Efficient Search Engine Measurements

Ziv Bar-Yossef*
Dept. of Electrical Engineering
Technion, Haifa 32000, Israel
Google Haifa Engineering Center, Israel
zivby@ee.technion.ac.il

Maxim Gurevich
Dept. of Electrical Engineering
Technion, Haifa 32000, Israel
gmax@tx.technion.ac.il

ABSTRACT

We address the problem of measuring global quality metrics of search engines, like corpus size, index freshness, and density of duplicates in the corpus. The recently proposed estimators for such metrics [2, 6] suffer from significant bias and/or poor performance, due to inaccurate approximation of the so called "document degrees".

We present two new estimators that are able to overcome the bias introduced by approximate degrees. Our estimators are based on a careful implementation of an approximate importance sampling procedure. Comprehensive theoretical and empirical analysis of the estimators demonstrates that they have essentially no bias even in situations where document degrees are poorly approximated.

Building on an idea from [6], we discuss *Rao Blackwellization* as a generic method for reducing variance in search engine estimators. We show that Rao-Blackwellizing our estimators results in significant performance improvements, while not compromising accuracy.

Categories and Subject Descriptors: H.3.3: Information Search and Retrieval.

General Terms: Measurement, Algorithms.

Keywords: search engines, evaluation, corpus size estimation.

1. INTRODUCTION

Background. In this paper we focus on methods for external evaluation of search engines. These methods interact only with the public interfaces of search engines and do not rely on privileged access to internal search engine data or on specific knowledge of how the search engines work. External evaluation provides the means for objective benchmarking of search engines. Such benchmarks can be used by search engine users and clients to gauge the quality of the service they get and by researchers to compare search engines. Even search engines themselves may benefit from external benchmarks, as they can help them reveal their strengths and weaknesses relative to their competitors.

Our study concentrates on measurement of global quality metrics of search engines, like corpus size, index freshness, and density of spam or duplicate pages in the corpus. Such metrics are relevance neutral, and therefore no human judgment is required for computing them. Still, as external access to search engine data is highly restricted, designing automatic methods for measuring these quality metrics is very challenging. Our objective is to design measurement algorithms that are both *accurate* and *efficient*. Efficiency is particularly important for two reasons. First, efficient algorithms can be executed even by parties whose resources are limited, like researchers. Second, as search engines are highly dynamic, efficient algorithms are necessary for capturing instantaneous snapshots of the search engines.

Problem statement. Let $\mathcal{D}$ denote the corpus of documents indexed by the search engine. We focus on measurement of metrics that can be expressed as either *sums* or *averages* over $\mathcal{D}$. Given a *target function* $f : \mathcal{D} \to \mathbb{R}$, the *sum metric* and the *average metric* corresponding to f are:

$$\mathrm{sum}(f) \triangleq \sum_{\mathrm{x}\in\mathcal{D}} f(\mathrm{x}), \quad \mathrm{avg}(f) \triangleq \frac{\mathrm{sum}(f)}{|\mathcal{D}|}.$$

(In fact, we address sums and averages w.r.t. arbitrary measures. See more details in Section 2.) Almost all the global quality metrics we are aware of can be expressed as sum or average metrics. For example, the corpus size, $|\mathcal{D}|$, is the sum of the constant 1 function ($f(\mathrm{x}) = 1$ for all x); the density of spam pages in the corpus is the average of the spam indicator function ($f(\mathrm{x}) = 1$, if x is a spam page, and $f(\mathrm{x}) = 0$, otherwise); the number of unique documents in the corpus is the sum of the inverse duplicate-count function ($f(\mathrm{x}) = 1/d_{\mathrm{x}}$, where d_{x} is the number of duplicates x has, including x itself). Many other metrics, like search engine overlap, sizes of subsets of the corpus, or index freshness can be expressed as sums or averages as well. We allow also sums and averages of vector-valued functions $f : \mathcal{D} \to \mathbb{R}^m$, which capture metrics like the distribution of pages in the corpus by language, country domain, or topic.

A *search engine estimator* for $\mathrm{sum}(f)$ (resp., $\mathrm{avg}(f)$) is a probabilistic procedure, which submits queries to the search engine, fetches pages from the web, computes the target function f on documents of its choice, and eventually outputs an estimate of $\mathrm{sum}(f)$ (resp., $\mathrm{avg}(f)$). The quality of an estimator is measured in terms of its *bias* and its *variance*. The efficiency of an estimator is measured in terms of the number of queries it submits to the search engine, the number of web pages it fetches, and the number of documents on which it computes the target function f.

State of the art. Brute-force computation of search quality metrics is infeasible, due to the huge size of the corpus

*Supported by the European Commission Marie Curie International Re-integration Grant.

WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.
ACM 978-1-59593-654-7/07/0005.

and the highly restricted access to it. Every user is limited to a few thousand queries per day and only the top k matches are returned. (k is the search engine's *result set size limit*; e.g., $k = 1,000$ for Google and Yahoo!) If a query "overflows", i.e., has more than k matches, the user has no way of accessing results beyond the top k. Thus, fetching all the pages in a search engine's corpus is practically impossible.

An alternative to brute-force computation is sampling. One samples random pages from the corpus and uses them to estimate the quality metrics. If the samples are unbiased, then a small number of them is sufficient to obtain accurate estimations. The main challenge is to design algorithms that can efficiently generate unbiased samples from the corpus using queries to the public interface. Bharat and Broder [4] were the first to propose such an algorithm. The samples produced by their algorithm, however, suffered from severe bias towards long, content-rich, documents. In our previous paper [2], we were able to correct this bias by proposing a technique for *simulating* unbiased sampling by biased sampling. To this end, we applied several stochastic simulation methods, like rejection sampling [24] and the Metropolis-Hastings algorithm [22, 13]. Stochastic simulation, however, incurs significant overhead: in order to generate each unbiased sample, numerous biased samples are used, and this translates into elevated query and fetch costs. For instance, our most efficient sampler needed about 2,000 queries to generate each uniform sample.

In an attempt to address this lack of efficiency, we also experimented [2] with *importance sampling* estimation. Importance sampling [21, 16] enables estimation of sums and averages *directly* from the biased samples, without first generating unbiased samples. This technique can significantly reduce the stochastic simulation overhead. Nevertheless, our estimators in [2] used stochastic simulation twice (once to select random queries and once to select random documents), and we were able to use importance sampling to eliminate only the latter of the two. Furthermore, our importance sampling estimator was still wasteful, as it used only a single result of each submitted query and discarded all the rest.

Broder *et al.* [6] have recently made remarkable progress by proposing a new estimator for search engine corpus size. Their estimator (implicitly) employs importance sampling and does not resort to stochastic simulation at all. Moreover, the estimator somehow makes use of *all* query results in the estimation and is thus less wasteful than the estimators in [2]. Broder *et al.* claimed their method can be generalized to estimate other metrics, but have not provided any details.

The degree mismatch problem. A prerequisite for applying importance sampling is the ability to compute for each biased sample an *importance weight*. The importance weights are used to balance the contributions of the different biased samples to the final estimator.

In the estimators of [2, 6], computing the importance weight of a sample document translates into calculation of the document's "degree". Given a large pool of queries (e.g., "phrase queries of length 5", or "8-digit string queries"), the *degree* of a document w.r.t. the pool is the number of queries from the pool to whose results x belongs. As the estimators of [2, 6] choose their sample documents from the results of random queries drawn from a query pool, these samples are biased towards high degree documents. Document degrees, therefore, constitute the primary factor in determining the importance weights of sample documents.

As importance weights (and hence degrees) are computed for every sample document, degree computation should be extremely efficient. Ideally, it should be done based on the content of x alone and without submitting queries to the search engine. The above estimators do this by extracting all the terms/phrases from x and counting how many of them belong to the pool. The resulting number is the document's *predicted degree* and is used as an approximation of the real degree.

In practice, the predicted degree may be quite different from the actual degree, since we do not exactly know how the search engine parses documents or how it selects the terms by which to index the document. Moreover, we do not know a priori which of the queries that the document matches overflow; the document may fail to belong to the result sets of such queries if it is ranked too low. These factors give rise to a *degree mismatch*—a gap between the predicted degree and the actual degree. The degree mismatch implies that the importance weights used by the estimators are not accurate, and this can significantly affect the quality of the produced estimates.

In [2], we proved that if the density of overflowing queries among all the queries that a document matches has low variance, then the bias incurred by degree mismatch is small. Broder *et al.* [6] have not analyzed the effect of degree mismatch on the quality of their estimations.

Several heuristic methods have been used by [2, 6] to overcome the degree mismatch problem. In order to reduce the effect of overflowing queries, a pool of queries that are unlikely to overflow was chosen ([2] used a pool of phrases of length 5, while [6] used a pool of 8-digit strings). However, pools that have low density of overflowing queries are also more likely to have poor *coverage*, creating another bias. [6] remove overflowing queries from the pool by eliminating terms that occur frequently in a training corpus. However, this heuristic can have many false positives or false negatives, depending on the choice of the frequency threshold.

Our contributions. In this paper we show how to overcome the degree mismatch problem. We present two search engine estimators that remain nearly unbiased and very efficient, even in the presence of highly mismatching degrees.

