when the element does not change between the $(i-1)$th and the ith access. Because the change of the element is a Poisson process, the changes at different accesses are independent, and each X_i takes the value 1 with probability $(1 - q)$ independently from other X_i's. Since X is equal to m when m X_i's are equal to 1

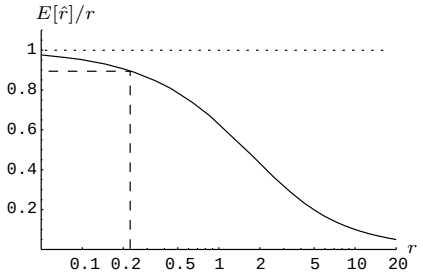

Fig. 3. Bias of the intuitive estimator $\hat{r} = X/n$

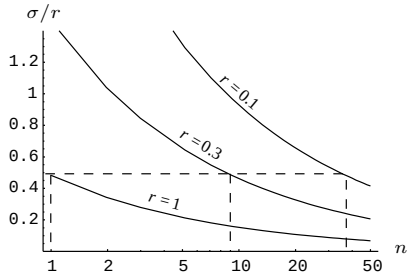

Fig. 4. Statistical variation of $\hat{r} = X/n$ over n

$\Pr\{X = m\} = \binom{n}{m}(1-q)^m q^{n-m}$. Therefore,

$$E[\hat{r}] = \sum_{m=0}^{n} \frac{m}{n} \Pr\left\{\hat{r} = \frac{m}{n}\right\} = \sum_{m=0}^{n} \frac{m}{n} \Pr\{X = m\} = 1 - e^{-r}.$$

Note that when $\hat{r}$ is unbiased, $E[\hat{r}]$ is always equal to r. Clearly, $1 - e^{-r}$ is not r, and the estimator $\hat{r}$ is biased. In Figure 3 we visualize the bias of $\hat{r}$ by plotting $E[\hat{r}]/r$ over r. The horizontal axis is logarithmic to show the values more clearly when r is small and large. (In the rest of this paper, we use a logarithmic scale, whenever convenient.) If $\hat{r}$ is unbiased ($E[\hat{r}] = r$), the graph $E[\hat{r}]/r$ would be equal to 1 for any r (the dotted line), but because the estimated $\hat{r}$ is smaller than the actual r, $E[\hat{r}]/r$ is always less than 1. From the graph, it is clear that the estimator $\hat{r}$ is not very biased ($E[\hat{r}]/r \approx 1$) when $r(= \lambda/f)$ is small (i.e., when the element changes less often than we access it), but the bias is significant ($E[\hat{r}]/r \ll 1$), when r is large. Intuitively, this happens because we miss more changes as we access the element less often (when r is large). From the graph, we can see that the bias is smaller than 10% ($E[\hat{r}]/r > 0.9$) when the frequency ratio r is smaller than 0.21. That is, we should access the element $1/0.21 \approx 5$ times as frequently as it changes, in order to get less than 10% bias.

(2) **Is the estimator $\hat{r}$ consistent?** The estimator $\hat{r} = X/n$ is not consistent, because the bias of $\hat{r}$ does not decrease even if we increase the sample size n; the difference between r and $E[\hat{r}]$ ($E[\hat{r}]/r = (1-e^{-r})/r$) remains the same independently of the size of n.

This result coincides with our intuition; $\hat{r}$ is biased because we miss some changes. Even if we access the element for a longer period, we still miss a certain fraction of changes, if we access the element at the same frequency.

(3) **How efficient is the estimator?** To evaluate the efficiency of $\hat{r}$, we compute its standard deviation.

COROLLARY 4.2 *The standard deviation of the estimator $\hat{r} = X/n$ is*

$$\sigma[\hat{r}] = \sqrt{e^{-r}(1-e^{-r})/n}.$$ □

PROOF. The proof is similar to that of Theorem 4.1. For the complete proof, see Appendix A.

Remember that the standard deviation tells us how clustered the distribution of $\hat{r}$ is around $E[\hat{r}]$; Even if $E[\hat{r}] \approx r$, the estimator $\hat{r}$ may take a value other than r,

because our sampling process (i.e., access to the element) inherently induces some statistical variation.

From the statistics theory, we know that $\hat{r}$ takes a value in the interval $(E[\hat{r}] - 2\sigma, E[\hat{r}]+2\sigma)$ with 95% probability, assuming $\hat{r}$ follows the normal distribution [Wackerly et al. 1997]. In most applications, we want to make this confidence interval (whose length is proportional to σ) small compared to the actual frequency ratio r. Therefore, we want to reduce σ/r, the ratio of the confidence interval to the frequency ratio, as much as we can. In Figure 4, we show how this ratio changes over the sample size n by plotting its graph. Clearly, the statistical variation σ/r decreases as n increases; While we *cannot* decrease the *bias* of $\hat{r}$ by increasing the sample size, we *can* reduce the *statistical variation* with more samples.

Also note that when r is small, we need a larger sample size n to get the same variation σ/r. For example, to make $\sigma/r = 0.5$, n should be 1 when $r = 1$, while n should be 9 when $r = 0.3$. We explain what this implies by the following example.

EXAMPLE 3 A crawler wants to estimate the change frequency of a Web page by visiting the page 10 times, and it needs to decide on the access frequency.

Intuitively, the crawler should not visit the page too slowly, because the crawler misses many changes and the estimated change frequency is biased. But at the same time, the crawler should not visit the page too often, because the statistical variation σ/r can be large and the estimated change frequency may be inaccurate.

For example, let us assume that the actual change frequency of the page is, say, once every week ($\lambda = 1$/week), and the crawler accesses the page once every two weeks ($f = 1/2$ weeks). Then the bias of the estimated change frequency is 57%! (When $r = 2$, $E[\hat{r}]/r \approx 0.43$.) On the other hand, if the crawler revisits the page every day ($f = 1$/day), then the statistical variation is large and the 95% confidence interval is 1.5 times as large as the actual frequency! (When $r = 1/7$, the 95% confidence interval, $2\sigma/r$ is 1.5.) In the next subsection, we will try to identify the best revisit frequency for this example based on an improved estimator. □

4.2 Improved estimator: $-\log\left(\frac{\bar{X}+0.5}{n+0.5}\right)$

While the estimator X/T is known to be quite effective when we have a *complete* change history of an element [Taylor and Karlin 1998; Winkler 1972], our analysis showed that it is less than desirable when we have an *incomplete* change history. The estimator is highly biased and we cannot reduce the bias by increasing the sample size. In this subsection, we propose another estimator $-\log((\bar{X}+0.5)/(n+0.5))$, which has more desirable properties.

Essentially, our new estimator is based on the result of Theorem 4.1. According to the theorem, the expected value of the estimator $\frac{X}{n}$ is $E\left[\frac{X}{n}\right] = 1 - e^{-r}$. Intuitively, we may consider this formula as $\frac{X}{n} = 1 - e^{-r}$, which can be rearranged to $r = -\log\left(\frac{n-X}{n}\right)$. From this formula we suspect that if we use $-\log\left(\frac{n-X}{n}\right)$ as the estimator for r, we may get an unbiased estimate.[1]

[1]We can also derive the estimator $-\log(n - X/n)$ through the maximum likelihood estimator method. Here, we provide more intuitive derivation.

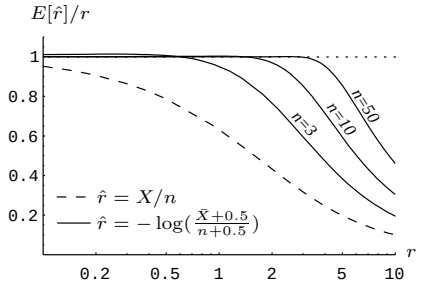

Fig. 5. Bias of the estimator $-\log(\frac{\bar{X}+0.5}{n+0.5})$

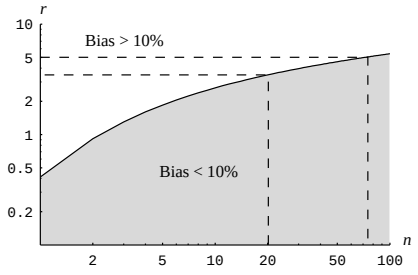

Fig. 6. The region where the estimator $-\log(\frac{\bar{X}+0.5}{n+0.5})$ is less than 10% biased

While intuitively attractive, the estimator $-\log\left(\frac{n-X}{n}\right)$ has a mathematical singularity. When the element changes in all our accesses (i.e., $X = n$), the estimator produces infinity, because $-\log(0/n) = \infty$. This singularity makes the estimator technically unappealing, because the expected value of the new estimator is now infinity. (In other words, the estimator is biased to infinity!) We can avoid this singularity by adding a small constant, 0.5, to $\bar{X}$ and n[2]:

$$\hat{r} = -\log\left(\frac{n - X + 0.5}{n + 0.5}\right)$$

In the remainder of this subsection, we will formally study the properties of this new estimator. Assuming $\bar{X}$ is $n - X$ (the number of accesses that the element did *not* change) we can simplify the estimator as follows.

$$\hat{r} = -\log\left(\frac{\bar{X} + 0.5}{n + 0.5}\right)$$

(1) **Is the estimator biased?** To see whether the estimator is biased, we compute the expected value of $\hat{r}$ as we did for Theorem 4.1.

