**Paper ID 135**

# Estimating Frequency of Change

Junghoo Cho[1] Hector Garcia-Molina
Stanford University
{cho, hector}@cs.stanford.edu

Category: Research Paper

[1]Contact Author: Junghoo Cho
Email: cho@cs.stanford.edu
Mail: P.O. Box 11549
Stanford, CA 94309
USA
Phone: (650)723-0587

# Estimating Frequency of Change

Junghoo Cho, Hector Garcia-Molina
Stanford University
{cho, hector}@cs.stanford.edu

**Abstract**

Many online data sources are updated autonomously and independently. In this paper, we make the case for estimating the change frequency of the 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. In developing the estimators, we analytically show how precise/effective the estimators are, and we show that the estimators that we propose can improve precision significantly.

## 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/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, we believe that the clients can significantly benefit by estimating the change frequency of the sources. 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/index up-to-date, the crawler periodically revisits the 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 [4], the 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 to minimize 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 a 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 frequency of change," and how we accomplish the task. However, there exist important challenges in estimating 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 interval, because the page may have changed more than once between some accesses. Then, what would be the fair estimate for the change interval? 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 [10, 13]. As far as we know, there has been no work on how to estimate the frequency based on incomplete information.

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 $(10 \text{ days})/2 = 5$ 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 a majority of 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. 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 6). In summary, our paper makes the following contributions:

- We identify the problem of estimating the frequency of change and we present a formal framework to study the problem.
- We propose several estimators that measure the frequency of change much more effectively than existing ones. 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 the analytical results that show how precise/effective our proposed estimators are.

### 1.1 Related work

The problem of estimating change frequency has been long studied in statistics community [10, 13, 9, 2]. 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.

Reference [4] studies 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, the reference 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.

In our companion paper [3] (submitted to the VLDB 2000 Experience track), we explain how a web crawler can use the techniques that we develop here. The paper compares various crawler design choices based on the statistics collected from more than half million web pages, and it proposes an architecture that employs the estimation techniques that we develop in this paper.

Many researchers studied how to build a scalable and effective web caching system, to minimize the access delay, the server load and the bandwidth usage [14, 5, 1]. 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 [6, 7, 15]. However, most of the work focused on different issues, such as minimizing the size of the view while reducing the query response time [7].

## 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 the 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 website or a single row in a database table, etc. Also the 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 the 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 minutes, 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 [10, 13]. In this paper, we do not study this case.
   - **Last date of change:** We only 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, note that 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.
    - **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 Section 4 and 5, we study the problem of *estimating* the frequency of change, and in Section 6, we study the problem of *categorizing* the frequency of change.

# 3 Preliminaries

In this section, we will review some of the basic concepts for the estimation of frequency, to help readers understand our later discussion. We also summarize experimental data that shows that web page changes follow a Poisson process. 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 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. In particular, web pages are known to follow a Poisson Process. In our companion paper [3], we trace the change history of 720,000 web page collected from 270 sites for 4 months and compare the result against what the Poisson process model predicts. Based on this comparison, we show that the Poisson process describes the changes of web pages very well. For example, Figure 1 is one of the graphs that compare the *actual* change intervals of pages against the prediction of the Poisson process model. (The graph is based on the pages that change every 10 days *on average*. We also plotted similar

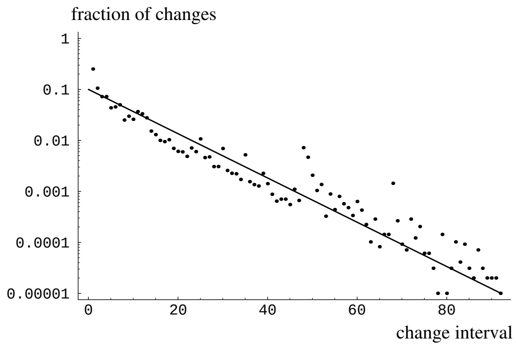


Figure 1: The actual change intervals of pages and the prediction of the Poisson process model

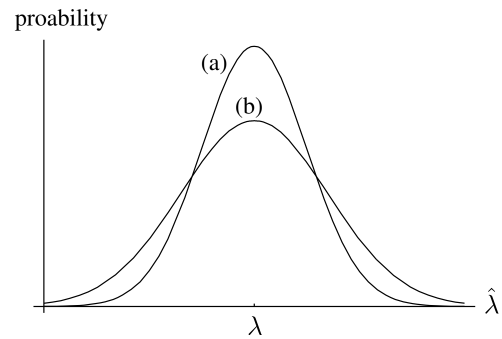


Figure 2: Two possible distributions of the estimator $\hat{\lambda}$

graphs for the pages that change at other average change interval and got similar result.) The horizontal axis represents the change interval of pages, and the vertical axis represents the fraction of changes with that given change interval. The straight line is the prediction of a Poisson process and the dots are the observed changes in the experiment. Clearly, the experimental data is clustered around the predicted line.

While the study clearly shows that web pages follow the Poisson process, it is also limited because the experiment was conducted for a limited time window (4 months) and the web pages were accessed on a daily basis. Therefore, our study does not verify the Poisson process model for the pages that change very often (more than once every day) or the pages that change very slowly (less than once every 4 months). However, we believe these pages are of low interest to most practical applications. For example, crawlers rarely can access pages more than once every day,[1] so a crawler does not care too much whether a page changes exactly once every day or more than once every day. Also, there may exist a set of pages that change at a regular interval, which do not necessarily follow the Poisson process. However, these pages are not easy to identify when a crawler manages hundreds of millions of web pages. For this reason we believe it is safe to assume that the entire set of pages change by a random process *on average*.

Returning to the description of a Poisson process, 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* $\lambda$ has the following properties:

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

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

### 3.2 Quality of estimator

The goal of this paper is to estimate the frequency of change $\lambda$, 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

[1]Crawlers should not abuse web sites. If a crawler accesses a web site too often, the site often blocks access completely.

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 $6/10 = 0.6/\text{day}$. 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 $\lambda$ 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. **Unbiasedness:** Let us assume that the element changes at the average frequency $\lambda$, 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 $\lambda$. Mathematically, $\hat{\lambda}$ is said to be *unbiased*, when the expected value of $\hat{\lambda}$, $E[\hat{\lambda}]$, is equal to $\lambda$.

