Distinct Values Estimators for Power Law Distributions

Rajeev Motwani* Sergei Vassilvitskii†

Abstract

The number of distinct values in a relation is an important statistic for database query optimization. As databases have grown in size, scalability of distinct values estimators has become extremely important, since a naïve linear scan through the data is no longer feasible. An approach that scales very well involves taking a sample of the data, and performing the estimate on the sample. Unfortunately, it has been shown that obtaining estimators with guaranteed small error bounds requires an extremely large sample size in the worst case. On the other hand, it is typically the case that the data is not worst-case, but follows some form of a Power Law or Zipfian distribution. We exploit data distribution assumptions to devise distinct-values estimators with analytic error guarantees for Zipfian distributions. Our estimators are the first to have the required number of samples depend only on the number of distinct values present, D, and not the database size, n. This allows the estimators to scale well with the size of the database, particularly if the growth is due to multiple copies of the data. In addition to theoretical analysis, we also provide experimental evidence of the effectiveness of our estimators by benchmarking their performance against previously best known heuristic and analytic estimators on both synthetic and real-world datasets.

1 Introduction

The number of distinct values of an attribute in a relation is one of the critical statistics necessary for effective query optimization. It is well-established [9] that a bad estimate to the number of distinct values can slow down the query execution time by several orders of magnitude. Unfortunately, as the amount of data stored in a database increases, this vital statistic becomes increasingly difficult to estimate quickly with reasonable accuracy. While the exact number of distinct values in a column can be determined by a full scan of the table, query optimizers would like to obtain a (low-error) estimate with significantly lower effort. Even if one is willing to perform a full scan, determining the exact number of distinct values requires significant memory overhead.

Several approaches have been considered in the literature to deal with this issue. Recently, much of the work has focused on streaming models, or algorithms which are allowed to take only a single pass over the data [1, 5, 6]. The challenge for these algorithms lies in minimizing the space used, since the naïve schemes run out of memory long before a single scan is complete. Another natural approach is to take a small random sample from the large dataset (often on the order of 1-10%) and then to estimate the number of distinct values from the sample. This problem has a rich history in statistics [2, 8, 19], but the statistical methods are essentially heuristic and in any case do not perform well in the context of databases [12, 13]. There has been some recent work in database literature [3, 7, 9, 10] on trying to devise good distinct-values estimators for random samples; but again, these are mostly based on heuristics and are not supported by analytic error guarantees.

An explanation for the apparent difficulty of distinct-values estimation was provided in the powerful negative result of Charikar, Chaudhuri, Motwani, and Narasayya [3]. They demonstrate two data distribution scenarios where the numbers of distinct values differ dramatically, yet a large number of random samples is required to distinguish between the two scenarios. For example, to guarantee that an estimate has less than 10% error with high probability, requires sampling almost the entire table. While this negative result explains the difficulty of obtaining estimators with good analytic error guarantees, the worst case scenarios rarely occur in practice. This leaves open the possibility of exploiting our knowledge of real-world data distributions to obtain estimators that are efficient and scalable, have analytic error guarantees, and perform well in practice. Indeed, in this paper we show that such positive results are possible once we make some assumptions about the underlying data distribution, thereby allowing us to circumvent the seemingly crippling negative result of Charikar et al. [3].

It has been observed for over a half-century that many large datasets follow a Power Law (also known as Zipfian) distribution; for example, the distribution

*Stanford University. Supported in part by NSF Grants EIA-0137761 and ITR-0331640, and grants from Media-X and SNRC.

†Stanford University. Supported in part by NSF Grants EIA-0137761 and ITR-0331640, and grants from Media-X and SNRC.

of words in a natural language [20] or the distribution of the (out-)degrees in the web graph [14]. We refer the reader to the book by Knuth [15] and the survey article by Mitzenmacher [17] for further examples and an in-depth discussion. The underlying reasons for the ubiquity of this class of data distributions have been a subject of debate ever since the original paper by Zipf [20], but as Mitzenmacher points out one thing is clear: "Power law distributions are now pervasive in computer science."

1.1 Our Results In this work, we assume that the n data items in the column of interest follow a Zipfian distribution (with some skew parameter θ) on some number of distinct elements D. Our estimators work on a random sample of the data from the column. We assume that the value of θ is known ahead of time. In our experimental tests we show that the value of θ can be easily estimated on real world datasets using linear regression techniques. Of course, the value of D is assumed to be unknown, since that is precisely the quantity that we seek to estimate.