COROLLARY 4.3 *The expected value of the new estimator* $\hat{r} = -\log\left(\frac{\bar{X}+0.5}{n+0.5}\right)$ *is*

$$E[\hat{r}] = -\sum_{i=0}^{n} \log\left(\frac{i + 0.5}{n + 0.5}\right) \binom{n}{i} (1 - e^{-r})^{n-i} (e^{-r})^{i}. \quad (1)$$

PROOF. Proof is straightforward. See Appendix A.

To show the bias of our new estimator we plot the graph of $E[\hat{r}]/r$ over r in Figure 5. For comparison, we also show the graph of the previous estimator X/n in the figure. From the graph, we can see that our new estimator $-\log(\frac{\bar{X}+0.5}{n+0.5})$ is much better than X/n. While X/n is heavily biased for most r, our new estimator is practically unbiased for $r < 1$ for any $n \geq 3$.

Also, note that the bias of the new estimator *decreases* as we increase the sample size n. For instance, when $n = 3$, $\hat{r}$ shows bias if $r > 1$, but when $n = 50$, it is not heavily biased until $r > 3$. This property has a significant implication in

[2]In Appendix B, we show why we selected the value 0.5 to avoid the singularity.

practice. If we use the estimator X/n, we can reduce the bias *only* by adjusting the *access frequency* f (or by adjusting r), which might not be possible for certain applications. However, if we use our new estimator, we can reduce the bias to the desirable level, simply by increasing the *number of accesses* to the element. For this reason, we believe our new estimator can be useful for a wider range of applications than X/n is.

Given that the estimator becomes less biased as the sample size grows, we may ask how large the sample size should be in order to get an *unbiased* result. For instance, what sample size gives us less than 10% bias? Mathematically, this can be formulated as follows: Find the region of n and r, where

$$\left| \frac{E[\hat{r}] - r}{r} \right| \leq 0.1$$

is satisfied. From the formula of Equation 1, we can numerically compute the region of n and r where the above condition is met, and we show the result in Figure 6. In the figure, the gray area is where the bias is less than 10%. Note that the unbiased region grows larger as we increase the sample size n. When $n = 20$, $\hat{r}$ is unbiased when $r < 3.5$, but when $n = 80$, $\hat{r}$ is unbiased when $r < 5$. We illustrate how we can use these graphs for the selection of the revisit frequency shortly.

(2) **How efficient is the estimator?** As we discussed in Section 4.1, $\hat{r}$ may take a value other than r even if $E[\hat{r}] \approx r$, and the value of σ/r tells us how large this statistical variation can be.

We computed σ/r of $-\log(\frac{\bar{X}+0.5}{n+0.5})$, similarly to Theorem 4.1, and we show the results in Figure 7. As expected, the statistical variation σ/r gets smaller as the sample size n increases. For instance, σ/r is 0.4 for $r = 1.5$ when $n = 10$, but σ/r is 0.2 for the same r value when $n = 40$.

Also note that the statistical variation σ/r takes its minimum at $r \approx 1.5$ within the unbiased region of r. (When r is large, the estimator is heavily biased and is not very useful.) For instance, when $n = 20$, the estimator is practically unbiased when $r < 2$ (the bias is less than 0.1% in this region) and within this range, σ/r is minimum when $r \approx 1.35$. For other values of n, we can similarly see that σ/r takes its minimum when $r \approx 1.5$. We can use this result to decide on the revisit frequency for an element.

EXAMPLE 4 A crawler wants to estimate the change frequency of a Web page by visiting it 10 times. While the crawler does not know exactly how often that particular page changes, say many pages within the same domain are known to change roughly once every week. Based on this information, the crawler wants to decide how often to access that page.

Because the statistical variation (thus the confidence interval) is smallest when $r \approx 1.5$ and because the current guess for the change frequency is once every week, the optimal revisit frequency for that page is 7 days $\times$ 1.5 $\approx$ once every 10 days. Under these parameters, the estimated change frequency is less than 0.3% biased and the estimated frequency may be different from the actual frequency by up to 35% with 75% probability. We believe that this confidence interval will be more than adequate for most crawling and caching applications.

In certain cases, however, the crawler may learn that its initial guess for the change

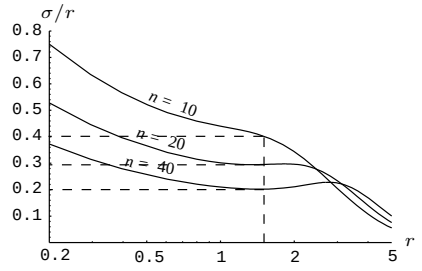

Fig. 7. The graph of σ/r for the estimator $-\log(\frac{\bar{X}+0.5}{n+0.5})$

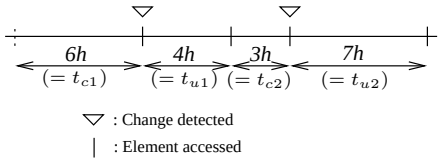

Fig. 8. An example of irregular accesses

frequency may be quite different from the actual change frequency, and the crawler may want to adjust the access frequency in the subsequent visits. We briefly discuss on this adaptive policy later. □

(3) **Is the estimator consistent?** The following theorem shows that our new estimator is indeed consistent:

THEOREM 4.4 *For the estimator* $\hat{r} = -\log(\frac{\bar{X}+0.5}{n+0.5})$, $\lim_{n\to\infty} E[\hat{r}] = r$ *and* $\lim_{n\to\infty} V[\hat{r}] = 0$ □

PROOF. From Corollary 4.3,

$$E[\hat{r}] = -\sum_{i=0}^{n} \log\left(\frac{i+0.5}{n+0.5}\right)\binom{n}{i}(1-e^{-r})^{n-i}(e^{-r})^{i}.$$

That is, $E[\hat{r}]$ is the expected value of $-\log\left(\frac{i+0.5}{n+0.5}\right)$, where i follows the binomial distribution $B(n, e^{-r})$. When n goes to infinity $B(n, e^{-r})$ becomes the normal distribution $N(ne^{-r}, \sqrt{ne^{-r}(1-e^{-r})})$ by the DeMoivre-Laplace limit theorem [Wackerly et al. 1997].
Then, assuming x denotes $(i+0.5)/(n+0.5)$, x follows the normal distribution

$$N\left(\frac{e^{-r}+1/2n}{1+1/2n}, \sqrt{\frac{e^{-r}(1-e^{-r})}{n(1+1/2n)^2}}\right).$$

When n goes to infinity, $\frac{e^{-r}+1/2n}{1+1/2n} \to e^{-r}$ and $\sqrt{\frac{e^{-r}(1-e^{-r})}{n(1+1/2n)^2}} \to 0$, so the distribution of x becomes an impulse function $\delta(x-e^{-r})$, whose non-zero value is only at e^{-r} [Courant and David 1989]. Therefore, when $n \to \infty$, $E[\hat{r}]$ is the same as the expectation of $-\log x$ when x follows $\delta(x-e^{-r})$ distribution. That is,

$$\lim_{n\to\infty} E[\hat{r}] = -\int_0^1 (\log x)\ \ \delta(x-e^{-r})dx = -\log e^{-r} = r.$$

Similarly, we can prove that $\lim_{n\to\infty} E[\hat{r}^2] = r^2$ and

$$\lim_{n\to\infty} V[\hat{r}] = \lim_{n\to\infty}\left(E[\hat{r}^2] - \{E[\hat{r}]\}^2\right) = r^2 - r^2 = 0.$$

4.3 Irregular access interval

When we access an element at irregular intervals, estimation becomes more complicated. For example, assume that we detected a change when we accessed an element after 1 hour and we detected another change when we accessed the element after 10 hours. While all changes are considered equal when we access the element at regular intervals, in this case the first change "carries more information" than the second, because if the element changes more than once every hour, we will definitely detect a change when we accessed the element after 10 hours.