2. **Efficiency:** In Figure 2, it is clear that $\hat{\lambda}$ may take a value other than $\lambda$, even if $E[\hat{\lambda}] = \lambda$. For any estimator, the estimated value might be off from the real value $\lambda$, 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 $\lambda$, as we increase the sample size. This convergence of $\hat{\lambda}$ to $\lambda$ 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 $\lambda$ 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. In Section 4.1 we 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 the discussion, we first define some notation. We assume that we access the element at a regular interval $I$ and that we access the element $n$ times. (Estimating the change frequency for irregular accesses is a very difficult problem, and we defer the discussion to Section 6.) Also, we use $X_i$ to indicate whether the element changed or not in the $i$th 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 defined as the sum of all $X_i$'s, $X = \sum_{i=1}^{n} X_i$, and represents the total number of changes that we detected. We use $T$ to refer to the total time elapsed during our $n$ accesses. Since we access the element every $I$ time units, $T = nI = n/f$, where $f(= 1/I)$ is the frequency at which we access the element. We also assume that the changes of the element follow a Poisson process with rate $\lambda$. Then, we can define the frequency ratio $r = \lambda/f$, the ratio of the change frequency to the access frequency. When $r$ is large ($\lambda \gg f$), the element changes more often than we access it, and when $r$ is small ($\lambda \ll f$), we access the element more often than it changes.

Note that our goal is to estimate $\lambda$, given $X_i$'s and $T(= n/f)$. However, we may estimate the frequency ratio $r(= \lambda/f)$ first and estimate $\lambda$ 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 $X$ through an experiment and then use the number $X$ 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.

   We can verify this intuition by computing the expected value of $\hat{r}$, $E[\hat{r}]$, and comparing it with the actual $r$. To compute $E[\hat{r}]$, we first compute the probability $q$ that the element does not change between accesses [10].

   **Lemma 1** *Let $q$ be the probability that the element does not change during time interval $I$ $(= 1/f)$. Then*
   $$q = \Pr\{X(t+I) - X(t) = 0\} = \frac{\lambda^0 e^{-\lambda I}}{0!} = e^{-\lambda I} = e^{-\lambda/f} = e^{-r} \qquad \square$$

   By definition, $X_i$ is equal to zero when the element does not change between the $(i-1)$th and the $i$th 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

   $$X_i = \begin{cases} 1 & \text{with probability } 1-q \\ 0 & \text{with probability } q \ (= e^{-r}) \end{cases}$$

   independently from other $X_i$'s. Then from the definition of $X$, $X$ is equal to $m$, when $m$ $X_i$'s are equal to 1: $\Pr\{X = m\} = \binom{n}{m}(1-q)^m q^{n-m}$. Therefore,

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

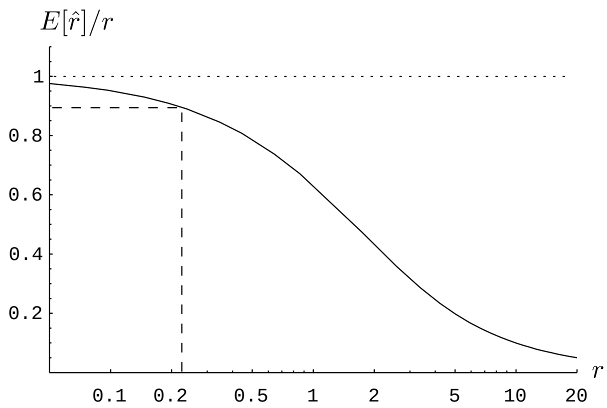


Figure 3: Bias of the intuitive estimator $\hat{r} = X/n$

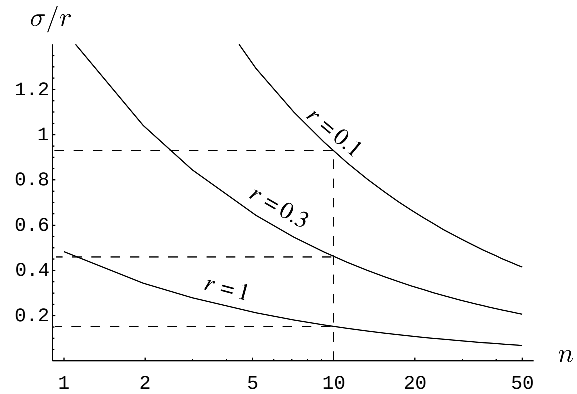


Figure 4: Statistical variation of $\hat{r} = X/n$ over $n$

Note that for $\hat{r}$ to be unbiased, $E[\hat{r}]$ should be 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 or 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 $E[\hat{r}]$ is about the same as $r$ ($E[\hat{r}]/r \approx 1$) when $r$ $(= \lambda/f)$ is small (i.e., when the element changes less often than we access it), but $E[\hat{r}]$ is significantly smaller than $r$ ($E[\hat{r}]/r \ll 1$), when $r$ is large. Intuitively, this is because we miss more changes as we access the element less often ($r = \lambda/f \gg 1$). 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 $\hat{r}$, let us compute its variance.

$$V[\hat{r}] = E[\hat{r}^2] - E[\hat{r}]^2 = e^{-r}(1 - e^{-r})/n$$

   Then, the standard deviation of $\hat{r}$ is $\quad \sigma = \sqrt{V[\hat{r}]} = \sqrt{e^{-r}(1 - e^{-r})/n}$

   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 (or access to the element) inherently induces some statistical variation.

   From the basic 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 [12]. In most applications, we want to minimize this confidence interval (whose length is proportional to $\sigma$) relative to the actual frequency ratio $r$. Therefore, we want to reduce the the ratio of the confidence interval to the frequency ratio, $\sigma/r$, 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 $\sigma/r$ decreases as $n$ increases; While we *cannot* decrease the *bias* of $\hat{r}$ by increasing the sample size, we *can* minimize the *statistical variation* (or the confidence interval) with more samples.

Also note that when $r$ is small, we need a larger sample size $n$ to get the same variation $\sigma/r$. 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 $\sigma/r$ 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 ($\lambda = 1/\text{week}$), and the crawler accesses the page once every two weeks ($f = 1/2$ weeks). Then the bias of the estimated change frequency is $57\%$ ($E[\hat{r}]/r \approx 0.43$)! On the other hand, if the crawler revisits the page every day ($f = 1/\text{day}$), then the statistical variation $\sigma/r$ is $0.75$ and the $95\%$ confidence interval is $\sigma/r = 150\%$! 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: $-\log(\bar{X}/n)$

While the estimator $X/T$ is known to be quite effective when we have a *complete* change history of an element [10, 13], our analysis showed that it is less than desirable when we have an *incomplete* change history. The estimator is biased and we cannot reduce the bias by increasing the sample size. In this subsection, we propose another estimator $-\log(\bar{X}/n)$, which has more desirable properties.