Our first contribution is a rigorous analysis of an "approximate importance sampling" procedure. We prove that using importance sampling with approximate weights rather than the real weights incurs both a multiplicative bias and an additive bias. The analysis immediately implies that the estimator of Broder *et al.* [6] suffers from significant bias in the presence of degree mismatch.

Our second contribution is the design of two new importance sampling estimators. Our estimators use approximate weights, but are able to eliminate the more significant multiplicative bias, leading to nearly unbiased estimates. These estimators can be viewed as generalizations of "ratio importance sampling" (cf. [20]) and importance sampling with approximate trial weights [2]. The first estimator, the *Accurate Estimator (AccEst)*, uses few search engine queries to *probabilistically* calculate exact document degrees, and is thereby able to achieve essentially unbiased estimations. The second estimator, the *Efficient Estimator (EffEst)*, predicts document degrees from the contents of documents alone, without submitting queries to the search engine, similarly to [2, 6]. To eliminate the multiplicative bias factor incurred by degree mismatch, EffEst estimates this factor by invoking

AccEst. Nevertheless, as the bias factor is independent of the target function being measured, this costly computation can be done only once in a pre-processing step, and then be reused in multiple invocations of EffEst. Hence, the amortized cost of EffEst is much lower than that of AccEst. The estimations produced by EffEst may be slightly less accurate than those of AccEst, because the additive bias cannot be always eliminated.

We note that our estimators are applicable to both the sum metric and the average metric w.r.t. any target function. This in contrast to the estimators in [2], which are efficiently applicable only to average metrics, and the estimator in [6], which is applicable only to sum metrics.

Our last contribution builds on the observation that the estimator of Broder *et al.* implicitly applies *Rao Blackwellization* [7], which is a well-known statistical tool for reducing estimation variance. This technique is what makes their estimator so efficient. We show that Rao Blackwellization can be applied to our estimators as well and prove that it is guaranteed to make them more efficient as long as the results of queries are sufficiently variable.

Experimental results. We evaluated the bias and the efficiency of our estimators as well as the estimators from [2, 6] on a local search engine that we built over a corpus of 2.4 million documents. To this end, we used the estimators to estimate two different metrics: corpus size and density of sports pages. The empirical study confirms our analytical findings: in the presence of many overflowing queries, our estimators have essentially no bias, while the estimator of Broder *et al.* suffers from significant bias. For example, the relative bias of AccEst in the corpus size estimation was 0.01%, while the relative bias of the estimator of Broder *et al.* was 75%. The study also showed that our new estimators are up to 375 times more efficient than the rejection sampling estimator from [2]. Finally, the study demonstrated the effectiveness of Rao-Blackwellization by reducing the query cost of estimators by up to 79%.

We used our estimators to measure the absolute sizes of three major search engines. The results of this study gave gross underestimates of the true search engine sizes, largely due to the limited coverage of the pool of queries we used. Even so, we showed that our estimates are up to 74 times higher than the estimates produced by (our implementation of) the *Broder et al.* estimator.

Other related work. Apart from [4, 2, 6], several other studies estimated global quality metrics of search engines, like relative corpus size. These studies are based on analyzing anecdotal queries [5], queries collected from user query logs [17, 10], or queries selected randomly from a pool a la Bharat and Broder [12, 8]. Using capture-recapture techniques (cf. [19]) some of these studies infer measurements of the whole web. Due to the bias in the samples, though, these estimates lack any statistical guarantees.

A different approach for evaluating search quality is by sampling pages from the whole web [18, 14, 15, 1, 23]. Sampling from the whole web, however, is a more difficult problem, and therefore all the known algorithms suffer from severe bias.

2. PRELIMINARIES

In this section we introduce notations and definitions used throughout the paper.

Search engines. Fix a search engine $\mathcal{S}$ whose corpus $\mathcal{D}$ we wish to measure. For each query q, the *index* of the search engine consists of a list of *matching documents* from $\mathcal{D}$. Given q, the search engine returns these matches, ranked by relevance. The search engine has a *result set size limit* k. If the number of matches for a query q exceeds k, only the top k matches are returned. We denote the actual list of results returned on query q by $\mathsf{results}(\mathrm{q})$.

The *cardinality* of a query q, denoted $\mathrm{card}(\mathrm{q})$, is the total number of matches it has. We say that q *overflows*, if $\mathrm{card}(\mathrm{q}) > k$, and that it *underflows*, if $\mathrm{card}(\mathrm{q}) = 0$.

Measures and integrals. We restrict to measurements of metrics that can be written as discrete integrals over $\mathcal{D}$:

$$\mathrm{Int}_\pi(f) \triangleq \sum_{\mathrm{x}\in\mathcal{D}} f(\mathrm{x})\pi(\mathrm{x}).$$

Here, $f : \mathcal{D} \to \mathbb{R}$ is a *target function* and $\pi : \mathcal{D} \to [0, \infty)$ is a *target measure*.

π induces a corresponding probability distribution on $\mathcal{D}$: $\pi'(\mathrm{x}) = \frac{\pi(\mathrm{x})}{Z_\pi}$, where $Z_\pi = \sum_{\mathrm{x}\in\mathcal{D}} \pi(\mathrm{x})$ is the *normalization constant* of π. We say that two different measures are the same *up to normalization*, if they induce the same probability distribution, but have different normalization constants.

When the target measure is a distribution, we call the integral $\mathrm{Int}_\pi(f)$ an *average metric*. Otherwise, it is a *sum metric*. For example, corpus size is a sum metric, as the corresponding target measure is the uniform 1-measure ($\pi(\mathrm{x}) = 1$ for all $\mathrm{x} \in \mathcal{D}$), while the density of spam pages in the corpus is an average metric, because its corresponding measure is the uniform distribution on $\mathcal{D}$ ($\pi(\mathrm{x}) = 1/|\mathcal{D}|$ for all x).

Everything we do in this paper can be generalized to deal with vector-valued functions $f : \mathcal{D} \to \mathbb{R}^m$. Yet, for simplicity of exposition, we focus on scalar functions.

Search engine estimators. A *search engine estimator* for a metric $\mathrm{Int}_\pi(f)$ is a randomized procedure P, which interacts only with the public interface of the search engine and/or with the web, and produces an estimate of $\mathrm{Int}_\pi(f)$. The procedure is assumed to have access to an "f-oracle" and to a "π-oracle". Given a document $\mathrm{x} \in \mathcal{D}$, the f-oracle returns $f(\mathrm{x})$ and the π-oracle returns $\hat{\pi}(\mathrm{x})$. $\hat{\pi}$ is a measure on $\mathcal{D}$, which is identical to π up to normalization. We denote by $Z_{\hat{\pi}}$ the normalization constant of $\hat{\pi}$. If $\mathrm{Int}_\pi(f)$ is a sum metric, we require $\hat{\pi} = \pi$. If $\mathrm{Int}_\pi(f)$ is an average metric, $\hat{\pi}$ can be any measure that is the same as π up to normalization and the normalization constant $Z_{\hat{\pi}}$ need not be known in advance. For example, when estimating the density of spam in the corpus, the π-oracle can return for each document x the value $\hat{\pi}(\mathrm{x}) = 1$, which is identical to the target uniform distribution π up to normalization.

The quality of an estimator is measured in terms of two parameters: *bias* and *variance*. The *bias* of P is the difference $|\,\mathbb{E}(\mathrm{P}) - \mathrm{Int}_\pi(f)|$. P is called *unbiased*, if $\mathbb{E}(\mathrm{P}) = \mathrm{Int}_\pi(f)$. The *variance* of P is $\mathrm{var}(\mathrm{P}) = \mathbb{E}((\mathrm{P} - \mathbb{E}(\mathrm{P}))^2)$. The estimator's bias and variance can be used (via Chebyshev's inequality) to determine estimation confidence intervals, i.e., parameters $\epsilon > 0$ and $0 < \delta < 1$ for which $\Pr((1-\epsilon)\,\mathrm{Int}_\pi(f) \le \mathrm{P} \le (1+\epsilon)\,\mathrm{Int}_\pi(f)) \ge 1 - \delta$.

The three expensive resources used by search engine estimators are: (1) queries submitted to the search engine; (2) web pages fetched; (3) calculations of the function f. The *expected query cost* of an estimator P, denoted $\mathrm{qcost}(\mathrm{P})$, is the expected number of queries P submits to the search en-

gine. Expected query cost cannot be used to compare the efficiency of different estimators, as they may have different variances. The *amortized query cost*, defined as $\text{qcost}(\text{P}) \cdot \text{var}(\text{P})$, is a more robust measure of efficiency. Expected and amortized fetch/function costs are defined similarly.