In order to obtain an estimator for irregular case, we can use a *maximum likelihood estimator* [Wackerly et al. 1997]. Informally, the maximum likelihood estimator computes which λ value has the highest probability of producing the observed set of events, and use this value as the estimated λ value. Using this method for the irregular access case, we obtain the following equation[3]:

$$\sum_{i=1}^{m} \frac{t_{ci}}{e^{\lambda t_{ci}} - 1} = \sum_{j=1}^{n-m} t_{uj} \tag{2}$$

Here, t_{ci} represents the interval in which we detected the ith change, and t_{uj} represents the jth interval in which we did not detect a change. Also, m represents the total number of changes we detected from n accesses. Note that all variables in Equation 2 (except λ) can be measured by an experiment. Therefore, we can compute the estimated frequency by solving this equation for λ. Also note that all access intervals, t_{ci}'s and t_{uj}'s, take part in the equation. It is because depending on the access interval, the detected change/non-change carries a different level of information. We illustrate how we can use the above estimator by the following example.

Example 5 We accessed an element 4 times in 20 hours (Figure 8), in which we detected 2 changes (the first and the third accesses). Therefore, the two changed intervals are $t_{c1} = 6h$, $t_{c2} = 3h$ and the two unchanged intervals are $t_{u1} = 4h$, $t_{u2} = 7h$. Then by solving Equation 2 using these numbers, we can estimate that $\lambda = 2.67$ changes/20 hours. Note that the estimated λ is slightly larger than 2 changes/20 hours, which is what we actually observed. This result is because the estimator takes "missed" changes into account. □

In this paper we do not formally analyze the bias and the efficiency of the above estimator, because the analysis requires additional assumption on how we access the element. However, we believe the proposed estimator is "good" for two reasons:

(1) The estimated λ has the highest probability to generate the observed changes.

(2) When the access to the element follows a Poisson process, the estimator is consistent. That is, as we access the element more, the estimated λ converges to the actual λ.

Note, however, that when we detect changes in all accesses ($m = n$), the estimator suffers from a singularity of $\lambda = \infty$. In particular, the estimator reduces to $-\log(\bar{X}/n)$ (with the singularity problem) when the access interval is regular. We

[3]For derivation, see Appendix C

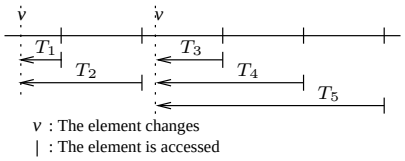

Fig. 9. Problems with the estimator based on last modified date

have not found a disciplined way to avoid this singularity, and we believe that the estimator in Section 4.2, $-\log(\bar{X}+0.5/n+0.5)$, is a better choice when the access interval is regular.

5. ESTIMATION OF FREQUENCY: LAST DATE OF CHANGE

When the last-modification date of an element is available, how can we use it to estimate change frequency? For example, assume that a page changed 10 hours before our first access and 20 hours before our second access. Then what will be a fair guess for its change frequency? Would it be once every 15 hours? In this section, we propose a new estimator that uses the last-modified date for frequency estimation. Since our final estimator is not easy to understand directly, we derive the estimator step by step in this section. We note that our derivation in this section is different from standard estimation techniques (e.g., the maximum likelihood estimator) because standard methods lead to complex and not-easy-to-implement estimators for practical applications. As we will see, the estimator proposed in this section is algorithmically simple and has negligible bias.

We first derive the initial version of our estimator based on the following well-known lemma [Wackerly et al. 1997]:

LEMMA 5.1 *Let T be the time to the previous event in a Poisson process with rate λ. Then the expected value of T is $E[T]=1/\lambda$.* □

That is, in a Poisson process the expected time to the last change is $1/\lambda$. Therefore, if we define T_i as the time from the last change at the ith access, $E[T_i]$ is equal to $1/\lambda$. When we accessed the element n times, the sum of all T_i's, $\mathcal{T}=\sum_{i=1}^{n} T_i$, is $E[\mathcal{T}]=\sum_{i=1}^{n} E[T_i]=n/\lambda$. From this equation, we suspect that if we use $n/\mathcal{T}$ as our estimator, we may get an unbiased estimator $E[n/\mathcal{T}]=\lambda$. Note that $\mathcal{T}$ in this equation is a number that needs to be measured by repeated accesses.

While intuitively appealing, this estimator has a serious problem because the element may not change between some accesses. In Figure 9, for example, the element is accessed 5 times but it changed only twice. If we apply the above estimator naively to this example, n will be 5 and $\mathcal{T}$ will be $T_1+\cdots+T_5$. Therefore, this naive estimator practically considers that the element changed 5 times with the last modified dates of $T_1, T_2, \ldots, T_5$. This estimation clearly does not match with the actual changes of the element, and thus leads to bias.[4] Intuitively, we may get a better result if we divide the actual number of changes, 2, by the sum of T_2 and T_5, the final last-modified dates for the two changes. Based on this intuition, we modify the naive estimator to the one shown in Figure 10.

[4]We can verify the bias by computing $E[n/\mathcal{T}]$ when $\lambda \ll f$.

```
Init()  /* initialize variables */
   N = 0;  /* total number of accesses */
   X = 0;  /* number of detected changes */
   T = 0;  /* sum of the times from changes */

Update(Ti, Ii) /* update variables */
   N = N + 1;
   /* Has the element changed? */
   If (Ti < Ii) then
      /* The element has changed. */
      X = X + 1;
      T = T + Ti;
   else
      /* The element has not changed */
      T = T + Ii;

Estimate() /* return the estimated lambda */
   return X/T;
```

Fig. 10. The estimator using last-modified dates

The new estimator consists of three functions, `Init()`, `Update()` and `Estimate()`, and it maintains three global variables `N`, `X`, and `T`. Informally, `N` represents the number of accesses to the element, `X` represents the number of detected changes, and `T` represents the sum of the time to the previous change at each access. (We do not use the variable `N` in the current version of the estimator, but we will need it later.) Initially, the `Init()` function is called to set all variables to zero. Then whenever the element is accessed, the `Update()` function is called, which increases `N` by one and updates `X` and `T` values based on the detected change. The argument `Ti` to `Update()` is the time to the previous change in the ith access and the argument `Ii` is the interval between the accesses. If the element has changed between the $(i-1)$th access and the ith access, `Ti` will be smaller than the access interval `Ii`. Note that the `Update()` function increases `X` by one, only when the element has changed (i.e., when `Ti` $<$ `Ii`). Also note that the function increases `T` by `Ii`, not by `Ti`, when the element has not changed. By updating `X` and `T` in this way, this algorithm implements the estimator that we intend. Also note that the estimator of Figure 10 predicts the change frequency λ directly. In contrast, the estimator of Section 4 predicts the change frequency by estimating the frequency ratio r.

To study the bias of this estimator, we show the the graph of $E[\hat{r}]/r$ of this estimator in Figure 11. We computed this graph analytically using the formula derived in Appendix D. To compute the graph, we assumed that we access the element at a regular interval I $(= 1/f)$ and we estimate the frequency ratio $r = \lambda/f$ (the ratio of the change frequency to the access frequency). Remember that $E[\hat{r}]/r = 1$ when the estimator is not biased, which is shown as a dotted line. The solid line shows the actual graphs of the estimator for various n.