Intuitively, we can derive our new estimator from Lemma 1. In the lemma, we computed the probability $q$ that the element does not change at each access: $q = e^{-\lambda/f} = e^{-r}$. By rearranging this equation, we get

$$r = -\log q$$

Note that we can measure the probability $q$ in the above equation by an experiment; For example, if we accessed an element 100 times, and if the element did not change in 70 accesses, we can reasonably infer that the probability $q$ is 70/100. Then from the equation, we can estimate that the frequency ratio $r = -\log(70/100) = 0.36$. Note that this estimated frequency ratio $0.36$ is slightly larger than $0.30$ ($= 30/100$) that the previous $X/n$ estimates. This is because our new estimator accounts for the changes that we may have missed between some accesses. Also note that we can estimate the value of $q$ more accurately by increasing the sample size. Therefore, we can estimate $r$ more precisely and *decrease the bias* by increasing the sample size! (We verify this claim later.) This property has a significant implication in 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 the estimator $-\log q$, 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 new estimator can be useful for a wider range of applications than $X/n$ is.

To define the new estimator more formally, let $\bar{X}$ be the number of accesses where the element did not change ($\bar{X} = n - X$). Then, our new estimator is

$$\hat{\lambda}/f = -\log(\bar{X}/n) \quad \text{or} \quad \hat{r} = -\log(\bar{X}/n)$$

We can derive our new estimator in a slightly different way. In Section 4.1, we showed that the estimator $X/n$ is biased, such that $E[X/n] = 1 - e^{-r}$. By rearranging the equation, we get

$$r = -\log(1 - E[X/n])$$

Intuitively from the equation, we suspect that we may get the correct $r$ value if we use $-\log(1 - X/n)$ as our estimator, instead of $X/n$. Notice that $1 - X/n = (n - X)/n = \bar{X}/n$. That is, we may consider our new estimator $-\log(\bar{X}/n)$ as the bias corrected estimator of $X/n$!

While intuitively attractive, the estimator $-\log(\bar{X}/n)$ has a mathematical singularity. When the element changes whenever we access it (i.e., $\bar{X} = 0$), the estimator produces infinity, because $-\log(0/n) = \infty$. This singularity makes the estimator technically unappealing, because the expected value of the estimator, $E[\hat{r}]$, is now infinity due to this singularity. (In other words, $\hat{r}$ is biased to infinity!) We can avoid this singularity by adding a small constant, 0.5, to $\bar{X}$ and $n$ as follows[2]:

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

This modified estimator does not have a singularity when $\bar{X} = 0$, because $\log(\frac{0+0.5}{n+0.5}) = \log(\frac{0.5}{n+0.5}) \neq \infty$. In the rest of this subsection, we will study the properties of this modified estimator $\hat{r} = -\log(\frac{\bar{X}+0.5}{n+0.5})$

1. **Is the estimator unbiased?** To see whether the estimator is biased, let us compute the expected value of $\hat{r}$. From the definition of $\bar{X}$,

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

Then,

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

We cannot obtain a closed-form expression in this case. Thus we study its property by numerically evaluating the expression and plotting the results. In Figure 5 we show the graph of $E[\hat{r}]/r$ over $r$ for several $n$ values. 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 when $r > 0.5$, $-\log(\frac{\bar{X}+0.5}{n+0.5})$ is not heavily biased until $r > 1$ for $n \geq 3$.

[2] In Appendix A, we show why we add 0.5, not some other number, to avoid the singularity.

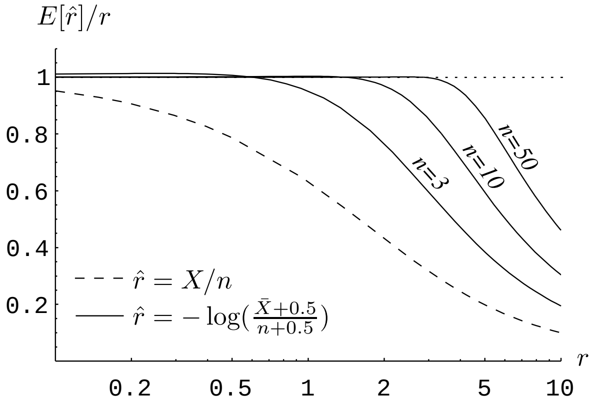


Figure 5: Bias of the estimator $-\log(\frac{\bar{X}+0.5}{n+0.5})$

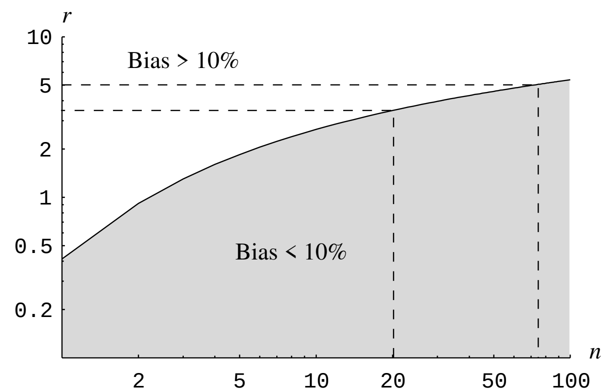


Figure 6: The region where the estimator $-\log(\frac{\bar{X}+0.5}{n+0.5})$ is less than 10% biased

Clearly, the new estimator becomes less biased 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 $r > 3$.

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 Eq. 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 grey area is where the bias is less than 10%. For instance, when $n = 20$, the estimator is less than 10% biased when $r < 3.48$. 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 to select the revisit frequency when we discuss the efficiency of the new estimator.

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 $\sigma/r$ tells us how large this statistical variation can be.