Query pools and document degrees. Let $\mathcal{P}$ be a pool of queries. For a document $\text{x} \in \mathcal{D}$, we denote by $\mathsf{queries}_{\mathcal{P}}(\text{x})$ the set of queries to whose result sets x belongs:

$$\mathsf{queries}_{\mathcal{P}}(\text{x}) \triangleq \{\text{q} \in \mathcal{P} \mid \text{x} \in \mathsf{results}(\text{q})\}.$$

The *degree* of x w.r.t. $\mathcal{P}$, denoted $\deg_{\mathcal{P}}(\text{x})$, is $|\mathsf{queries}_{\mathcal{P}}(\text{x})|$. A basic task carried out again and again by our algorithms is the following: given a query pool $\mathcal{P}$ and a document x, compute $\deg_{\mathcal{P}}(\text{x})$. Since we need to perform this task ultra-efficiently, we would like to do it based on the content of x alone and without submitting any queries to the search engine. This may seem initially impossible to do. However, if $\mathcal{P}$ consists of term/phrase queries, we can predict the queries in $\mathcal{P}$ that x matches, simply by extracting all the terms/phrases that occur in the text of x and that also belong to $\mathcal{P}$. Let $\mathsf{pqueries}_{\mathcal{P}}(\text{x})$ denote this set of *predicted queries* and let $\text{pdeg}_{\mathcal{P}}(\text{x})$ denote the *predicted degree* of x, i.e., $|\mathsf{pqueries}_{\mathcal{P}}(\text{x})|$. $\text{pdeg}_{\mathcal{P}}(\text{x})$ will be our approximation for $\deg_{\mathcal{P}}(\text{x})$.

In practice, however, $\mathsf{pqueries}_{\mathcal{P}}(\text{x})$ may contain queries that do not belong to $\mathsf{queries}_{\mathcal{P}}(\text{x})$, and, conversely, $\mathsf{queries}_{\mathcal{P}}(\text{x})$ may contain queries that do not belong to $\mathsf{pqueries}_{\mathcal{P}}(\text{x})$. We would like to somehow bridge the gap between the two. Dealing with queries in $\mathsf{queries}_{\mathcal{P}}(\text{x}) \setminus \mathsf{pqueries}_{\mathcal{P}}(\text{x})$ is relatively easy. We call a query q *valid for x*, if x belongs to the result set of q and we could have anticipated that by inspecting the content of x. That is, $\text{q} \in \mathsf{queries}_{\mathcal{P}}(\text{x}) \cap \mathsf{pqueries}_{\mathcal{P}}(\text{x})$. The set of *valid results* for a query q is defined as follows: $\mathsf{vresults}(\text{q}) \triangleq \{\text{x} \in \mathsf{results}(\text{q}) \mid \text{q is valid for x}\}$. Our algorithms will use only the valid results of queries, rather than all the results. Therefore, for each document x, the set of *valid queries for x* is: $\mathsf{vqueries}_{\mathcal{P}}(\text{x}) \triangleq \{\text{q} \in \mathcal{P} \mid \text{x} \in \mathsf{vresults}(\text{q})\} = \mathsf{queries}_{\mathcal{P}}(\text{x}) \cap \mathsf{pqueries}_{\mathcal{P}}(\text{x})$. The *valid degree* of x, denoted $\text{vdeg}_{\mathcal{P}}(\text{x})$, is $|\mathsf{vqueries}_{\mathcal{P}}(\text{x})|$. By using only valid results, we are guaranteed that $\mathsf{vqueries}_{\mathcal{P}}(\text{x}) \subseteq \mathsf{pqueries}_{\mathcal{P}}(\text{x})$, contributing to shrinking the difference between the two.

Addressing queries in $\mathsf{pqueries}_{\mathcal{P}}(\text{x}) \setminus \mathsf{vqueries}_{\mathcal{P}}(\text{x})$ is a more serious problem, because figuring out which of the queries in $\mathsf{pqueries}_{\mathcal{P}}(\text{x})$ do not belong to $\mathsf{vqueries}_{\mathcal{P}}(\text{x})$ requires submitting all these queries to the search engine—a prohibitively expensive task. We call the ratio $\frac{\text{vdeg}_{\mathcal{P}}(\text{X})}{\text{pdeg}_{\mathcal{P}}(\text{X})}$ the *validity density* of x and denote it by $\text{vdensity}_{\mathcal{P}}(\text{x})$. From the above, $\text{vdensity}_{\mathcal{P}}(\text{x}) \in [0, 1]$ for all x. The closer $\text{vdensity}_{\mathcal{P}}(\text{x})$ is to 1, the better is our approximation of $\text{vdeg}_{\mathcal{P}}(\text{x})$.

Usage of the k available results of overflowing queries in our estimations is a potential source of bias, since such queries may favor documents with high static rank. In order to eliminate any bias towards such documents, our algorithms simply ignore overflowing queries. Technically, we do this by defining all the results of overflowing queries to be "invalid". Therefore, if q overflows, $\mathsf{vresults}(\text{q}) = \emptyset$. This in particular means that $\mathsf{vqueries}_{\mathcal{P}}(\text{x})$ cannot contain any overflowing query.

We say that the pool $\mathcal{P}$ *covers* a document x, if there is at least one query $\text{q} \in \mathcal{P}$ which is valid for x. That is, $\mathsf{vqueries}_{\mathcal{P}}(\text{x}) \neq \emptyset$. Note that documents that do not contain any of the terms/phrases in $\mathcal{P}$ or that match only overflowing queries in $\mathcal{P}$ are not covered by $\mathcal{P}$.

3. IMPORTANCE SAMPLING

In this section we present a basic importance sampling search engine estimator. It will be used as a basis for the more advanced estimators presented in subsequent sections.

Setup. Like the pool-based estimators in [4, 2, 6], our estimator assumes knowledge of an *explicit* query pool $\mathcal{P}$. For example, in our experiments, we used a pool of 1.75 billion English phrases of lengths 3-5. Such a pool can be constructed in a pre-processing step, by crawling a representative corpus of web documents and extracting terms/phrases that occur therein (we used the ODP [9] directory for this purpose). We can run the estimator with any such pool, yet the choice of the pool may affect the bias and the efficiency of the estimator.

No matter how large $\mathcal{P}$ is, in practice, there will always be documents in $\mathcal{D}$ that $\mathcal{P}$ does not cover. Since our estimator uses only queries from $\mathcal{P}$, it can never reach such documents. This means that our estimator can estimate integrals only over the set of documents that are covered by $\mathcal{P}$ and not over the entire corpus $\mathcal{D}$. So regardless of the statistical bias of the estimator, there will be an additional "built-in" bias that depends on the coverage of the chosen query pool. To simplify our discussion, from now on, we suppress this coverage-induced bias and use $\mathcal{D}$ to denote only the documents that are covered by $\mathcal{P}$.

Importance sampling estimation. Recall that we would like to estimate the integral $\text{Int}_{\pi}(f)$ for some given target function f and target measure π on $\mathcal{D}$. The naive Monte Carlo estimator (cf. [20]) for $\text{Int}_{\pi}(f)$ works as follows: (1) sample a random document X from the distribution π' induced by π; (2) compute the normalization constant Z_{π} of π; (3) output $f(\text{X}) \cdot Z_{\pi}$. It is easy to check that this estimator is unbiased. Its variance can be reduced by averaging over multiple independent instances of the estimator.[1]

In our setting, however, this simple estimator is impractical, for the following reasons: (1) sampling from the distribution π' may be hard or costly (e.g., when π is a uniform measure on $\mathcal{D}$); (2) computation of the normalization constant Z_{π} may be costly (e.g., in corpus size estimation, $Z_{\pi} = |\mathcal{D}|$, which is exactly the quantity we need to estimate); (3) the random variable $f(\text{X})$ may have high variance. Importance sampling [21, 16, 20] can be used in these circumstances to obtain a more efficient estimator.

The basic idea of importance sampling is the following. Instead of sampling a document X from π', the estimator samples a document Y from a different *trial distribution* p on $\mathcal{D}$. p can be any distribution, as long as $\text{supp}(p) \supseteq \text{supp}(\pi)$ (here, $\text{supp}(p) = \{\text{x} \in \mathcal{D} \mid p(\text{x}) > 0\}$ is the *support* of p; $\text{supp}(\pi)$ is defined similarly). In particular, we can choose it to be a distribution that is easy to sample from. The importance sampling estimator is then defined as follows:

$$\text{IS}(\text{Y}) \triangleq f(\text{Y}) \cdot \frac{\pi(\text{Y})}{p(\text{Y})} = f(\text{Y}) \cdot w(\text{Y}).$$

The correction term $w(\text{Y}) \triangleq \frac{\pi(\text{Y})}{p(\text{Y})}$ is called the "importance weight" and it guarantees that $\text{IS}(\text{Y})$ is an unbiased estima-

[1] More efficient aggregation techniques, like the *median of averages* (cf. [11, 6]), exist. For simplicity of exposition, we will focus mainly on averaging in this paper.

tor of $\mathrm{Int}_\pi(f)$: $\mathbb{E}_p(\mathrm{IS}(\mathrm{Y})) =$

$$= \sum_{\mathrm{y}\in\mathrm{supp}(\pi)} p(\mathrm{y}) f(\mathrm{y}) \frac{\pi(\mathrm{y})}{p(\mathrm{y})} = \sum_{\mathrm{y}\in\mathrm{supp}(\pi)} f(\mathrm{y})\pi(\mathrm{y}) = \mathrm{Int}_\pi(f).$$

Implementation of an importance sampling estimator requires: (1) ability to sample efficiently from the trial distribution p; and (2) ability to compute the importance weight $w(\mathrm{y})$ and the function value $f(\mathrm{y})$, for any given element $\mathrm{y} \in \mathcal{D}$. There is no need to know the normalization constant Z_π or to be able to sample from π'.