We can see that the estimator has significant bias when n is small, while the bias is relatively small when n is large (i.e, after many accesses to the element). For instance, when $n = 10$, the graph of the new estimator is fairly close to 1,

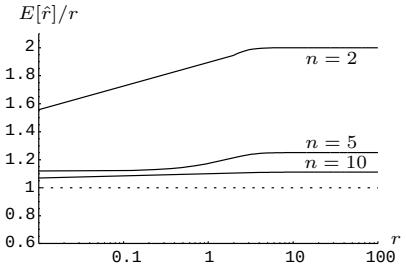

Fig. 11. Bias of the estimator in Figure 10

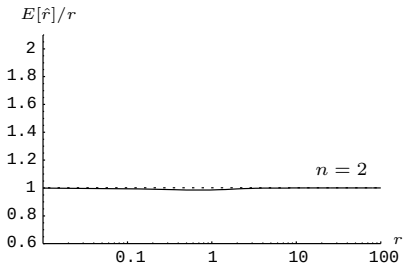

Fig. 12. Bias of the estimator with the new `Estimate()` function

```
Estimate()
   X' = (X-1) - X/(N*log(1-X/N));
   return X'/T;
```

Fig. 13. New `Estimate()` function that reduces the bias

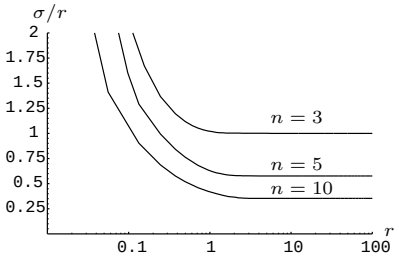

Fig. 14. Statistical variation of the new estimator over r

but when $n = 2$, the estimated frequency ratio is twice as big as the actual ratio ($E[\hat{r}]/r \approx 2$) when $r > 5$. In fact, in Appendix D, we prove that the $E[\hat{r}]/r = \frac{n}{n-1}$ when r is large, and $E[\hat{r}]/r = n \log(\frac{n}{n-1})$ when r is close to zero. For example, when $n = 2$, $E[\hat{r}]/r$ converges to $\frac{2}{2-1} = 2$ as r increases, and $E[\hat{r}]/r$ converges to $2 \log(\frac{2}{2-1}) = \log 4$ as r approaches 0. Based on this analysis, we propose to modify the `Estimate()` function to the one shown in Figure 13 so that we can remove the bias from the estimator.[5] For detailed analysis of the bias and the derivation of the new estimator, see Appendix D.

To show that our new estimator is practically unbiased, we plot the graph of $E[\hat{r}]/r$ for the new `Estimate()` function in Figure 12. The axes in the graph are the same as in Figure 11. Clearly, the estimator is practically unbiased. Even when $n = 2$, $E[\hat{r}]/r$ is very close to 1 (the bias is less than 2% for any r value.). We show the graph only for $n = 2$, because the graphs for other n values essentially overlap with that of $n = 2$.

While we derived the new `Estimate()` based on the analysis of regular access cases, note that the new `Estimate()` function does not require that access be regular. In fact, through multiple simulations, we have experimentally verified that the new function still gives negligible bias even when access is irregular. We illustrate the usage of this new estimator through the following example.

[5]The function $(X-1) - X/(N \log(1 - X/N))$ is not defined when $X = 0$, but we use $\lim_{X \to 0}[(X - 1) - X/(N \log(1 - X/N))] = 0$ as its value when $X = 0$. In short, we assume `X'` $= 0$ when `X` $= 0$.

EXAMPLE 6 A crawler wants to estimate the change frequency of a page by visiting it 5 times. However, the crawler cannot access the page more than once every month, because the site administrator does not allow more frequent crawls. Fortunately, the site provides the last modified date whenever the crawler accesses the page.

To show the improvement, let us assume that the page changes, say, once every week and we crawl the page once every month. Then, without the last modified date, the bias is 43% on average ($E[\hat{r}]/r \approx 0.57$), while we can practically eliminate the bias when we use the last modified date. (The bias is less than 0.1%.) □

Finally in Figure 14, we show the statistical variation σ/r of the new estimator, for various n. The horizontal axis in the graph is the frequency ratio r, and the vertical axis is the statistical variation σ/r. We can see that as n increases, the variation (or the standard deviation) gets smaller.

6. EXPERIMENTS

In this section we study the effectiveness of our estimator through various experiments. First in Section 6.1, we experimentally show that our estimator performs well even if the elements do *not* follow the Poisson model. In Section 6.2 we study the improvement when we use the last-modification date for frequency estimation. In Section 6.3 we compare the naive estimator (Section 4.1) with ours (Section 4.2) using *real* Web data. This experiment shows that our estimator is much more effective than the naive one: In 84% of the cases, our estimator predicted more "accurate" change frequency than the naive one. Finally in Section 6.4, we quantify the improvement a crawler may achieve when it uses our improved estimator. As we will see, our experiments suggest that a crawler may improve its effectiveness by 35% by using our estimator.

6.1 Non-Poisson model

Although the experiments in the existing literature suggest that a Poisson change distribution (that we have used so far) is a good approximate model for Web page changes [Brewington and Cybenko 2000a; 2000b; Cho and Garcia-Molina 2000a], it is interesting to study what happens to our estimators if the distribution were "not quite Poisson." One simple way to model deviations from Poisson is to change the exponential inter-arrival distribution to a more general *gamma distribution*.

A gamma distribution has two parameters, α and λ. When $\alpha = 1$, the gamma distribution becomes an exponential distribution with change rate λ, and we have a Poisson change process. If $\alpha < 1$, small change intervals become more frequent. As α grows beyond 1, small intervals become rare, and the distributions starts approximating a normal one. Thus, we can capture a wide range of behaviors by varying the α around its initial value of 1.

Through simulations, we experimentally measured the biases of the naive estimator (Section 4.1) and our proposed estimator (Section 4.2), and we show the result in Figure 15. This simulation was done on synthetic data, where the change intervals followed gamma distributions of various α values. For the simulation, we let the element change once every second on average, and we accessed the element every second. In the figure, the horizontal axis shows the α values that we used for

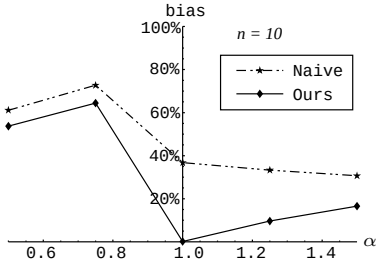

Fig. 15. Bias of the naive and our estimators for a gamma distribution

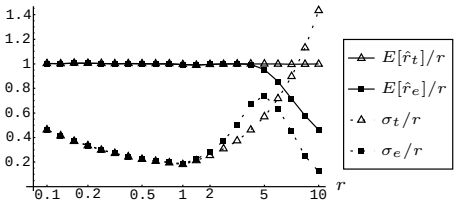

Fig. 16. Comparison of $\hat{r}_e$ and $\hat{r}_t$ for $n = 50$

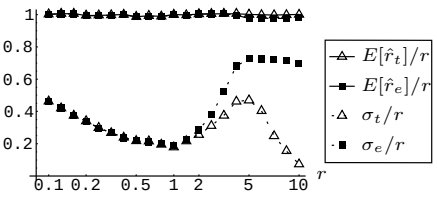

Fig. 17. Comparison of $\hat{r}_e$ and $\hat{r}_t$ for varying n values

the simulation.

From the graph, we can see that the bias of our new estimator is consistently smaller than that of the naive estimator. When $\alpha = 1$, or when the element follows a Poisson process, the bias of our estimator is 0, while the bias of the naive estimator is around 37%. Even when α is not equal to 1, our estimator still has smaller bias than that of the naive one. This graph suggests that our estimator is significantly better than the naive one, even if the change distribution is not quite a Poisson distribution.

6.2 Improvement from last modification date

We now study the improvement we can get when we use the last modification date of a page to estimate change frequency. To study the improvement, we ran multiple simulations assuming that page changes follow a Poisson process. From the simulation results we compared the bias and efficiency of the estimator in Section 4.2 (the estimator that uses page change information only) and in Section 5 (the estimator that uses the last-modification date). We refer to the estimator in Section 4.2 as $\hat{r}_e$ and the estimator in Section 5 as $\hat{r}_t$.

Figure 16 shows a set of results from these experiments. For these experiments, we assumed that we estimate the change frequency of a page after 50 visits ($n = 50$). The horizontal axis is logarithmic and represents the frequency ratio r used for a particular experiment. The solid lines show $E[\hat{r}]/r$, the average frequency ratio estimate over the actual frequency ratio. An unbiased estimator will be a straight line at 1. From the graph it is clear that $\hat{r}_e$ is not very useful when r is large ($r > 5$, when the access frequency is smaller than the change frequency). In contrast, the estimator $\hat{r}_t$ shows no bias for the whole range of r values.