   To plot the $\sigma/r$ graph of $-\log(\frac{\bar{X}+0.5}{n+0.5})$, we first compute the variance of $\hat{r}$

$$\begin{aligned} V[\hat{r}] &= E[\hat{r}^2] - E[\hat{r}]^2 \\ &= \sum_{i=0}^{n} \left( \log \left( \frac{i+0.5}{n+0.5} \right) \right)^2 \binom{n}{i} (e^{-r})^i (1-e^{-r})^{n-i} \\ &\quad - \left( \sum_{i=0}^{n} \log \left( \frac{i+0.5}{n+0.5} \right) \binom{n}{i} (e^{-r})^i (1-e^{-r})^{n-i} \right)^2 \end{aligned}$$

   From this formula, we can numerically compute $\sigma/r = \sqrt{V[\hat{r}]}/r$ for various $r$ and $n$ values and we show the result in Figure 7. As expected, the statistical variation $\sigma/r$ gets smaller as the sample size $n$ increases. For instance, $\sigma/r$ is 0.4 for $r = 1.5$ when $n = 10$, but $\sigma/r$ is 0.2 for the same $r$ value

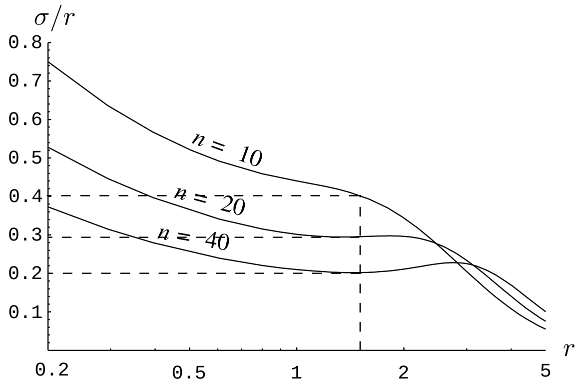


Figure 7: The graph of $\sigma/r$ for the estimator $-\log(\frac{\bar{X}+0.5}{n+0.5})$

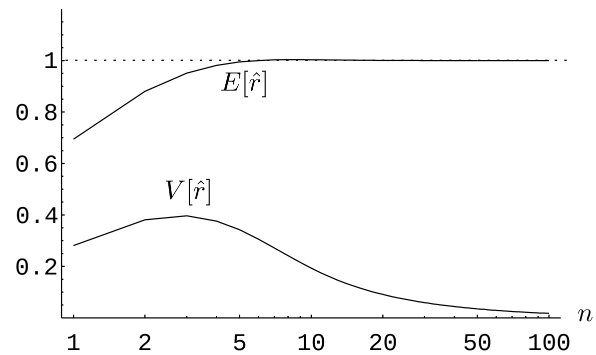


Figure 8: The graphs of $E[\hat{r}]$ and $V[\hat{r}]$ over $n$, when $r = 1$

when $n = 40$.

Also note that the statistical variation $\sigma/r$ takes its minimum at $r \approx 1.5$ within the unbiased region of $r$. 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, $\sigma/r$ is minimum when $r \approx 1.35$. For other values of $n$, we can similarly see that $\sigma/r$ 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 off 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 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?** We can prove that the estimator $\hat{r}$ is consistent, by showing that $\lim_{n\to\infty} E[\hat{r}] = r$ and $\lim_{n\to\infty} V[\hat{r}] = 0$ for any $r$ [12]. Although it is not easy to formally prove, we believe our estimator $\hat{r}$ is indeed consistent. In Figure 5, $E[\hat{r}]/r$ gets close to 1 as $n \to \infty$ for any $r$, and in Figure 7, $\sigma/r$ (thus $V[\hat{r}]$) approaches zero as $n \to \infty$ for any $r$. As an empirical evidence, we show the graphs of $E[\hat{r}]$ and $V[\hat{r}]$ over $n$ when $r = 1$ in Figure 8. $E[\hat{r}]$ clearly approaches 1 and $V[\hat{r}]$ approaches zero.

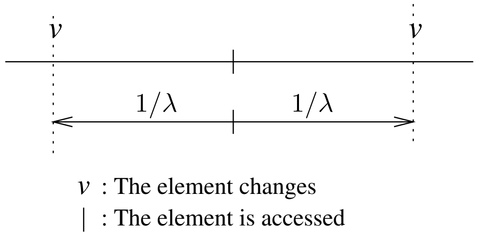


Figure 9: Expected time to the previous change

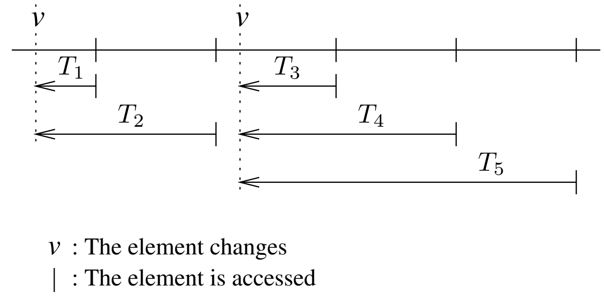


Figure 10: Problems with the estimator based on last modified date

# 5 Estimation of frequency: last date of change

In this section, we propose a new estimator which can estimate the change frequency more precisely when the last-modified date is available. To help readers, we show how we obtained the final estimator step by step, instead of presenting the estimator in its final form and studying its quality. We first start from a simple idea, which is the basis for our new estimator. Then we gradually modify the estimator to fix the "bugs" in the estimator.

## 5.1 Construction of the estimator

How can we use the last-modified date to estimate the change frequency? In a Poisson process, the expected time to the previous event is the same as the expected time to the next event, which is equal to $1/\lambda$ (Figure 9) [8]. Therefore, if we define $T_i$ as the time to the previous change at the $i$th access,

$$T_i = \text{(last-modified time} - \text{access time) at } i\text{th access}$$

$E[T_i]$ is equal to $1/\lambda$. When we accessed the element $n$ times, the sum of all $T_i$'s, $\tau = \sum_{i=1}^{n} T_i$, is then $E[\tau] = \sum_{i=1}^{n} E[T_i] = n/\lambda$. From this equation, we suspect that if we use $n/\tau$ as our estimator, we may get an unbiased estimator $E[n/\tau] = \lambda$. Note that $\tau$ in this equation is a number that needs to be measured by repeated accesses.

While intuitively appealing, this estimator has some problems. The most serious problem is when the element does not change between some accesses. In Figure 10, 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 $\tau$ 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.[3] 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 11.