The sample space. The sample space of the importance sampling estimator proposed in our previous paper [2] was the corpus of documents $\mathcal{D}$. The trial distribution p was the "document degree distribution", in which documents are sampled proportionally to their degree. In order to sample documents from this distribution, we had to sample queries from the pool $\mathcal{P}$ proportionally to their cardinality, and then to sample random documents from the result sets of these queries. As cardinalities of queries are not known in advance, sampling queries from $\mathcal{P}$ required application of rejection sampling. This step incurred significant overhead.

In this paper we propose a different sample space. Rather than sampling queries and then documents in two separate steps, we sample them *together*. The sample space is then $\Omega = \mathcal{P} \times \mathcal{D}$. Each sample is a *query-document pair* (q, x). We extend the target measure π on $\mathcal{D}$ into a target measure μ on Ω and the function f on $\mathcal{D}$ into a function F on Ω. The extension is done in such a way that $\mathrm{Int}_\mu(F)$ equals $\mathrm{Int}_\pi(f)$. We thus reduce the problem of estimating $\mathrm{Int}_\pi(f)$ to the problem of estimating $\mathrm{Int}_\mu(F)$. For the latter, we can apply importance sampling directly on the two-dimensional sample space Ω, without having to resort to rejection sampling.

Let $\Omega = \mathcal{P} \times \mathcal{D}$. We extend π into a measure on Ω as follows: $\mu(\mathrm{q}, \mathrm{x}) \triangleq \frac{I(\mathrm{x}\in\mathsf{vresults}(\mathrm{q}))\cdot\pi(\mathrm{x})}{\mathrm{vdeg}(\mathrm{x})}$, where I is an indicator function: $I(\text{condition}) = 1$ if the condition is true, and 0 otherwise. The connection between μ and π is given by the following proposition:

PROPOSITION 3.1. *π is the marginal measure of μ on $\mathcal{D}$. Furthermore, the normalization constants of π and μ are the same.*

It follows from the proposition that μ is a distribution if and only if π is a distribution.

Similarly, we extend the function f on $\mathcal{D}$ into a function F on Ω as follows: $F(\mathrm{q}, \mathrm{x}) \triangleq f(\mathrm{x})$. It is easy to see that $\mathrm{Int}_\mu(F) = \mathrm{Int}_\pi(f)$. Our estimator therefore estimates the integral $\mathrm{Int}_\mu(F)$.

The trial distribution. We next describe the trial distribution for selecting query-document pairs from Ω. Let $\mathcal{P}_+$ denote the collection of queries in $\mathcal{P}$ that have at least one valid result: $\mathcal{P}_+ \triangleq \{\mathrm{q} \in \mathcal{P} \mid \mathsf{vresults}(\mathrm{q}) \neq \emptyset\}$. Our trial distribution selects a pair (q, x) as follows: (1) pick a query $\mathrm{q} \in \mathcal{P}_+$ uniformly at random; (2) pick a document $\mathrm{x} \in \mathsf{vresults}(\mathrm{q})$ uniformly at random:

$$p(\mathrm{q}, \mathrm{x}) \triangleq \frac{1}{|\mathcal{P}_+|} \cdot \frac{I(\mathrm{x} \in \mathsf{vresults}(\mathrm{q}))}{\mathrm{vcard}(\mathrm{q})}.$$

Here, $\mathrm{vcard}(\mathrm{q})$ is the number of valid results q has. Sampling from p can be done easily (see Figure 1): we repeatedly select queries from $\mathcal{P}$ uniformly at random, submit them to the search engine, and extract the valid results from each such query. We stop when reaching a query that has at least one valid result. We then select a document from the set of valid results of this query uniformly at random.

```
1: Function TrialDistributionSampler(S, P)
2:   while (true) do
3:     Q := uniformly chosen query from P
4:     submit Q to S
5:     if Q overflows continue
6:     results(Q) := results returned by S
7:     download all pages in results(Q)
8:     vresults(Q) := valid results extracted from results(Q)
9:     if (vresults(Q) ≠ ∅) then
10:      X := uniformly chosen document from vresults(Q)
11:      return (Q,X)
```

Figure 1: Trial distribution sampler

The importance weights. The importance weights corresponding to the target measure μ and the trial distribution p are the following:

$$w(\mathrm{q}, \mathrm{x}) = \frac{\mu(\mathrm{q}, \mathrm{x})}{p(\mathrm{q}, \mathrm{x})} = \frac{\pi(\mathrm{x}) \cdot |\mathcal{P}_+| \cdot \mathrm{vcard}(\mathrm{q})}{\mathrm{vdeg}(\mathrm{x})}.$$

Thus, the importance sampling estimator for $\mathrm{Int}_\pi(f)$ is:

$$\mathrm{IS}(\mathrm{Q}, \mathrm{X}) \triangleq \frac{f(\mathrm{X}) \cdot \pi(\mathrm{X}) \cdot |\mathcal{P}_+| \cdot \mathrm{vcard}(\mathrm{Q})}{\mathrm{vdeg}(\mathrm{X})},$$

where (Q, X) is a sample from the trial distribution p. The caveat with this estimator is that computing the importance weights may be hard or costly to do for three reasons: (1) we cannot compute $\mathrm{vdeg}(\mathrm{x})$ without submitting queries to the search engine; (2) we do not know a priori which of the queries in $\mathcal{P}$ have at least one valid result and therefore cannot compute $|\mathcal{P}_+|$; and (3) if $\mathrm{Int}_\pi(f)$ is an average metric, we may know $\pi(\mathrm{x})$ only up to normalization.

The estimator of Broder *et al.* [6] resembles the above importance sampling estimator for the special case of corpus size estimation. As they could not compute the exact importance weights, they used approximate importance weights, by substituting $\mathrm{pdeg}(\mathrm{x})$ for $\mathrm{vdeg}(\mathrm{x})$. It is not clear, however, what is the effect of the approximate importance weights on the bias of the estimator. Also, it is unknown how to extend the estimator to work for average metrics. We address these issues in the next section.

4. APPROXIMATE IMPORTANCE SAMPLING

Suppose we come up with an *approximate weight function* $u(\mathrm{q}, \mathrm{x})$, which is "similar", but not identical, to $w(\mathrm{q}, \mathrm{x})$ (we will discuss possible alternatives for $u(\mathrm{q}, \mathrm{x})$ in the next section). What is the effect of using $u(\mathrm{q}, \mathrm{x})$ rather than $w(\mathrm{q}, \mathrm{x})$ in importance sampling? In the following we analyze the quality and performance of this "approximate importance sampling" procedure.

Bias analysis. Suppose $\mathrm{supp}(u) \subseteq \mathrm{supp}(w)$. The following lemma analyzes the bias of the approximate importance sampling estimator: $\mathrm{AIS}(\mathrm{Q}, \mathrm{X}) = f(\mathrm{X}) \cdot u(\mathrm{Q}, \mathrm{X})$.

LEMMA 4.1. $\mathbb{E}_p(\mathrm{AIS}(Q, X)) =$

$$= \mathrm{Int}_\pi(f) \cdot \mathbb{E}_{\mu'}\left(\frac{u(Q,X)}{w(Q,X)}\right) + Z_\pi \cdot \mathrm{cov}_{\mu'}\left(f(X), \frac{u(Q,X)}{w(Q,X)}\right),$$

where μ' is the distribution induced by μ and Z_π is the normalization constant of π.

For lack of space, the proof of this lemma, as the proofs of all the other results in this paper, are postponed to the full version of the paper [3].

It follows from the lemma that there are two sources of bias in this estimator: (1) multiplicative bias, depending on the expectation of u/w under μ'; and (2) additive bias, depending on the correlation between f and u/w and on the normalization constant Z_π. Note that the multiplicative factor, even if small, may have a significant effect on the estimator's bias, and thus must be eliminated. The additive bias is typically less significant, as in many practical situations f and u/w are uncorrelated (e.g., when f is a constant function as is the case with corpus size estimation).