The dotted lines in the graph show their efficiency, σ/r (the standard deviation

of an estimator over the actual frequency ratio). For most r values ($r < 5$), we can see that the estimator $\hat{r}_t$ shows similar or better efficiency (smaller σ/r value) than the estimator $\hat{r}_e$. While $\hat{r}_e$ seems more efficient for large r values ($r > 7$), note that $\hat{r}_e$ is not very useful in this range because of its large bias. The result simply means that $\hat{r}_e$ consistently gives wrong frequency estimates in this range.

To compare the efficiency of $\hat{r}_e$ and $\hat{r}_t$ when r is large and when $\hat{r}_e$ is unbiased (thus useful), in another set of experiments we increased the n values (number of accesses) for large r, so that the bias of $\hat{r}_e$ is reduced to less than 5%. (We remind the reader that the bias of $\hat{r}_e$ decreases when n increases.) For example, when $r = 5$, both $\hat{r}_t$ and $\hat{r}_e$ have less than 5% bias if $n = 160$ and when $r = 6$ we get less than 5% bias if $n = 500$. After making the bias of both estimators less than 5% by adjusting n, we then compared the efficiency of the estimators. Figure 17 shows the result from these experiments. The axes in the graph are the same as Figure 16. The bias of $\hat{r}_e$ and $\hat{r}_t$ are both close to a straight line because we used larger n for larger r. From the graph, it is clear that $\hat{r}_t$ is more efficient than $\hat{r}_e$ when their bias is comparable, i.e., σ_t/r is consistently smaller than σ_e/r for all r values.

6.3 Effectiveness of estimators for real Web data

In this subsection, we compare the effectiveness of our proposed estimator (Section 4.2) and that of the naive estimator (Section 4.1), using *real* change data collected from the Web. Before the comparison, we first describe how the data was obtained.

The change data was collected for a period of 4 months (from February 17, 1999 until June 24, 1999) from 720,000 Web pages on 270 sites. During this period, the pages were downloaded *every day*, so that we could detect daily changes to the pages. From this daily download, we could tell whether a page had changed or not on a particular day and obtain an accurate daily change history of 720,000 pages.

We note that the 270 sites were *not* randomly selected. Instead, we selected the most "popular" 270 Web sites, based on the snapshot of Web pages storied in our repository (25 million HTML pages at that time). To measure popularity, we counted how many pages in our repository had links to each site, and we used the count as the popularity measure of a site. From each site selected this way, we then downloaded about 3,000 pages. Therefore, the results in our experiment are biased toward more "popular" sites, but we believe that most people, thus most crawlers, are interested in these "popular" pages rather than a random Web page.

To compare the naive and our estimators, it is important to know the *actual* change frequencies of the pages: We need to tell which estimator predicts a closer frequency to the actual change frequency. However, it is not possible to obtain the *actual* change frequency of a page from our daily change data, because we do not know exactly how many times each page changed during a day. We may have missed some changes if more than one change occurred per a day.

To address this problem, we used the following method: First, we identified the pages for which we monitored "most" of the changes. That is, we selected only the pages that changed *less than once in three days*, because we would probably have missed many changes if a page changed more often. Also, we filtered out the pages that changed *less than 3 times* during our monitoring period, because we may have

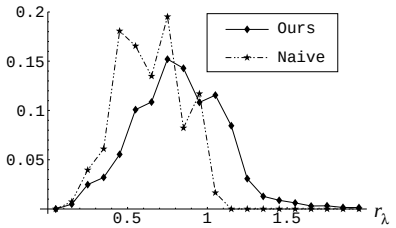

Fig. 18. Comparison of the naive estimator and ours

Table I. Total number of changes detected for each policy

policy	*Changes detected*	*Percentage over uniform*
Uniform	2,147,589	100%
Naive	4,145,581	193%
Ours	4,892,116	228%

not monitored the page long enough if it changed less often.[6] Then for the selected pages, we assumed that we did *not* miss any of their changes and we could estimate their actual frequencies by X/T (X: number of changes detected, T: monitoring period). We refer to this value as a *projected change frequency*.

On these selected pages we then ran a *simulated* crawler, which visited each page only *once every week*. Therefore, the crawler had *less* change information than our original dataset. Based on this limited information, the crawler estimated change frequency and we compared the estimates to the projected change frequency.

We emphasize that the simulated crawler did not actually crawl pages. Instead, the simulated crawler was run on the change data collected, so the projected change frequency and the crawler's estimated change frequency are based on the same dataset (The crawler simply had less information than our dataset). Therefore, we believe that the better estimator would predicts a frequency close to the projected change frequency.

From this comparison, we could observe the following:

— For 83% of pages, our proposed estimator predicted a value closer to the projected change frequency than the naive one. The naive estimator was "better" for less than 17% pages.

— Assuming that the projected change frequency is the actual change frequency, our estimator showed about 15% bias on average over all pages, while the naive estimator showed more than 35% bias. Clearly, this result shows that our proposed estimator is significantly more effective than the naive one. We can decrease the bias by half by using our estimator.

In Figure 18, we show more detailed results from this experiment. The horizontal axis in the graph shows the ratio of the *estimated* change frequency to the *projected* change frequency (r_λ) and the vertical axis shows the faction of pages that had the given ratio. Therefore, for the pages with $r_\lambda < 1$, the estimate was smaller than the projected frequency, and for the pages with $r_\lambda > 1$, the estimate was larger than the projected frequency. Assuming that the projected frequency is the actual

[6]We also used less/more stringent range for the selection, and the results were similar.

frequency, a better estimator is the one centered around $r_\lambda = 1$. We can clearly see that our proposed estimator is better than the naive one. The distribution of the naive estimator is skewed to the left quite significantly, while the distribution of our estimator is more centered around 1.

6.4 Application to a Web crawler

Our results show that our proposed estimator gives more accurate change frequency than the naive one. Now we study how much improvement an application may achieve by using our estimator. To illustrate this point, we consider a Web crawler as an example.

Typically, a Web crawler revisits all pages at the *same* frequency, regardless of how often they change. Instead, a crawler may revisit pages at different frequencies depending on their estimated change frequency. In this subsection, we compare the following revisit policies:

— **Uniform policy:** A crawler revisits all pages at the frequency of once every week. The crawler does not try to estimate how often a page changes.

— **Naive policy:** In the first 5 visits, a crawler visits each page at the frequency of once every week. After the 5 visits, the crawler estimates the change frequencies of the pages using the *naive estimator* (Section 4.1). Based on these estimates, the crawler adjusts revisit frequencies for the remaining visits.
Note that there exist multiple ways to adjust revisit frequency. For example, the crawler may visit a page proportionally more often based on its change frequency ($f \propto \lambda$, where f is revisit frequency, and λ is change frequency). In our experiment, we set the revisit frequency f to be proportional to the square root of its change frequency λ ($f \propto \sqrt{\lambda}$), because this policy gave the best result.[7]

— **Our policy:** The crawler uses our proposed estimator (Section 4.2) to estimate change frequency. Other than this fact, the crawler uses the same policy as the naive policy.

In our experiments, we ran a *simulated* crawler on the change history data described in Section 6.3 for each policy. For fair comparison, we made sure that the *average* revisit frequency *over all pages* was equal to once a week under any policy. That is, the crawler used the same total download/revisit resources, but allocated these resources differently under different policies. Since we have the change history of 720,000 pages for about 3 months,[8] and since the simulated crawler visited pages once every week on average, the crawler visited pages $720,000 \times 13$ weeks $\approx 9,260,000$ times in total.

Out of these 9.2M visits, we counted how many times the crawler detected changes, and we report the results in Table I. The second column shows the total number of changes detected under each policy, and the third column shows the percentage improvement over the uniform policy. Note that the best policy is the one that detected the highest number of changes from the same number of visits.

[7] We can analyze our quality metric (described shortly) using the techniques in [Cho and Garcia-Molina 2000b]. This analysis shows that $f \propto \sqrt{\lambda}$ policy gives close to optimal result.
[8] While we monitored pages for 4 months, some pages were deleted during our experiment, so each page was monitored for 3 months on average

That is, we use the *total number of changes detected* as our quality metric. From these results, we can observe the following:

— A crawler can significantly improve its effectiveness by adjusting revisit frequency. For example, the crawler detected 2 times more changes when it used the naive policy than the uniform policy.

— Our proposed estimator makes the crawler much more effective than the naive one. Compared to the naive policy, our policy detected 35% more changes!

7. CONCLUSION

In this paper, we studied the problem of estimating the change frequency of an element, when we do *not* have the complete change history. In this case, we cannot simply divide the detected number of changes by time because this estimation leads to significant bias due to undetected changes. By careful analysis we designed estimators with low bias and with high consistency.