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 time to the previous change at

[3] We can verify the bias by computing $E[n/\tau]$ when $\lambda \ll f$. We do not show the derivation/graph in this paper.

```
Init()
   N = 0;  /* number of accessed */
   X = 0;  /* number of changes detected */
   T = 0;  /* sum of last modified dates */

Update(Ti, Ii)
   N = N + 1;
   /* Has the element changed? */
   If (Ti < Ii) then
      X = X + 1;
      T = T + Ti;
   else
      T = T + Ii;

Estimate()
   return X/T;
```

Figure 11: Improved estimator based on the last modified date

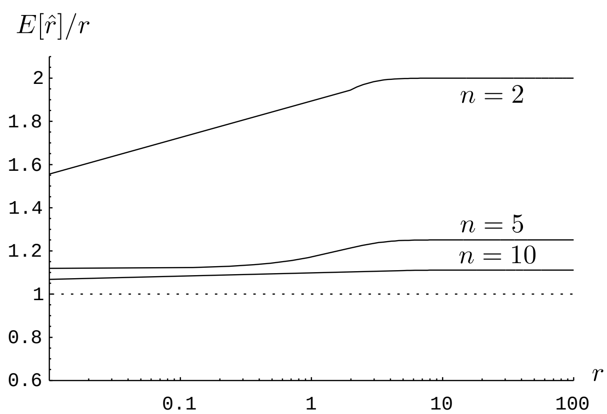


Figure 12: Bias of the estimator in Figure 11

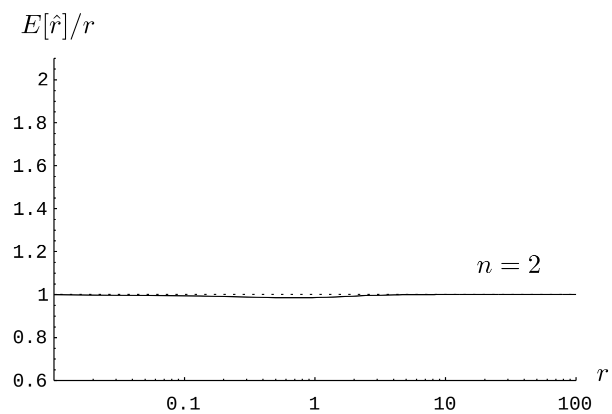


Figure 13: Bias of the estimator with the new `Estimate()` function

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 detected changes. The argument `Ti` to `Update()` is the time to the previous change in the $i$th access and the argument `Ii` is the interval between the accesses. If the element has changed between the $(i-1)$th access and the $i$th 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.

To study the bias of this estimator, we show the the graph of $E[\hat{r}]/r$ over $r$ in Figure 12. We computed this graph analytically (Appendix B) and verified the result by simulations. 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 $n$.

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

Figure 14: New `Estimate()` function that reduces the bias

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, 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, we can prove that the $E[\hat{r}]/r \approx \frac{n}{n-1}$ when $r$ is large, and $E[\hat{r}]/r \approx n\log(\frac{n}{n-1})$ 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.

To correct this bias, we finally modify the `Estimate()` function to the one shown in Figure 14.[4] In the new `Estimate()` function, we use `X'` in place of `X`. The variable `X'` is computed from `X` and `N`, where `X'` is a monotonically increasing function of `X`.[5] Also, `X'` $\approx$ `X` when `N` is large, so the new `Estimate()` function returns similar result to the previous function, when `N` is large.

We show the bias of the estimator with the new `Estimate()` function in Figure 13. The axes in the graph are the same as in Figure 12. Note that the estimator is now 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 only the graph for $n = 2$, because the graphs of other $n$ values essentially overlap with that of $n = 2$. Clearly, this result is a significant win over the estimator of Section 4. We illustrate the implication of this result through an example.

**Example 5** 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 $0.1\%$.) □

While we assumed that we access the element at a regular interval to obtain the graph of Figure 13, note that the estimator of Figures 11 and 14 does not assume that we access the element at a regular interval. In fact, we have experimentally verified through simulations that the bias is negligible even for the irregular access case.

Finally in Figure 15, we show the statistical variation $\sigma/r$ 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 $\sigma/r$. We can see that as $n$ increases, the variation (or the standard deviation) gets smaller. Since $E[\hat{r}] \approx r$ and $\sigma \to 0$ as $n \to \infty$, we believe our new estimator is consistent, although we cannot formally prove it.

[4] 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.

[5] `X'` does not increase monotonically for *real* `X` values, but `X'` *does* increase monotonically for *integer* `X` values. Remember that `X` takes only integer values.

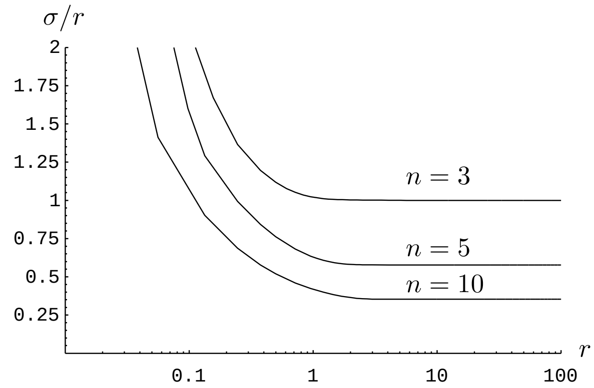


Figure 15: Statistical variation of the new estimator over $r$

# 6 Categorization of frequency: Bayesian inference

So far, we have studied how to *estimate* the change frequency given its change history. But for certain applications, we may only want to categorize elements into several classes based on their change frequencies.

**Example 6** A crawler completely recrawls the web once every month and partially updates a small subset of the pages once every week. Therefore, the crawler does not particularly care whether an element changes every week or every 10 days, but it is mainly interested in whether it needs to crawl a page either every week or every month. That is, it only wants to classify pages into two categories based on their change history. $\square$

For this example, we may still use the estimators of previous sections and classify pages by some threshold frequency. For example, we may classify a page into the every month category if its *estimated* frequency is lower than once every 15 days, and otherwise categorize the page into the every week category. In this section, however, we will study an alternative approach, which is based on the Bayesian decision theory. While the machinery that we use in this section has been long used in statistics community, it has not been applied to the *incomplete* change history case. After a brief description of the estimator, we will study the effectiveness of this method and the implications when the change histories are incomplete.