For a query-document pair (q, x), the ratio $u(\mathrm{q}, \mathrm{x})/w(\mathrm{q}, \mathrm{x})$ is called the *weight skew at* (q, x). The multiplicative bias factor is the expected weight skew under the target distribution μ'. In order to eliminate this bias, we need to somehow estimate the expected weight skew. For now, let us assume we have some unbiased estimator WSE for $\mathbb{E}_{\mu'}\left(\frac{u(\mathrm{Q},\mathrm{X})}{w(\mathrm{Q},\mathrm{X})}\right)$ (WSE may depend on the same sample (Q, X) used by the importance sampling estimator). It follows from Lemma 4.1 that:

$$\frac{\mathbb{E}_p(\mathrm{AIS}(\mathrm{Q},\mathrm{X}))}{\mathbb{E}(\mathrm{WSE})} \;=\; \mathrm{Int}_\pi(f) + \frac{Z_\pi \cdot \mathrm{cov}_{\mu'}\left(f(\mathrm{X}), \frac{u(\mathrm{Q},\mathrm{X})}{w(\mathrm{Q},\mathrm{X})}\right)}{\mathbb{E}_{\mu'}\left(\frac{u(\mathrm{Q},\mathrm{X})}{w(\mathrm{Q},\mathrm{X})}\right)}.$$

Thus, the ratio of the expectations of the two estimators, AIS and WSE, gives us the desired result ($\mathrm{Int}_\pi(f)$), modulo an additive bias factor. Ignoring for the moment this additive bias, it would seem that a good estimator for $\mathrm{Int}_\pi(f)$ is the ratio $\frac{\mathrm{AIS}}{\mathrm{WSE}}$. However, there is one problem: the expectation of a ratio is not the ratio of the expectations, i.e., $\mathbb{E}\left(\frac{\mathrm{AIS}}{\mathrm{WSE}}\right) \neq \frac{\mathbb{E}(\mathrm{AIS})}{\mathbb{E}(\mathrm{WSE})}$.

To solve this problem, we resort to a well-known trick from statistics: if we replace the numerator and the denominator by *averages* of multiple independent instances of the numerator estimator and of the denominator estimator, the difference between the expected ratio and the ratio of expectations can be diminished to 0. This idea is formalized by the following theorem:

THEOREM 4.2. *Suppose A and B are two estimators such that $\frac{\mathbb{E}(A)}{\mathbb{E}(B)} = I$. Let $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$ be n independent instances of A and B, respectively. Then,*

$$\left|\,\mathbb{E}\left(\frac{\frac{1}{n}\sum_{i=1}^n A_i}{\frac{1}{n}\sum_{i=1}^n B_i}\right) - I\right| \;\leq\; \frac{1}{n}\cdot\left(I\cdot\frac{\mathrm{var}(B)}{\mathbb{E}^2(B)} + \frac{|\,\mathrm{cov}(A,B)|}{\mathbb{E}^2(B)}\right) + o\left(\frac{1}{n}\right).$$

We can therefore define the approximate ratio importance sampling estimator for $\mathrm{Int}_\pi(f)$ as follows:

$$\mathrm{ARIS} \;\triangleq\; \frac{\frac{1}{n}\sum_{i=1}^n f(\mathrm{X}_i)\cdot u(\mathrm{Q}_i,\mathrm{X}_i)}{\frac{1}{n}\sum_{i=1}^n \mathrm{WSE}_i},$$

where $(\mathrm{Q}_1, \mathrm{X}_1), \ldots, (\mathrm{Q}_n, \mathrm{X}_n)$ are n independent samples from the trial distribution p and $\mathrm{WSE}_1, \ldots, \mathrm{WSE}_n$ are n independent estimators of the weight skew (WSE_i may depend on $(\mathrm{Q}_i, \mathrm{X}_i)$). Using Lemma 4.1 and Theorem 4.2, we can analyze the bias of this estimator:

LEMMA 4.3. *If $\mathbb{E}(\mathrm{WSE}) = \mathbb{E}_{\mu'}\left(\frac{u(Q,X)}{w(Q,X)}\right)$, then*

$$|\,\mathbb{E}_p(\mathrm{ARIS}) - \mathrm{Int}_\mu(f)| \;\leq\; \frac{Z_\pi\cdot\mathrm{cov}_{\mu'}\left(f(X),\frac{u(Q,X)}{w(Q,X)}\right)}{\mathbb{E}_{\mu'}\left(\frac{u(Q,X)}{w(Q,X)}\right)} + O\left(\frac{1}{n}\right),$$

where the $O(1/n)$ term suppresses constant factors that depend on $\mathrm{var}(\mathrm{WSE})$ *and on* $\mathrm{cov}(f(X)\cdot u(Q,X), \mathrm{WSE})$.

We conclude from the lemma that if we use sufficiently many samples, then we are likely to get an estimate of $\mathrm{Int}_\pi(f)$, which has only additive bias that depends on the correlation between f and u/w.

5. TWO ESTIMATORS

In this section we describe two variants of the approximate ratio importance sampling estimator (ARIS) discussed above. The two estimators, the Accurate Estimator (AccEst) and the Efficient Estimator (EffEst), offer different tradeoffs between accuracy and efficiency. The former has lower bias, while the latter is more efficient.

The estimators differ in the choice of the approximate importance weight function $u(\mathrm{q}, \mathrm{x})$ and in the expected weight skew estimator WSE. Before we show how u and WSE are defined in each of the estimators, let us rewrite the importance weights:

$$w(\mathrm{q},\mathrm{x}) \;=\; \frac{\pi(\mathrm{x})\cdot|\mathcal{P}_+|\cdot\mathrm{vcard}(\mathrm{q})}{\mathrm{vdeg}(\mathrm{x})} \;=\; \frac{\pi(\mathrm{x})\cdot|\mathcal{P}_+|\cdot\mathrm{vcard}(\mathrm{q})}{\mathrm{pdeg}(\mathrm{x})\cdot\mathrm{vdensity}(\mathrm{x})}.$$

Of the different terms that constitute the weight, the three we may not know a priori are $\pi(\mathrm{x})$, $|\mathcal{P}_+|$, and $\mathrm{vdensity}(\mathrm{x})$. $\mathrm{vcard}(\mathrm{q})$ is known, because we always obtain the pair (q, x) after having submitted q to the search engine and extracting its valid results. $\mathrm{pdeg}(\mathrm{x})$ is known, because we can extract the predicted queries from the content of x.

5.1 The Accurate Estimator

The Accurate Estimator (AccEst) uses approximate weights $u_{\mathrm{acc}}(\mathrm{q}, \mathrm{x})$ that are equal to the exact weights $w(\mathrm{q}, \mathrm{x})$, up to a constant factor. It follows that $u_{\mathrm{acc}}(\mathrm{q}, \mathrm{x})/w(\mathrm{q}, \mathrm{x})$ is constant, and hence the correlation between f and u_{acc}/w is 0. Using Lemma 4.3, the bias of AccEst is then only $O(1/n)$.

How do we come up with approximate weights that equal the exact weights up to a constant factor? Well, we are unable to efficiently do this with deterministic approximate weights, but rather with *probabilistic* ones. That is, AccEst uses a probabilistic weight function $u_{\mathrm{acc}}(\mathrm{q}, \mathrm{x})$, for which $\mathbb{E}(u_{\mathrm{acc}}(\mathrm{q}, \mathrm{x})) = const \cdot w(\mathrm{q}, \mathrm{x})$. As our analysis for approximate importance sampling easily carries over to probabilistic weights as well, we can still apply Lemma 4.3 and obtain the desired bound on the bias of AccEst.

The approximate weights are defined as follows:

$$u_{\mathrm{acc}}(\mathrm{q},\mathrm{x}) \;\triangleq\; \frac{\hat{\pi}(\mathrm{x})\cdot|\mathcal{P}|\cdot\mathrm{vcard}(\mathrm{q})\cdot\mathrm{IVD}(\mathrm{x})}{\mathrm{pdeg}(\mathrm{x})}.$$

That is, $\pi(\mathrm{x})$ is approximated by $\hat{\pi}(\mathrm{x})$ (they are the same, if $\mathrm{Int}_\pi(f)$ is a sum metric), $|\mathcal{P}_+|$ is approximated by $|\mathcal{P}|$, and the term $1/\,\mathrm{vdensity}(\mathrm{x})$ is estimated probabilistically by the "Inverse Validity Density Estimator" (IVD) described below. Note that apart from the computation of $\mathrm{IVD}(\mathrm{x})$, computing $u_{\mathrm{acc}}(\mathrm{q}, \mathrm{x})$ requires no search engine queries.

Figure 2 shows a procedure for estimating $1/\,\mathrm{vdensity}(\mathrm{x})$ for a given document x, using a limited number of queries.