We also studied how effective our estimators are using real Web change data. Our experimental results strongly indicate that our estimator predicts the change frequency much more accurately than existing ones and improves the effectiveness of a crawler significantly. We also verified that our estimator works well even if the change model is not "quite Poisson."

7.1 Future Work

We briefly discuss some venues for future work. In this paper we assumed that we pick an estimator *before* we start an experiment and use the method throughout the experiment. But in certain cases, we may need to dynamically adjust our estimation method, depending on what we detect during the experiment.

(1) **Adaptive scheme:** Even if we initially decide on a certain access frequency, we may want to adjust it during the experiment, when the estimated change frequency is very different from our initial guess. Then exactly when and how much should we adjust the access frequency?

EXAMPLE 7 Initially, we guessed that a page changes once every week and started visiting the page every 10 days. In the first 4 accesses, however, we detected 4 changes, which signals that the page may change much more frequently than we initially guessed.
In this scenario, should we increase the access frequency immediately or should we wait a bit longer until we collect more evidence? When we access the page less often than it changes, we need a large sample size to get an unbiased result, so it might be good to adjust the access frequency immediately. On the other hand, it is also possible that the page indeed changes once every week on average, but it changed in the first 4 accesses by statistical variation. Then when should we adjust the change frequency to get the optimal result?

(2) **Changing λ:** In this paper we also assumed that the change frequency λ of an element is stationary (i.e., does not change). This assumption may not be valid in certain cases, and we may need to test whether λ changes or not. There has been work on how we can verify whether a Poisson process is stationary and, if

not, how we can estimate the change rate of λ [Yacout and Chang 1996; Canavos 1972]. However, this work assumes that the complete change history of an element is available, so careful study is necessary to apply it to incomplete change history.

Finally, in this paper we mainly focused on the problem of change-frequency estimation, assuming that other algorithms [Cho and Garcia-Molina 2000b; Edwards et al. 2001; Coffman, Jr. et al. 1998] can use the change frequency estimate to determine the optimal revisit frequency. Instead, we may directly determine the optimal revisit frequency based on *all change history* that we have observed so far. Since this approach removes one level of indirection (i.e., estimating the change frequency), we may be able to get a better revisit frequency.[9]

A. PROOFS FOR COROLLARIES

PROOF OF COROLLARY 4.2.

$$\begin{aligned}
\sigma[\hat{r}] &= \sqrt{E[\hat{r}^2] - E[\hat{r}]^2} \\
&= \sqrt{\left(\sum_{m=0}^{n} \left(\frac{m}{n}\right)^2 \Pr\left\{\hat{r} = \frac{m}{n}\right\}\right) - \left(\sum_{m=0}^{n} \left(\frac{m}{n}\right) \Pr\left\{\hat{r} = \frac{m}{n}\right\}\right)^2} \\
&\qquad \left(\text{where } \Pr\left\{\hat{r} = \frac{m}{n}\right\} = \binom{n}{m}(1-e^{-r})^m e^{-r(n-m)} \text{ from Theorem 4.1}\right) \\
&= \sqrt{\left[(1-e^{-r})\left\{\frac{e^{-r}}{n} + (1-e^{-r})\right\}\right] - \left[1-e^{-r}\right]^2} \\
&= \sqrt{e^{-r}(1-e^{-r})/n}
\end{aligned}$$

PROOF OF COROLLARY 4.3. From the definition of $\bar{X}$,

$$\Pr\{\bar{X} = i\} = \Pr\{X = n - i\} = \binom{n}{i}(1-e^{-r})^{n-i}(e^{-r})^i.$$

Then,

$$\mathrm{E}\left[-\log\left(\frac{\bar{X}+0.5}{n+0.5}\right)\right] = -\sum_{i=0}^{n} \log\left(\frac{i+0.5}{n+0.5}\right)\binom{n}{i}(1-e^{-r})^{n-i}(e^{-r})^i.$$

B. AVOIDING THE SINGULARITY OF THE ESTIMATOR $-\mathrm{LOG}(\bar{X}/N)$

The estimator $-\log(\bar{X}/n)$ has a singularity when $\bar{X} = 0$, because $\log(0/n) = \infty$. In this section, we study how we can avoid the singularity.

Note that the singularity arises because we pass the number 0 as the parameter of a logarithmic function. Intuitively, we can avoid the singularity if we increase $\bar{X}$ slightly when $\bar{X} = 0$, so that the logarithmic function does not get 0 even when $\bar{X} = 0$. In general, we may avoid the singularity if we add small numbers a and b (> 0) to the numerator and the denominator of the estimator, so that the estimator is $-\log(\frac{\bar{X}+a}{n+b})$. Note that when $\bar{X} = 0$, $-\log(\frac{\bar{X}+a}{n+b}) = -\log(\frac{a}{n+b}) \neq \infty$ if $a > 0$.

[9] We thank an anonymous reviewer for suggesting this idea.

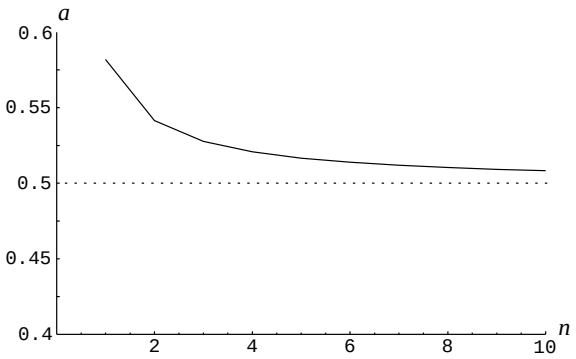

Fig. 19. The a values which satisfy the equation $n \log(\frac{n+a}{n-1+a}) = 1$

Then what value should we use for a and b? To answer this question, we use the fact that we want the expected value, $E[\hat{r}]$, to be as close to r as possible. As we showed in Section 4.2, the expected value is

$$E[\hat{r}] = \mathrm{E}\left[-\log\left(\frac{\bar{X}+a}{n+b}\right)\right] = -\sum_{i=0}^{n} \log\left(\frac{i+a}{n+b}\right)\binom{n}{i}(1-e^{-r})^{n-i}(e^{-r})^{i},$$

which can be approximated to

$$E[\hat{r}] \approx \left[-\log\left(\frac{n+a}{n+b}\right)\right] + \left[n\log\left(\frac{n+a}{n-1+b}\right)\right] r + \ldots$$

by Taylor expansion [Thomas, Jr. 1969]. Note that we can make the above equation to $E[\hat{r}] \approx r + \ldots$, by setting the constant term $-\log(\frac{n+a}{n+b}) = 0$, and the factor of the r term, $n\log(\frac{n+a}{n-1+b}) = 1$.

From the equation $-\log(\frac{n+a}{n+b}) = 0$, we get $a = b$, and from $n\log(\frac{n+a}{n-1+a}) = 1$, we get the graph of Figure 19. In the graph, the horizontal axis shows the value of n and the vertical axis shows the value of a which satisfies the equation $\log(\frac{n+a}{n-1+a}) = 1$ for a given n. We can see that the value of a converges to 0.5 as n increases and that a is close to 0.5 even when n is small. Therefore, we can conclude that we can minimize the bias by setting $a = b = 0.5$.

Alternatively, we may derive an estimator that avoids the singularity through Bayesian estimation [Bernardo and Smith 1994; Lee 1997].[10] In Bayesian estimation, we model the estimated λ of a page as a *random variable* and assign a probability distribution $\Pr(\lambda)$ to the estimated λ. Before we start visiting a page we assume a certain *prior distribution* for $\Pr(\lambda)$, say a uniform distribution, based on our knowledge on the λ value. After a visit, we then update the distribution based on the observed change/non-change (i.e., the outcome of random variable X) using the following formula:

$$\Pr(\lambda|X) = \frac{\Pr(X|\lambda)\Pr(\lambda)}{\Pr(X)}$$

[10]We thank an anonymous reviewer for suggesting this alternative method.

This new distribution $\Pr(\lambda|X)$ is called a *posteriori distribution*. Note that we can compute $\Pr(X|\lambda)$ and $\Pr(X)$ on the right side of formula using the Poisson model, and $\Pr(\lambda)$ is the assumed prior distribution. In the successive visits, the computed posteriori distribution then works as the prior distribution for the visit and we compute a new posteriori distribution based on observed changes.