To help our discussion, let us assume that we want to categorize a web page ($p_1$) into two classes, the pages that change every week ($C_W$) and the pages that change every month ($C_M$). To trace which category $p_1$ belongs to, we maintain two probabilities $P\{p_1 \in C_W\}$ (the probability that $p_1$ belongs to $C_W$) and $P\{p_1 \in C_M\}$ (the probability that $p_1$ belongs to $C_M$). As we access $p_1$ and detect changes, we update these two probabilities based on the detected changes. Then at each point of time, if $P\{p_1 \in C_W\} > P\{p_1 \in C_M\}$, we consider $p_1$ belongs to $C_W$, and otherwise we consider $p_1$ belongs to $C_M$. (While we use a two category example to simplify our discussion, the technique can be generalized to more than two categories.)

Initially we do not have any information on how often $p_1$ changes, so we start with fair values $P\{p_1 \in C_W\} = 0.5$ and $P\{p_1 \in C_M\} = 0.5$. Now let us assume we first accessed $p_1$ after 5 days and we learned that $p_1$ had changed. Then how should we update $P\{p_1 \in C_W\}$ and $P\{p_1 \in C_M\}$? Intuitively, we need to increase $P\{p_1 \in C_W\}$ and decrease $P\{p_1 \in C_M\}$, because $p_1$ had changed in less than a week. But how much should we increase $P\{p_1 \in C_W\}$? We can use Bayesian theorem to answer this question. Mathematically, we want to reevaluate $P\{p_1 \in C_W\}$ and $P\{p_1 \in C_M\}$ given the event $E$, where $E$ represents the change of $p_1$. That

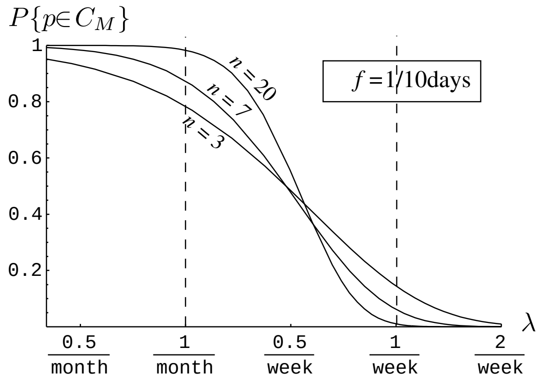


Figure 16: The accuracy of the Bayesian estimator

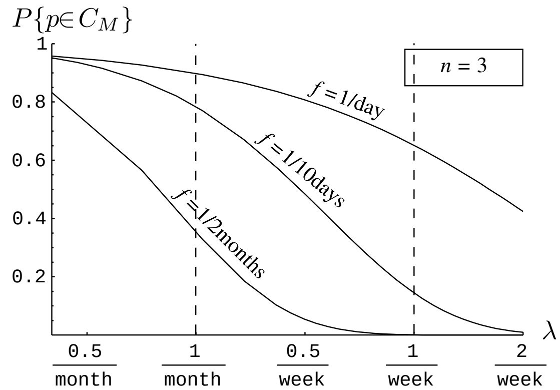


Figure 17: The accuracy of the Bayesian estimator for various access frequencies

is, we want to compute $P\{p_1 \in C_W \,|\, E\}$ and $P\{p_1 \in C_M \,|\, E\}$. According to Bayesian theorem,

$$\begin{aligned} P\{p_1 \in C_W \,|\, E\} &= \frac{P\{(p_1 \in C_W) \wedge E\}}{P\{E\}} = \frac{P\{(p_1 \in C_W) \wedge E\}}{P\{E \wedge (p_1 \in C_W)\} + P\{E \wedge (p_1 \in C_M)\}} \\ &= \frac{P\{E \,|\, p_1 \in C_W\} P\{p_1 \in C_W\}}{P\{E \,|\, p_1 \in C_W\} P\{p_1 \in C_W\} + P\{E \,|\, p_1 \in C_M\} P\{p_1 \in C_M\}} \end{aligned} \tag{2}$$

In the equation, we can compute $P\{E \,|\, p_1 \in C_W\}$ (the probability that $p_1$ changes in 5 days, when its change frequency is a week) and $P\{E \,|\, p_1 \in C_M\}$ (the probability that $p_1$ changes in 5 days, when its change frequency is a month) based on the Poisson process assumption. Also we previously assumed that $P\{p_1 \in C_W\} = P\{p_1 \in C_M\} = 0.5$. Then,

$$P\{p_1 \in C_W \,|\, E\} = \frac{(1 - e^{-5/7})0.5}{(1 - e^{-5/7})0.5 + (1 - e^{-5/30})0.5} \approx 0.77$$

$$P\{p_1 \in C_M \,|\, E\} = \frac{(1 - e^{-5/30})0.5}{(1 - e^{-5/7})0.5 + (1 - e^{-5/30})0.5} \approx 0.23$$

That is, $p_1$ now belongs to $C_W$ with probability $0.77$ and $p_1$ belongs to $C_M$ with probability $0.23$. Note that these new probabilities, $0.77$ and $0.23$, coincides with our intuition. $P\{p_1 \in C_W\}$ has indeed increased to $0.77$ from $0.5$ and $P\{p_1 \in C_M\}$ has decreased.

For the next access, we can repeat the above process. If we detect another change after 5 days, we can update $P\{p_1 \in C_W \,|\, E\}$ and $P\{p_1 \in C_M \,|\, E\}$ by using Eq 2, but now with $P\{p_1 \in C_W\} = 0.77$ and $P\{p_1 \in C_M\} = 0.23$. After this step, $P\{p_1 \in C_W\}$ increases to $0.92$ and $P\{p_1 \in C_M\}$ becomes $0.08$.

Contrary to the estimator of Section 4, note that this new estimator does not require that we access the page at a regular interval. Therefore, even if we access an element at an irregular interval and if we only know whether the element changed or not, we can still categorize pages. Also note that we do not set an arbitrary threshold to categorize elements. Even if we can apply the previous estimators, we still need to set a threshold to classify pages, which can be quite arbitrary. By using the Bayesian estimator, we can avoid setting this arbitrary threshold, because the estimator itself naturally classifies pages.

In Figure 16 we show how accurate the Bayesian estimator is. In the graph, we show the probability that a page is classified into $C_M$ when its change frequency is $\lambda$ (the horizontal axis) for various $n$ values.

We obtained the graph analytically assuming that we access the page every 10 days. From the graph we can see that the estimator classifies the page quite accurately. For instance, when $\lambda \leq \frac{1}{\text{month}}$ the estimator places the page in $C_M$ with more than 80% probability for $n = 3$. Also, when $\lambda \geq \frac{1}{\text{week}}$ it places the page in $C_W$ with more than 80% probability for $n = 3$. Clearly the estimator categorizes the page more accurately as the sample size increases. When $n = 10$, the estimator categorizes the page correctly with more than 95% probability.