The procedure repeatedly samples queries uniformly at random from the set of predicted queries pqueries(x). It submits each query to the search engine and checks whether they are valid for x. The procedure stops when reaching the first valid query and returns the number of queries sampled so far. As this number is geometrically distributed with vdensity(x) as the success parameter, the expectation of this estimator is exactly $1/\,\text{vdensity(x)}$. Note that the procedure is always guaranteed to terminate, because we apply it only on documents x for which vdensity(x) > 0.

```
1: Function InverseValidityDensityEstimator(S, P, x)
2:   pqueries(x) := predicted queries for x
3:   i := 1
4:   while (true) do
5:     Q := uniformly chosen query from pqueries(x)
6:     submit Q to S
7:     results(Q) := results returned by S
8:     if (Q does not overflow and x ∈ results(Q)) return i
9:     i := i + 1
```

Figure 2: Estimator for the inverse of the validity density

The expectation of $u_{\text{acc}}(\text{q},\text{x})$ is analyzed as follows:

$$\begin{aligned}\mathbb{E}(u_{\text{acc}}(\text{q},\text{x})) &= \mathbb{E}\left(\frac{\hat{\pi}(\text{x})\cdot|\mathcal{P}|\cdot\text{vcard(q)}\cdot\text{IVD(x)}}{\text{pdeg(x)}}\right)\\ &= \frac{|\mathcal{P}|}{|\mathcal{P}_+|}\cdot\frac{\hat{\pi}(\text{x})\cdot|\mathcal{P}_+|\cdot\text{vcard(q)}\cdot\mathbb{E}(\text{IVD(x)})}{\text{pdeg(x)}}\\ &= \frac{|\mathcal{P}|}{|\mathcal{P}_+|}\cdot\frac{Z_{\hat{\pi}}}{Z_\pi}\cdot w(\text{q},\text{x}).\end{aligned}$$

Hence, the expectation of $u_{\text{acc}}(\text{q},\text{x})$ equals the weight $w(\text{q},\text{x})$, up to the unknown multiplicative constant $\frac{|\mathcal{P}|}{|\mathcal{P}_+|}\cdot\frac{Z_{\hat{\pi}}}{Z_\pi}$. It immediately follows that also the expected weight skew is $\frac{|\mathcal{P}|}{|\mathcal{P}_+|}\cdot\frac{Z_{\hat{\pi}}}{Z_\pi}$ and that $\text{cov}_{\mu'}(f(\text{X}),u_{\text{acc}}(\text{Q},\text{X})/w(\text{Q},\text{X})) = 0$. How we obtain an unbiased estimator WSE for the expected weight skew is different between the case $\text{Int}_\pi(f)$ is a sum a metric and the case it is an average metric.

Case 1: $\text{Int}_\pi(f)$ is a sum metric. Here, the π-oracle computes the target measure π explicitly. That is, $\hat{\pi}=\pi$ and $\frac{Z_{\hat{\pi}}}{Z_\pi}=1$, so the only term we need to estimate is $\frac{|\mathcal{P}|}{|\mathcal{P}_+|}$.

If we sample a query Q′ uniformly at random from $\mathcal{P}$, it has a probability of $\frac{|\mathcal{P}_+|}{|\mathcal{P}|}$ to have at least one valid result. Therefore, we can estimate $\frac{|\mathcal{P}|}{|\mathcal{P}_+|}$ as follows: repeatedly sample queries uniformly at random from $\mathcal{P}$ and submit them to the search engine; stop when reaching the first query that has at least one valid result; the number of queries submitted is an unbiased estimator of $\frac{|\mathcal{P}|}{|\mathcal{P}_+|}$.

Case 2: $\text{Int}_\pi(f)$ is an average metric. In this case π is a distribution, and thus $Z_\pi=1$. Therefore, the expected weight skew is $\frac{|\mathcal{P}|}{|\mathcal{P}_+|}\cdot Z_{\hat{\pi}}$. As $Z_{\hat{\pi}}$ is not known in advance, we cannot use the same expected weight skew estimator as above. On the other hand, we observe that since π is a distribution, then the approximate weight $u_{\text{acc}}(\text{Q},\text{X})$ itself, where $(\text{Q},\text{X})\sim p$, is an unbiased estimator of the expected weight skew. This follows from Lemma 4.1 with $f\equiv 1$:

PROPOSITION 5.1.

$$E_p(u_{acc}(Q,X)) = Z_\pi\cdot\mathbb{E}_{\mu'}\left(\frac{u_{acc}(Q,X)}{w(Q,X)}\right).$$

As $Z_\pi=1$, $u_{\text{acc}}(\text{Q},\text{X})$ is indeed an unbiased estimator of the expected weight skew.

Analysis. By Lemma 4.3, the bias of the estimator is at most $\frac{Z_\pi\cdot\text{cov}_{\mu'}\left(f(\text{X}),\frac{u_{\text{acc}}(\text{Q},\text{X})}{w(\text{Q},\text{X})}\right)}{\mathbb{E}_{\mu'}\left(\frac{u_{\text{acc}}(\text{Q},\text{X})}{w(\text{Q},\text{X})}\right)}+O\left(\frac{1}{n}\right)$. Recall that the covariance term is 0, and thus the bias is only $O(1/n)$.

The cost of the computing $u_{\text{acc}}(\text{q},\text{x})$ is dominated by the cost of computing the inverse validity density. This computation requires $O(1/\,\text{vdensity(x)})$ queries to the search engine in expectation. Complete analysis of the efficiency of the estimator is postponed to the full version of the paper.

5.2 The Efficient Estimator

The Efficient Estimator (EffEst) uses deterministic approximate weights, which require no queries to the search engine to compute, similarly to the estimators of [2, 6]:

$$u_{\text{eff}}(\text{q},\text{x}) \triangleq \frac{\hat{\pi}(\text{x})\cdot|\mathcal{P}|\cdot\text{vcard(q)}}{\text{pdeg(x)}}.$$

That is, $\pi(\text{x})$ is approximated by $\hat{\pi}(\text{x})$ (they are the same, if $\text{Int}_\pi(f)$ is a sum metric), $|\mathcal{P}_+|$ is approximated by $|\mathcal{P}|$, and the term $1/\,\text{vdensity(x)}$ is ignored. The weight skew in this case is characterized as follows:

PROPOSITION 5.2. $u_{\mathit{eff}}(q,x)/w(q,x) = \frac{|\mathcal{P}|}{|\mathcal{P}_+|}\cdot Z_{\hat{\pi}}\cdot\text{vdensity}(x)$.

The estimator for the expected weight skew is again different between the case $\text{Int}_\pi(f)$ is a sum metric and the case $\text{Int}_\pi(f)$ is an average metric.

Case 1: $\text{Int}_\pi(f)$ is a sum metric. By Proposition 5.1, $u_{\text{eff}}(\text{Q},\text{X})$, where $(\text{Q},\text{X})\sim p$, is an unbiased estimator of the expected weight skew, modulo the constant factor Z_π:

$$\mathbb{E}_p(u_{\text{eff}}(\text{Q},\text{X})) = Z_\pi\cdot\mathbb{E}_{\mu'}\left(\frac{u_{\text{eff}}(\text{Q},\text{X})}{w(\text{Q},\text{X})}\right).$$

In our case π is not a distribution, so Z_π is unknown. Hence, $u_{\text{eff}}(\text{Q},\text{X})$ is not sufficient by itself to estimate the expected weight skew. In order to obtain an estimator for the expected weight skew, we will divide $u_{\text{eff}}(\text{Q},\text{X})$ by an unbiased estimator for Z_π.

We observe that $Z_\pi=\text{Int}_\pi(\mathbf{1})$, where $\mathbf{1}$ is the constant 1 function. So by applying the Accurate Estimator with $f\equiv\mathbf{1}$, we can obtain an unbiased estimator of Z_π. We thus have: $\frac{\mathbb{E}_p(u_{\text{eff}}(\text{Q},\text{X}))}{\mathbb{E}(\text{AccEst})}=\mathbb{E}_{\mu'}\left(\frac{u_{\text{eff}}(\text{Q},\text{X})}{w(\text{Q},\text{X})}\right)$. So, using again Theorem 4.2, we can get a nearly unbiased estimator of the expected weight skew by averaging over multiple instances of $u_{\text{eff}}(\text{Q},\text{X})$ and AccEst:

$$\text{WSE}=\frac{\frac{1}{n}\sum_{i=1}^n u_{\text{eff}}(\text{Q}_i,\text{X}_i)}{\frac{1}{n}\sum_{i=1}^n\text{AccEst}_i}.$$

Here, $(\text{Q}_1,\text{X}_1),\ldots,(\text{Q}_n,\text{X}_n)$ are n independent samples from p and $\text{AccEst}_1,\ldots,\text{AccEst}_n$ are n independent Accurate Estimators for $\text{Int}_\pi(\mathbf{1})$.

At this point the reader may be wonder why the Efficient Estimator is more efficient than the Accurate Estimator. After all, the Efficient Estimator calls the Accurate Estimator! The rationale behind this is the following. Indeed, if we need to estimate only a single integral $\text{Int}_\pi(f)$, then the Efficient Estimator is less efficient than the Accurate Estimator. However, in practice, we usually need to compute multiple integrals $\text{Int}_\pi(f_1),\ldots,\text{Int}_\pi(f_t)$ w.r.t. the same target measure π. Note that the constant Z_π, for whose

estimation we used the Accurate Estimator, depends only on π and is independent of the target function f. Therefore, we can *reuse* the estimation of Z_π in the estimations of $\mathrm{Int}_\pi(f_1), \ldots, \mathrm{Int}_\pi(f_t)$. This implies that the *amortized* cost of the Efficient Estimator is lower than that of the Accurate Estimator.

Case 2: $\mathrm{Int}_\pi(f)$ is an average metric. In this case π is a distribution and thus $Z_\pi = 1$. Therefore, by Proposition 5.1, $u_{\mathrm{eff}}(\mathrm{Q},\mathrm{X})$, where $(\mathrm{Q},\mathrm{X}) \sim p$, is an unbiased estimator of the expected weight skew.