Notice that after every visit we have a probability distribution (not a single number) associated with the λ of the page. To obtain a single number for λ from the distribution, we can either use the *maximum a posteriori estimator* or the *posterior expectation* [Bernardo and Smith 1994; Lee 1997].

To simplify our derivation, we assume that our goal is to estimate $\theta = 1 - e^{-\lambda/f}$ (thus indirectly λ) and use the beta function

$$\Pr(\theta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1}$$

as the prior distribution of θ.[11] Under these assumptions, the posterior expectation estimator of θ becomes $\frac{X+\alpha}{n+\alpha+\beta}$. If we assume a uniform prior for θ (i.e., $\alpha = \beta = 1$), then the estimator becomes $\frac{X+1}{n+2}$. Given that $\theta = 1 - e^{-r} = \frac{X+1}{n+2}$,

$$\hat{r} = -\log\left(\frac{n-X+1}{n+2}\right).$$

This estimator also avoids the singularity of

$$\hat{r} = -\log\left(\frac{n-X}{n}\right).$$

C. ESTIMATOR FOR AN IRREGULAR ACCESS CASE

For our analysis, we assume that t_{ci} represents the interval in which we detected the $i$the change, and t_{uj} represents the jth interval in which we did not detect a change. Also m represents the total number of changes we detected out of n accesses.

Under a Poisson process, the probability that the element changes after interval t_{ci} is $1 - e^{-\lambda t_{ci}}$. Also,the probability that the element does not change after the interval t_{uj} is $e^{-\lambda t_{uj}}$. Therefore, the probability that the observed events happen is $\prod_{i=1}^{m}(1-e^{-\lambda t_{ci}}) \cdot \prod_{j=1}^{n-m} e^{-\lambda t_{uj}}$. We need to find the λ that maximizes this probability.

Since logarithm is a monotonic function, we can take the logarithm of the above formula and find the λ that minimizes the logarithm. That is, λ should minimize

$$\log\left(\prod_{i=1}^{m}(1-e^{-\lambda t_{ci}})\prod_{j=1}^{n-m} e^{-\lambda t_{uj}}\right) = \sum_{i=1}^{m}\log(1-e^{-\lambda t_{ci}}) - \sum_{j=1}^{n-m}\lambda t_{uj}.$$

To minimize this,

$$\frac{d}{d\lambda}\left(\sum_{i=1}^{m}\log(1-e^{-\lambda t_{ci}}) - \sum_{j=1}^{n-m}\lambda t_{uj}\right) = \sum_{i=1}^{m}\frac{t_{ci}}{e^{\lambda\, t_{ci}}-1} - \sum_{j=1}^{n-m} t_{uj} = 0.$$

[11] The beta function is commonly used when the observed event is binary.

That is, λ should satisfy

$$\sum_{i=1}^{m} \frac{t_{ci}}{e^{\lambda t_{ci}} - 1} = \sum_{j=1}^{n-m} t_{uj}.$$

D. BIAS OF THE ESTIMATOR IN FIGURE 10

In this section, we analyze the bias of the estimator described in Figure 10. For our analysis, we assume that we access the element at a regular interval $I(= 1/f)$, and we compute the bias of the *frequency ratio* $\hat{r} = \hat{\lambda}/f$, where $\hat{\lambda}$ is the estimator described in Figure 10.

The following lemma provides the formula to compute the bias of the estimator.

LEMMA D.1 *The bias of the estimator $\hat{r} = \hat{\lambda}/f$ ($\hat{\lambda}$ is the estimator of Figure 10) is:*

$$\frac{E[\hat{r}]}{r} = \sum_{k=0}^{n} \left[\frac{\Pr\{X = k\}}{r} \int_{(n-k)I}^{nI} \left(\frac{k}{ft}\right) \Pr\{\mathsf{T} = t \mid X = k\}\, dt \right] \tag{3}$$

Here, $\Pr\{X = k\}$ is the probability that the variable X *takes a value k, and $\Pr\{\mathsf{T} = t \mid X = k\}$ is the probability that the variable* T *takes a value t when* X $= k$. *We assume we access the element n times.* □

PROOF. The proof is straightforward. We can compute the expected value by summing all possible $\frac{X}{T}$ values multiplied by their probabilities. In the proof of Theorem D.2, we will show how we can compute $\Pr\{X = k\}$ and $\Pr\{\mathsf{T} = t \mid X = k\}$.

We now compute the closed form of the bias of Figure 10 in the following theorem. This theorem will give us an important idea on how we can eliminate the bias.

THEOREM D.2 *When $r \to \infty$, the only remaining term in the summation of Equation 3 is when $k = n$, and $E[\hat{r}]/r$ becomes $\frac{n}{n-1}$.*

$$\lim_{r\to\infty} \frac{E[\hat{r}]}{r} = \lim_{r\to\infty} \frac{\Pr\{X = n\}}{r} \int_{0}^{nI} \left(\frac{n}{ft}\right) \Pr\{\mathsf{T} = t \mid X = n\}\, dt = \frac{n}{n-1}$$

Also, when $r \to 0$, the only remaining term in the equation is when $k = 1$, and $E[\hat{r}]/r$ becomes $n \log(\frac{n}{n-1})$.

$$\lim_{r\to 0} \frac{E[\hat{r}]}{r} = \lim_{r\to 0} \frac{\Pr\{X = 1\}}{r} \int_{(n-1)I}^{nI} \left(\frac{1}{ft}\right) \Pr\{\mathsf{T} = t \mid X = 1\}\, dt = n \log\left(\frac{n}{n-1}\right)$$

□

PROOF. We first show how we can compute $\Pr\{X = k\}$ and $\Pr\{\mathsf{T} = t \mid X = k\}$. The variable X is equal to k when the element changed in k accesses. Since the element may change at each access with probability $1 - e^{-r}$ ($r = \lambda/f$, r: frequency ratio),

$$\Pr\{X = k\} = \binom{n}{k} (1 - e^{-r})^k (e^{-r})^{n-k}$$

Now we compute the probability that $\mathcal{T} = t$ $(0 \le t \le kI)$ when $X = k$. If we use t_i to represent the *time added to* $\mathcal{T}$ *at the* i*th access*, $\mathcal{T}$ can be expressed as

$$\mathcal{T} = \sum_{i=1}^{n} t_i$$

Since the element did not change in $(n-k)$ accesses when $X = k$, we added $(n-k)$ times of I to $\mathcal{T}$. Then

$$\mathcal{T} = (n-k)I + \sum_{i=1}^{k} t_{c_i}$$

where c_i is the ith access in which the element changed. Without losing generality, we can assume that the element changed only in the first k accesses. Then

$$\mathcal{T} = (n-k)I + \sum_{i=1}^{k} t_i \tag{4}$$

In those changed accesses, the probability that $t_i = t$ is

$$\Pr\{t_i = t\} = \begin{cases} \dfrac{\lambda e^{-\lambda t}}{1 - e^{-r}} & \text{if } 0 \le t \le I \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

because the element follows a Poisson process. From Equations 4 and 5, we can compute the $\Pr\{\mathcal{T} = t \mid X = k\}$ by repeatedly applying the *methods of transformations* [Wackerly et al. 1997]. For example, when $k = 3$

$$\Pr\{\mathcal{T} = (n-k)I + t \mid X = k\} = \begin{cases} \dfrac{(\lambda t)^2 \lambda e^{-\lambda t}}{2(1 - e^{-r})^3} & \text{for } 0 \le t < I \\ \dfrac{[6\lambda t - 2(\lambda t)^2 - 3]\ \lambda e^{-\lambda t}}{2(1 - e^{-r})^3} & \text{for } I \le t < 2I \\ \dfrac{(\lambda t - 3)^2 \lambda e^{-\lambda t}}{2(1 - e^{-r})^3} & \text{for } 2I \le t < 3I \\ 0 & \text{otherwise} \end{cases}$$

Now we study the limit values of Equation 3. When $r \to \infty$,

$$\lim_{r \to \infty} \frac{\Pr\{X = k\}}{r} = \begin{cases} 0 & \text{if } \ k = 0, 1, \ldots, n-1 \\ 1 & \text{if } \ k = n \end{cases}$$

Also when $r \to 0$,

$$\lim_{r \to 0} \frac{\Pr\{X = k\}}{r} = \begin{cases} n & \text{if } \ k = 1 \\ 0 & \text{if } \ k = 2, \ldots, n \end{cases}$$