While the Bayesian estimator is quite accurate and can handle the irregular access case, we can get more accurate result by carefully deciding on how often we access the page. To illustrate this issue, we show in Figure 17 the accuracy graph for various access frequencies. From the graph, we can see that the estimator is much more accurate when the access frequency lies between $\frac{1}{\text{month}}$ and $\frac{1}{\text{week}}$ ($f = 1/10$ days) than when it lies outside of this range ($f = 1/$day or $f = 1/2$ months). For example, when $f = 1/$day, the estimator places the page in $C_M$ with high probability even when $\lambda > \frac{1}{\text{week}}$. Intuitively, we can explain this phenomenon as follows: When we access the page much more often than it changes, we do not detect any changes to the page at all in the first several accesses. Until we detect the first change, we keep increasing $P\{p_1 \in C_M\}$ and keep decreasing $P\{p_1 \in C_W\}$. Therefore we tend to place the page in $C_M$ when $n$ is small. Although this bias disappears after we access the page many times (more specifically, when $n \approx f/\lambda$ or larger), we can avoid this bias in the first place by selecting an appropriate revisit frequency if we can.

# 7 Conclusion and Future work

In this paper, we studied the problem of estimating the change frequency of an element, when we do *not* have the complete change history. We proposed new estimators that compute the change frequency reasonably well even with the incomplete change history. Also, we analytically showed how effective the proposed estimators are, discussing the practical implications of the various choices.

While our estimators are much more effective than existing ones, we believe we can extend our current work in the following ways:

1. **Irregular access:** The estimators in Sections 5 and 6 can handle the irregular access case, but the estimator in Section 4 requires regular access to the element. We still need to develop an estimator, which can *estimate* the change frequency *without* the last-modified date when the access is *irregular*.

2. **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 pure luck. Then when should we adjust the change frequency to get the optimal result?

Although this problem is a very important question when we only know whether a page has changed or not, it is not a big issue when the last modified date is available. In Section 5, we showed that the bias is practically negligible independent of the access frequency (Figure 13) and that the statistical variation gets smaller as we access the page less frequently (Figure 15). Therefore, it is always good to access the page as slowly as we can. In this case, the only constraint will be how early we need to estimate the change frequency. □

3. **Changing $\lambda$:** Throughout this paper, we assumed that the change frequency $\lambda$ of an element is static (i.e., does not change). This assumption may not be valid in certain cases, and we may need to test whether $\lambda$ changes or not.

   If $\lambda$ changes very slowly, we may be able to detect the change of $\lambda$ and use the estimated $\lambda$ to improve, say, web crawling. For example, if $\lambda$ changes once every month, we may estimate $\lambda$ in the first few days of every month and use the estimated $\lambda$ for the rest of the month. Then by comparing the $\lambda$'s for each month, we may also compute how much $\lambda$ increases/decreases every month. However, when $\lambda$ changes very rapidly, it will be difficult and impractical to estimate $\lambda$ and use the estimated $\lambda$ to improve say crawling or caching.

# References


[1] Michael Baentsch, L. Baum, Georg Molter, S. Rothkugel, and P. Sturm. World Wide Web caching: The application-level view of the internet. *IEEE Communications Magazine*, pages 170–178, June 1997.

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

[3] Junghoo Cho and Hector Garcia-Molina. The evolution of the web and implications for an incremental crawler. *Submitted to VLDB 2000, Experience/Application track*, 2000.

[4] Junghoo Cho and Hector Garcia-Molina. Synchronizing a database to improve freshness. In *Proceedings of the 2000 ACM SIGMOD*, 2000.

[5] James Gwertzman and Margo Seltzer. World-Wide Web cache consistency. In *Proceedings of USENIX 1996 Annual Technical Conference*, 1996.

[6] Joachim Hammer, Hector Garcia-Molina, Jennifer Widom, Wilburt J. Labio, and Yue Zhuge. The Stanford data warehousing project. *IEEE Data Engineering Bulletin*, June 1995.

[7] Venky Harinarayan, Anand Rajaraman, and Jeffery D. Ullman. Implementing data cubes efficiently. In *Proceedings of the 1996 ACM SIGMOD*, 1996.

[8] Leonard Kleinrock. *Queueing Systems: Theory, Vol. 1.* Wiley, John & Sons, Incorporated, 1975.

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

[10] Howard M. Taylor and Samuel Karlin. *An Introduction To Stochastic Modeling*. Academic Press, 3rd edition, 1998.

[11] George B. Thomas, Jr. *Calculus and analytic geometry*. Addison-Wesley, 4th edition, 1969.

[12] Dennis D. Wackerly, William Mendenhall, and Richard L. Scheaffer. *Mathematical Statistics With Applications*. PWS Publishing, 5th edition, 1997.

[13] Robert L. Winkler. *An Introduction to Bayesian Inference and Decision*. Holt, Rinehart and Winston, Inc, 1972.

[14] Haobo Yu, Lee Breslau, and Scott Shenker. A scalable Web cache consistency architecture. In *Proceedings of the 1999 ACM SIGCOMM*, 1999.

[15] Yue Zhuge, Hector Garcia-Molina, Joachim Hammer, and Jennifer Widom. View maintenance in a warehousing environment. In *Proceedings of the 1995 ACM SIGMOD*, 1995.

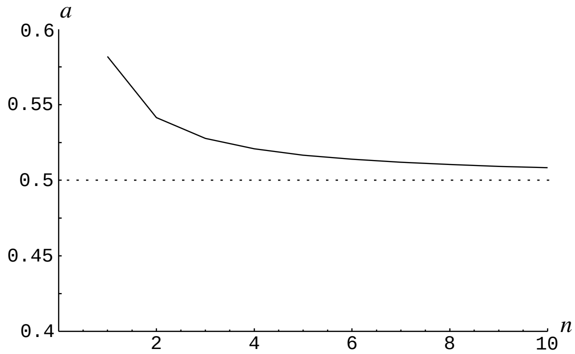


Figure 18: The $a$ values which satisfy the equation $n\log(\frac{n+a}{n-1+a}) = 1$

# A Avoiding the singularity of the estimator $-\log(\bar{X}/n)$

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 $a > 0$.

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 [11]. 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 18. 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 $a = b = 0.5$.