Analysis. By Lemma 4.3, the bias of the estimator is at most $\frac{Z_\pi \cdot \mathrm{cov}_{\mu'}\left(f(\mathrm{X}), \frac{u_{\mathrm{eff}}(\mathrm{Q},\mathrm{X})}{w(\mathrm{Q},\mathrm{X})}\right)}{\mathbb{E}_{\mu'}\left(\frac{u_{\mathrm{eff}}(\mathrm{Q},\mathrm{X})}{w(\mathrm{Q},\mathrm{X})}\right)} + O\left(\frac{1}{n}\right)$. By the characterization of the weight skew using the validity density (Proposition 5.2) and recalling that π' is the marginal distribution of μ' on $\mathcal{D}$ (Proposition 3.1), we can rewrite the bias as:

$$\frac{Z_\pi \cdot \mathrm{cov}_{\pi'}(f(\mathrm{X}), \mathrm{vdensity}(\mathrm{X}))}{E_{\pi'}(\mathrm{vdensity}(\mathrm{X}))} + O(1/n).$$

That is, as long as the target function is not correlated with the validity density of documents, the bias is low.

The Efficient Estimator is indeed efficient, because each approximate weight computation requires only fetching a single page from the web and no queries to the search engine. Complete analysis of the efficiency of the estimator is postponed to the full version of this paper.

6. RAO-BLACKWELLIZATION

There is some inherent inefficiency in the importance sampling estimators: although each random query they submit to the search engine returns many results, they use at most a single result per query. All other results are discarded. The corpus size estimator of Broder *et al.* [6] uses all query results, and not just one. We observe that what they did is an instance of the well-known Rao-Blackwellization technique for reducing estimation variance. We next show how to apply Rao-Blackwellization on our importance sampling estimators in a similar fashion.

Recall that the basic approximate importance sampling estimator is $\mathrm{AIS}(\mathrm{Q},\mathrm{X}) = f(\mathrm{X}) \cdot u(\mathrm{Q},\mathrm{X})$, where (Q,X) is a sample from the trial distribution p and $u(\mathrm{Q},\mathrm{X})$ is an approximate weight. Suppose now that instead of using only this single document in our basic estimator, we use all the query results:

$$\mathrm{AIS}^{\mathrm{RB}}(\mathrm{Q}) \;=\; \frac{1}{\mathrm{vcard}(\mathrm{Q})} \sum_{\mathrm{X} \in \mathsf{vresults}(\mathrm{Q})} f(\mathrm{X}) \cdot u(\mathrm{Q},\mathrm{X}).$$

Each instance of $\mathrm{AIS}^{\mathrm{RB}}$ is an average over several correlated instances of AIS. The main point is that computing these correlated instances in bulk can be done with a single query. The Rao-Blackwell theorem (cf. [7]) implies that $\mathrm{AIS}^{\mathrm{RB}}(\mathrm{Q})$ can be only better than $\mathrm{AIS}(\mathrm{Q},\mathrm{X})$ as an estimator of $\mathrm{Int}_\pi(f)$:

THEOREM 6.1 (RAO-BLACKWELL THEOREM). $\mathrm{AIS}^{\mathrm{RB}}$ *has the same bias as* AIS*:*

$$\mathbb{E}(\mathrm{AIS}^{\mathrm{RB}}(Q)) \;=\; \mathbb{E}(\mathrm{AIS}(Q,P)).$$

The variance of $\mathrm{AIS}^{\mathrm{RB}}$ *can only be lower:*

$$\mathrm{var}(\mathrm{AIS}^{\mathrm{RB}}(Q)) \;=\; \mathrm{var}(\mathrm{AIS}(Q,X)) - \mathbb{E}(\mathrm{var}(\mathrm{AIS}(Q,X)|Q)).$$

By the above theorem, the expected reduction in variance is $\mathbb{E}(\mathrm{var}(f(\mathrm{X}) \cdot u(\mathrm{Q},\mathrm{X}))|\mathrm{Q})$, where Q is a uniformly chosen query from $\mathcal{P}_+$ and X is a uniformly chosen document from Q. That is, the more variable are the results of queries w.r.t. the target function f, the higher are the chances that Rao-Blackwellization will help. In our empirical study we show that in practice Rao-Blackwellization can make a dramatic effect. See Section 7.

The variance reduction achieved by Rao-Blackwellization can lead to lower costs, as fewer instances of the estimator are needed in order to obtain a desired accuracy guarantee. On the other hand, each instance of the estimator requires many more weight and function calculations (as many as the number of results of the sampled query), and if these are very costly (as is the case with the Accurate Estimator), then the increase in cost per instance may outweigh the reduction in the number of instances, eventually leading to higher amortized costs. We conclude that Rao-Blackwellization should be used judiciously.

7. EXPERIMENTAL RESULTS

We conducted two sets of experiments. In the first set we performed comparative evaluation of the bias and amortized cost of our new estimators, of the rejection sampling estimator from our previous paper [2], and of the Broder *et al.* estimator [6]. To this end, we ran all these estimators on a local search engine that we built over 2.4 million English documents fetched from ODP [9]. As we have ground truth for this search engine, we could compare the measurements produced by the estimators against the real values.

The second set of experiments was conducted over three major real search engines. We used the Accurate Estimator to estimate the corpus size of each the search engines, with and without duplicate elimination. (More accurately, we estimated sizes of large subsets of the search engine corpora.)

Experimental setup. We used the same ODP search engine as in our previous paper [2]. The corpus of this search engine consists only of text, HTML, and pdf English-language documents from the ODP hierarchy. Each document was given a serial id and indexed by single terms and phrases. Only the first 10,000 terms in each document were considered. Exact phrases were not allowed to cross boundaries, such as paragraph boundaries. We used static ranking by serial id to rank query results.

In order to construct a query pool for the evaluation experiments, we split the ODP data set into two parts: a *training set*, consisting of every fifth page (when ordered by id), and a *test set*, consisting of the rest of the pages. We used the training set to create a pool of phrases of length 4. The measurements were done only on the test set.

The experiments on real search engines were conducted in February 2007. The pool used by our estimators was a pool of 1.75 billion phrases of lengths 3-5 extracted from the ODP data set (the entire data set, not just the test set).

Evaluation experiments. We compared the following 6 estimator configurations: (1) AccEst, without Rao Blackwellization; (2) AccEst, with Rao Blackwellization; (3) EffEst, without Rao Blackwellization; (4) EffEst, with Rao Blackwellization; (5) the *Broder et al.* estimator; (6) the rejection sampling estimator from our previous paper.

We used the estimators to measure two metrics: (1) corpus size (i.e., the size of the test set); (2) density of pages

in the test set about sports. (We used a simple keyword based classifier to determine whether a page is about sports or not.) Note that the first metric is a sum metric, while the second is an average metric. We did not use the rejection sampling estimator for estimating corpus size, as it can handle only average metrics. We did not use the Broder *et al.* estimator for estimating the density of sports pages, because it can handle only sum metrics.

In order to have a common baseline, we allowed each estimator to use exactly 1 million queries. Each estimator produced a different number of samples from these queries, depending on its amortized query cost.

We ran each experiment four times, with different values of the result set size limit k ($k = 5, 20, 100, 200$). This was done in order to track the dependence of the estimators' bias and cost on the density of overflowing queries (the lower k, the higher is the density).

Figure 3(a) compares the relative bias (bias divided by the estimated quantity) of our two estimators and the estimator of Broder *et al.* when measuring corpus size. Figure 3(b) compares the relative bias of our two estimators and the rejection sampling estimator when measuring density of sports pages. The results for the Rao-Blackwellized versions of these estimators are suppressed, since Rao-Blackwellization has no effect on bias.

The results for the corpus size clearly show that our estimators have no bias at all, while the estimator of Broder *et al* suffers from significant bias, which grows with the density of overflowing queries in the pool. For example, for $k = 5$, the relative bias of the Broder *et al.* estimator was about 75%, while the relative bias of our estimators was 0.01%. Note that since the target function is constant in this case, then its value has no correlation with the weight skew, which explains why also EffEst has no bias.

The results for the density of sports pages show that AccEst is unbiased, as expected. EffEst has small bias, which emanates from a weak correlation between the function value and the validity density. The rejection sampling method has a large observed bias, primarily because it produced a small number of samples and thus its variance is still high.

Figures 4(a) and 4(b) compare the amortized costs of the regular and the Rao-Blackwellized versions of our two estimators and of the rejection sampling estimator. We used a square root scale in order to fit all the bars in a single graph. The results clearly indicate that Rao-Blackwellization is effective in reducing estimation variance (and therefore also amortized costs) in both metrics and both estimators. For example, in corpus size estimation, when $k = 200$, Rao-Blackwellization reduced the amortized query cost of AccEst by 79% and the query cost of EffEst by 60%. Furthermore, the amortized cost of the rejection sampling estimator is tremendously higher than the amortized cost of our new estimators (even the non-Rao-Blackwellized ones). For example, when $k = 200$, Rao-Blackwellized EffEst was 375 times more efficient than rejection sampling!

Experiments on real search engines. We used our most accurate sampler, AccEst, to estimate the corpus sizes of three major search engines. For reference, we also ran the Broder *et al.* estimator with the same query pool. These measurements count duplicate pages as separate pages. We also measured the duplicate-free size of the corpus, by estimating the average number of duplicates each document in the corpus has. The results, together with confidence intervals, are plotted in Figure 5.