Then

$$\begin{aligned}
\lim_{r\to\infty}\frac{E[\hat{r}]}{r} &= \lim_{r\to\infty}\sum_{k=0}^{n}\left(\frac{\Pr\{X=k\}}{r}\int_{(n-k)I}^{nI}\left(\frac{k}{ft}\right)\Pr\{\mathcal{T}=t\mid X=k\}\,dt\right)\\
&= \lim_{r\to\infty}\int_{0}^{nI}\left(\frac{n}{ft}\right)\Pr\{\mathcal{T}=t\mid X=n\}\,dt\\
&= \lim_{r\to\infty}\int_{0}^{nI}\left(\frac{n}{ft}\right)\frac{(\lambda t)^{n-1}\lambda e^{-\lambda t}}{(n-1)!(1-e^{-r})^{n}}\,dt\\
&= \frac{n}{n-1}\\
\lim_{r\to 0}\frac{E[\hat{r}]}{r} &= \lim_{r\to 0}\sum_{k=0}^{n}\left(\frac{\Pr\{X=k\}}{r}\int_{(n-k)I}^{nI}\left(\frac{k}{ft}\right)\Pr\{\mathcal{T}=t\mid X=k\}\,dt\right)\\
&= n\lim_{r\to 0}\int_{(n-1)I}^{nI}\left(\frac{1}{ft}\right)\Pr\{\mathcal{T}=t\mid X=1\}\,dt\\
&= n\lim_{r\to 0}\int_{(n-1)I}^{nI}\left(\frac{1}{ft}\right)\frac{e^{(n-1)I}\lambda e^{-\lambda t}}{1-e^{-r}}\,dt\\
&= n\log\left(\frac{n}{n-1}\right)
\end{aligned}$$

Informally, we may explain the meaning of the theorem as follows:

> When r is very large (i.e., when the element changes much more often than we access it), we are very likely to detect n changes ($X = n$ with high probability), and the bias of the estimator is $\frac{n}{n-1}$. When r is very small (i.e., when we access the element much more often than it changes), we are very likely to detect either 0 or 1 change (X will be either 0 or 1 with high probability), and the bias is $n\log(n/(n-1))$ when $X = 1$.

We can use this result to design an estimator that eliminates the bias. Assume that $X = n$ after n accesses. Then it strongly indicates that r is very large, in which case the bias is $\frac{n}{n-1}$. To avoid this bias, we may divide the original estimator X/T by $\frac{n}{n-1}$ and use $\frac{X}{T}/\frac{n}{n-1} = \frac{n-1}{T}$ as our new estimator in this case. That is, when $X = n$ we may want to use $(X-1)/T$ as our estimator, instead of X/T. Also, assume that $X = 1$ after n accesses. Then it strongly indicates that r is very small, in which case the bias is $n\log(n/(n-1))$. To avoid this bias, we may use $\left(\frac{X}{T}\right) / n\log(n/(n-1)) = \frac{1}{T}\frac{X}{n\log(n/(n-X))}$ as our estimator when $X = 1$. In general, if we use the estimator X'/T where

$$X' = (X-1) - \frac{X}{n\log(1-X/n)}$$

we can avoid the bias both when $X = n$ and $X = 1$: $X' = n-1$ when $X = n$, and $X' = \frac{1}{n\log(n/(n-1))}$ when $X = 1$.[12]

[12]The function $(X-1) - X/(n\log(1-X/n))$ is not defined when $X = 0$ and $X = n$. However, we can use $\lim_{X\to 0}[(X-1) - X/(n\log(1-X/n))] = 0$ and $\lim_{X\to n}[(X-1) - X/(n\log(1-X/n))] =$

REFERENCES

Baentsch, M., Baum, L., Molter, G., Rothkugel, S., and Sturm, P. 1997. World Wide Web caching: The application-level view of the internet. *IEEE Communications Magazine 35,* 6 (June), 170–178.

Bernardo, J. and Smith, A. 1994. *Bayesian Theory*. Wiley.

Brewington, B. E. and Cybenko, G. 2000a. How dynamic is the web. In *Proceedings of the 9th World-Wide Web Conference*.

Brewington, B. E. and Cybenko, G. 2000b. Keeping up with the changing web. *IEEE Computer 33,* 5 (May), 52–58.

Canavos, G. 1972. A bayesian approach to parameter and reliability estimation in the Poisson distribution. *IEEE Transactions on Reliability R21*, 52–56.

Cho, J. and Garcia-Molina, H. 2000a. The evolution of the web and implications for an incremental crawler. In *Proceedings of the 26th International Conference on Very Large Databases*.

Cho, J. and Garcia-Molina, H. 2000b. Synchronizing a database to improve freshness. In *Proceedings of the 2000 ACM SIGMOD*.

Cho, J. and Garcia-Molina, H. 2002. Estimating frequency of change. Tech. rep., University of California, Los Angeles.

Coffman, Jr., E., Liu, Z., and Weber, R. R. 1998. Optimal robot scheduling for web search engines. *Journal of Scheduling 1,* 1 (June), 15–29.

Courant, R. and David, H. 1989. *Methods of mathematical physics*, 1st ed. John Wiley & Sons.

Douglis, F., Feldmann, A., Krishnamurthy, B., and Mogul, J. 1999. Rate of change and other metrics: a live study of the world wide web. In *USENIX Symposium on Internetworking Technologies and Systems*.

Edwards, J., McCurley, K., and Tomlin, J. 2001. An adaptive model for optimizing performance of an incremental web crawler. In *Proceedings of the 10th World-Wide Web Conference*. Hong Kong, China.

Gwertzman, J. and Seltzer, M. 1996. World-Wide Web cache consistency. In *Proceedings of USENIX 1996 Annual Technical Conference*.

Hammer, J., Garcia-Molina, H., Widom, J., Labio, W. J., and Zhuge, Y. 1995. The Stanford data warehousing project. *IEEE Data Engineering Bulletin 18,* 2 (June), 40–47.

Harinarayan, V., Rajaraman, A., and Ullman, J. D. 1996. Implementing data cubes efficiently. In *Proceedings of the 1996 ACM SIGMOD*.

Lee, P. M. 1997. *Bayesian Statistics: An Introduction*, 2nd ed. Arnold.

Matloff, N. 2002. Estimation of internet file-access/modification rates from incomplete data. Tech. rep., University of California, Davis.

Misra, P. and Sorenson, H. 1975. Parameter estimation in Poisson processes. *IEEE Transactions on Information Theory IT-21*, 87–90.

Snyder, D. L. 1975. *Random Point Processes*. Wiley.

Taylor, H. M. and Karlin, S. 1998. *An Introduction To Stochastic Modeling*, 3rd ed. Academic Press.

Thomas, Jr., G. B. 1969. *Calculus and analytic geometry*, 4th ed. Addison-Wesley.

Wackerly, D. D., Mendenhall, W., and Scheaffer, R. L. 1997. *Mathematical Statistics With Applications*, 5th ed. PWS Publishing.

Wills, C. E. and Mikhailov, M. 1999. Towards a better understanding of web resources and server responses for improved caching. In *Proceedings of the 8th World-Wide Web Conference*.

Winkler, R. L. 1972. *An Introduction to Bayesian Inference and Decision*. Holt, Rinehart and Winston, Inc.

Wolman, A., Voelker, G. M., Sharma, N., Cardwell, N., Karlin, A., and Levy, H. M. 1999. On the scale and performance of cooperative web proxy caching. In *Proceedings of the 17th ACM Symposium on Operating Systems Principles (SOSP '99)*. 16–31.

$n-1$ as its value when $X=0$ and $X=n$, respectively. That is, we assume $X'=0$ when $X=0$, and $X'=n-1$ when $X=n$.

YACOUT, S. AND CHANG, Y. 1996. Using control charts for parameter estimation of a homogeneous poisson process. In *19th International Conference on Computers and Industrial Engineering*.

YU, H., BRESLAU, L., AND SHENKER, S. 1999. A scalable Web cache consistency architecture. In *Proceedings of the 1999 ACM SIGCOMM*.

ZHUGE, Y., GARCIA-MOLINA, H., HAMMER, J., AND WIDOM, J. 1995. View maintenance in a warehousing environment. In *Proceedings of the 1995 ACM SIGMOD*.

Received March 2002; revised December 2002; accepted March 2003