# B Computing the bias of the estimator in Figure 11

We can compute the bias of the estimator by deriving the p.d.f. (probability density function) for $X/T$ and by computing the expected value of $X/T$ based on the density function. Deriving the p.d.f. is quite long and complex, but we briefly sketch how we can obtain it. For the derivation, we assume that we access the element $n$ times at the regular interval $I$ ($= 1/f$ $f$: the access frequency).

It is relatively straightforward to compute the probability that $X = k$ ($k = 1, \ldots, n$). 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),

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

Now we compute the probability that $T = t$ ($0 \leq t \leq nI$) when $X = k$. If we use $T_i$ to represent the *time added to $T$ at the $i$th access*, $T$ can be expressed as

$$T = \sum_{i=1}^{n} T_i$$

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

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

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

$$T = (n - k)I + \sum_{i=1}^{k} T_i \tag{3}$$

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

$$P\{T_i = t\} = \begin{cases} \dfrac{\lambda e^{-\lambda t}}{1 - e^{-r}} & \text{if } 0 \leq t \leq I \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

because the element follows a Poisson process. From Eq. 3 and 4, we can compute the p.d.f. of $T$ when $X = k$ by standard statistical techniques [12]. We use $P_k\{T = t\}$ to denote the p.d.f. of $T$ when $X = k$. Then the expected value of $X/T$ can be computed by the following formula.

$$\begin{aligned} E\left[\frac{X}{T}\right] &= \sum_{k=0}^{n} \left( P\{X = k\} \int_0^{nI} \left(\frac{k}{t}\right) P_k\{T = t\}\, dt \right) \\ &= \sum_{k=1}^{n} \left( P\{X = k\} \int_0^{nI} \left(\frac{k}{t}\right) P_k\{T = t\}\, dt \right) \qquad (\text{when } k = 0,\ \ \frac{k}{t} = 0) \end{aligned}$$

Computing $E[\hat{r}]/r$ from $E[X/T]$ is straightforward. From the equation $\hat{r} = \hat{\lambda}/f = \frac{X}{T}/f$, we can derive

$$\frac{E[\hat{r}]}{r} = \frac{E\left[\frac{X}{fT}\right]}{r} = \sum_{k=1}^{n} \left( \frac{P\{X = k\}}{r} \int_0^{nI} \left(\frac{k}{ft}\right) P_k\{T = t\}\, dt \right) \tag{5}$$

# C Correcting the bias of Figure 11

In this section, we study the mathematical property of Eq. 5 and try to remove its bias. The complete analytical form of Eq. 5 is very complex and hard to interpret. So we first study the limit values of Eq. 5 when $r \to 0$ and $r \to \infty$ and try to extend this result to the general case.

In essence, Eq. 5 is the sum of $n$ terms, where each term corresponds the case when we detect $k$ changes in $n$ accesses. Intuitively, when $\lambda \gg f$ (or $r = \lambda/f \to \infty$) the element will change in all of the accesses, so the dominant term in the equation will be the one with the largest $k$ ($k = n$). Similarly, when $\lambda \ll f$ (or $r = \lambda/f \to 0$) the element will not change in any of the accesses, so the dominant term will be the one with smallest $k$ ($k = 1$). We can confirm this intuition by computing the limit of the factor $P\{X = k\}/r$ in Eq. 5. When $r \to \infty$,

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

Also when $r \to 0$,

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

From these limits, we can compute the limits of Eq. 5.

$$\begin{aligned}
\lim_{r\to\infty} \frac{E[\hat{r}]}{r} &= \lim_{r\to\infty} \sum_{k=1}^{n} \left( \frac{P\{X = k\}}{r} \int_0^{nI} \left(\frac{k}{ft}\right) P_k\{T = t\}\, dt \right) \\
&= \lim_{r\to\infty} \int_0^{nI} \left(\frac{n}{ft}\right) P_n\{T = t\}\, dt \\
&= \frac{n}{n-1} \\
\lim_{r\to 0} \frac{E[\hat{r}]}{r} &= \lim_{r\to 0} \sum_{k=1}^{n} \left( \frac{P\{X = k\}}{r} \int_0^{nI} \left(\frac{k}{ft}\right) P_k\{T = t\}\, dt \right) \\
&= n \lim_{r\to 0} \int_0^{nI} \left(\frac{1}{ft}\right) P_1\{T = t\}\, dt \\
&= n \log\left(\frac{n}{n-1}\right)
\end{aligned}$$

Intuitively, we can interpret this result as follows: When $r$ is very large (i.e. we access the element much less frequent than it changes), we almost always detect $n$ changes in $n$ accesses. So the estimator practically becomes $n/T$ in this case, and according to the limit value the bias is $n/(n-1)$. Note that the estimated frequency ratio is slightly larger than the actual frequency ratio ($n$ versus $n-1$). Then how can we remove this bias when $r$ is large? One obvious solution is to use $(n-1)/T$ as our estimator instead of $n/T$. Then the bias of the estimator becomes

$$\lim_{r\to\infty} \frac{E[\hat{r}]}{r} = \lim_{r\to\infty} \int_0^{nI} \left(\frac{n-1}{ft}\right) P_n\{T = t\}\, dt = \frac{n-1}{n-1} = 1$$

That is, the estimated frequency ratio becomes the same as the actual frequency ratio!

At the other extreme, when $r$ is very small, we almost always detect no change or at best only one change. Since $X/T = 0$ when $X = 0$, the only important term now is when $X = 1$, and the estimator practically becomes $1/T$. As we derived, the bias is $n \log(\frac{n}{n-1})$ when $r \approx 0$. To eliminate this bias, we may use $\frac{1}{n\log(\frac{n}{n-1})}$ as the numerator, instead of 1 when $X = 1$. Then

$$\lim_{r\to 0} \frac{E[\hat{r}]}{r} = n \lim_{r\to 0} \int_0^{nI} \left(\frac{1/(n\log(\frac{n}{n-1}))}{ft}\right) P_1\{T = t\}\, dt = \frac{n\log(\frac{n}{n-1})}{n\log(\frac{n}{n-1})} = 1$$

In summary, we can eliminate the bias when $r \to \infty$ by using $(n-1)$ as the $X$ value when $X = n$, At the other extreme, when $r \to 0$ we can use $\frac{1}{n\log(\frac{n}{n-1})}$ as the $X$ value if $X = 1$.