It can be seen that the estimations we got are far below the reported sizes of search engines. The main reason for this is that our estimators effectively measure the sizes of only subsets of the corpora—the indexed pages that match at least one phrase from the pool. As our pool consists of English-only phrases, pages that are not in English, not in HTML, pdf, or text format, or pages that are poor in text, are excluded from the measurement. A second reason is that search engines may choose sometimes not to serve certain pages, even though these pages exist in their index and match the query, e.g., because these pages are spam or duplicates.

Another observation we can make from the results is that overflowing queries really hurt the Broder *et al.* estimator also on live search engines. We note that like Broder *et al*, we filtered out from the query pool all phrases that occurred frequently (at least 10 times) in the ODP corpus. Even after this filtering, 3% of the queries overflowed on the first search engine, 10% on the second, and 7% on the third. The overflowing queries probably incurred high bias that made the estimates produced by the Broder *et al.* estimator to be much lower than ours.

Finally, the experiments reveal search engines deal differently with duplicates. In one of the search engines, the corpus size, after removing duplicates, shrunk in 28%, while in the other two it shrunk in 14% and 17%, respectively.

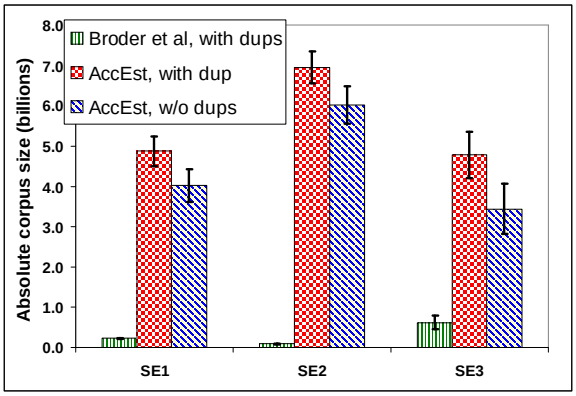
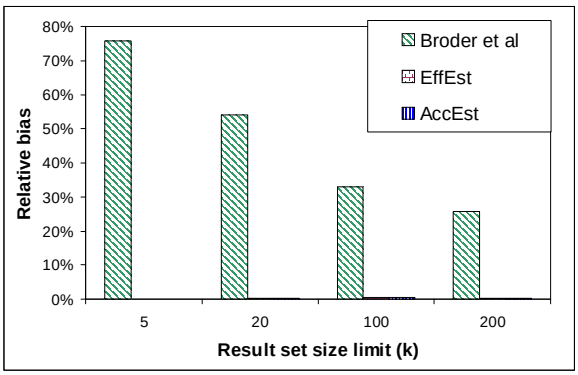


Figure 5: Corpus sizes of three major search engines, with and without elimination of duplicates.

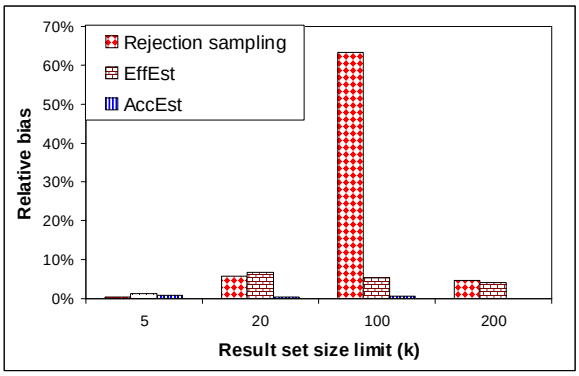
8. CONCLUSIONS

In this paper we presented two new estimators for search engine quality metrics that can be expressed as discrete integrals. Our estimators are able to overcome the "degree mismatch" problem and thereby be accurate and efficient at the same time. We show both analytically and empirically that our estimators beat recently proposed estimators [2, 6].

In designing our estimators we employ a combination of statistical tools, like importance sampling and Rao Blackwellization. By carefully analyzing the effect of approximate weights on the bias of importance sampling, we were able to design procedures to mitigate the bias. This bias-elimination technique for approximate importance sampling may be applicable in other scenarios as well.

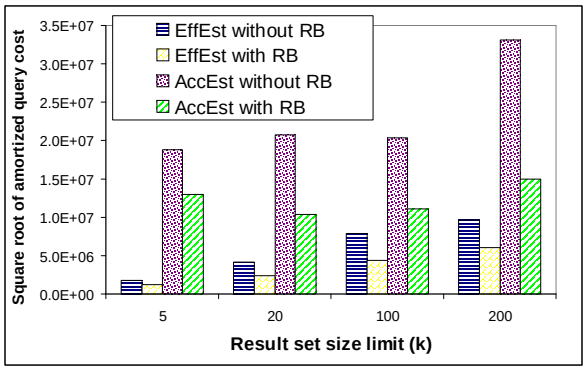


(a) Corpus size.

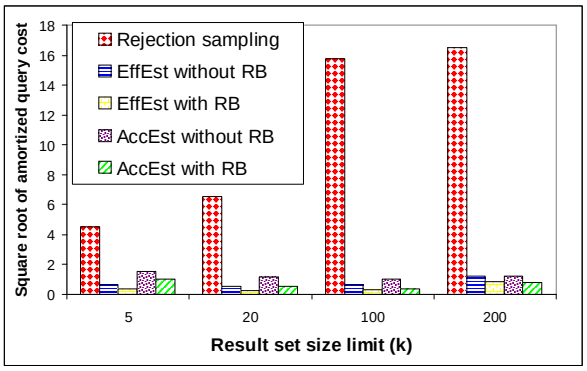


(b) Density of sports pages.

Figure 3: Relative bias of the estimators.



(a) Corpus size.



(b) Density of sports pages.

Figure 4: Square root of amortized query cost of the estimators.

9. REFERENCES

[1] Z. Bar-Yossef, A. Berg, S. Chien, J. Fakcharoenphol, and D. Weitz. Approximating aggregate queries about Web pages via random walks. In *Proc. 26th VLDB*, pages 535–544, 2000.

[2] Z. Bar-Yossef and M. Gurevich. Random sampling from a search engine's index. In *Proc. 15th WWW*, pages 367–376, 2006.

[3] Z. Bar-Yossef and M. Gurevich. Efficient search engine measurements, 2007. Full version available at `http://www.ee.technion.ac.il/people/zivby`.

[4] K. Bharat and A. Broder. A technique for measuring the relative size and overlap of public Web search engines. In *Proc. 7th WWW*, pages 379–388, 1998.

[5] E. T. Bradlow and D. C. Schmittlein. The little engines that could: Modeling the performance of World Wide Web search engines. *Marketing Science*, 19:43–62, 2000.

[6] A. Broder, M. Fontoura, V. Josifovski, R. Kumar, R. Motwani, S. Nabar, R. Panigrahy, A. Tomkins, and Y. Xu. Estimating corpus size via queries. Proc. 15th CIKM, 2006.

[7] G. Casella and C. P. Robert. Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996.

[8] M. Cheney and M. Perry. A comparison of the size of the Yahoo! and Google indices. Available at `http://vburton.ncsa.uiuc.edu/indexsize.html`, 2005.

[9] dmoz. The open directory project. `http://dmoz.org`.

[10] A. Dobra and S. E. Fienberg. How large is the World Wide Web? *Web Dynamics*, pages 23–44, 2004.

[11] O. Goldreich. A sample of samplers - a computational perspective on sampling (survey). *ECCC*, 4(20), 1997.

[12] A. Gulli and A. Signorini. The indexable Web is more than 11.5 billion pages. In *Proc. 14th WWW*, pages 902–903, 2005.

[13] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[14] M. R. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork. Measuring index quality using random walks on the Web. In *Proc. 8th WWW*, pages 213–225, 1999.

[15] M. R. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork. On near-uniform URL sampling. In *Proc. 9th WWW*, pages 295–308, 2000.

[16] T. C. Hesterberg. *Advances in Importance Sampling*. PhD thesis, Stanford University, 1988.

[17] S. Lawrence and C. L. Giles. Searching the World Wide Web. *Science*, 5360(280):98, 1998.

[18] S. Lawrence and C. L. Giles. Accessibility of information on the Web. *Nature*, 400:107–109, 1999.

[19] S.-M. Lee and A. Chao. Estimating population size via sample coverage for closed capture-recapture models. *Biometrics*, 50(1):88–97, 1994.

[20] J. S. Liu. Monte Carlo Strategies in Scientific Computing. Springer, 2001.

[21] A. W. Marshal. The use of multi-stage sampling schemes in Monte Carlo computations. In *Symposium on Monte Carlo Methods*, pages 123–140, 1956.

[22] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *J. of Chemical Physics*, 21:1087–1091, 1953.

[23] P. Rusmevichientong, D. Pennock, S. Lawrence, and C. L. Giles. Methods for sampling pages uniformly from the World Wide Web. In *Proc. AAAI Symp. on Using Uncertainty within Computation*, 2001.

[24] J. von Neumann. Various techniques used in connection with random digits. In *John von Neumann, Collected Works*, volume V. Oxford, 1963.