A key feature of the algorithms that we propose is the *independence* of their running time from the database size. These are the first algorithms where the number of samples and the running time are a function of solely the number of *distinct* elements present and not of the database size itself. This property allows our estimators to scale extremely well. In particular, the running time of the estimators remains the same if the database contains multiple copies of the same data.

We propose two algorithms for computing the number of distinct values. The first algorithm samples adaptively until a stopping condition is met. We prove that for a large family of distributions the algorithm returns D, the *exact* number of distinct values with high probability; and requires no more than $O(\log D/p_D)$ samples, where p_D is the probability of selecting the least likely element. Observe that if the underlying distribution is uniform, coupon-collector arguments provide a matching lower bound for the required number of samples.

The setting for the second algorithm is slightly different. In some applications we are not able to adaptively sample, but rather are presented with a small fraction of the database and are asked to provide the best possible estimate. In this setting the second algorithm returns $\widehat{D}$ a $(1+\epsilon)$ approximation to D with high probability after examining only this small number of random samples. In particular we analyze our algorithm for Zipfian distributions where the estimator is correct with probability $1 - \exp(-\Omega(D)/\epsilon^2)$ after examining roughly $1/p_D$ samples.

We demonstrate via experiments that our estimators not only have theoretical error guarantees, but also outperform previously-known estimators on synthetic and real world inputs.

The rest of this paper is organized as follows. We begin in Sections 2 and 3 by formally defining the problem and presenting the goals that an estimator should strive to achieve. In Section 4 we present an algorithm for computing the exact number of distinct values in a database with high probability. Section 5 introduces an estimator that returns an ϵ-error approximation to the true number of distinct values. Finally, in Section 6 we present experimental results comparing the performance of our estimator best known heuristic and guaranteed error estimators, on both synthetic and real world data.

2 Preliminaries

We assume the following set-up throughout the paper. Let $\mathcal{F} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots\}$ be a family of probability distributions, where $\mathcal{P}_j$ is a distribution on j elements. Let R be a relation on n rows, and assume that the elements in R are distributed along some $\mathcal{P}_{j^*} \in \mathcal{F}$. Our goal is to determine j^*, the number of distinct elements present in R.

To simplify notation, given a distribution $\mathcal{P}$ over a set X and an element $x \in X$, denote by $Pr_i(\mathcal{P})$ the probability of element $i \in X$ in the distribution $\mathcal{P}$.

As a special case we consider the family of Zipfian or Power Law distributions parametrized by their skew, θ. $\mathcal{Z}_\theta = \{Z_{1,\theta}, Z_{2,\theta}, \ldots\}$. where $Z_{D,\theta}$ is the Zipfian distribution of parameter θ on D elements defined as follows. Rank the elements 1 through D in decreasing order of probability mass. Then the probability of selecting the i^{th} element is

$$Pr_i(Z_{D,\theta}) = \frac{1}{i^\theta N_{D,\theta}},$$

$$\text{where } N_{D,\theta} = \sum_{i=1}^{D} \frac{1}{i^\theta} \text{ is a normalizing constant}$$

Observe that when $\theta = 0$, the Zipfian distribution is simply the uniform distribution and $N_{D,0} = D$. The skew in the distribution increases with θ. In real-life applications θ is typically less than 2.

We assume that the value of θ is known to our algorithm; in practice a good estimate for the value of θ can be obtained as a part of the sampling process as discussed in Section 6.1.

Our estimation algorithm will deliver an estimate $\widehat{D}$ of the number of distinct values in the column. To evaluate the performance of the estimators, we will use *ratio error*, which is the multiplicative error of $\widehat{D}$ with

respect to D. Formally, we define ratio error as

$$\max\left(\frac{\widehat{D}}{D}, \frac{D}{\widehat{D}}\right)$$

Under this definition, the error is always at least 1, and no distinction is made between underestimates and overestimates of the number of distinct values.

The following notation will be useful: After some r samples, let $f_{in}(r)$ be the number of distinct values that appeared in the sample and $f_{out}(r, D) = D - f_{in}$, the number of distinct values that were not part of the sample.

3 Estimators

Our goal is to obtain distinct-values estimators with the following desired properties:

Few Samples: The number of samples required for good performance by the estimator should be small.

Error Guarantees: The estimator should be backed by analytical error guarantees.

Scalability: The estimator should scale well as the database size n increases. This implies that the number of samples should grow sublinearly with (or ideally be independent of) the database size.

As mentioned earlier, the vast majority of estimators that operate on a random sample of the data (as opposed to those which perform a full scan of the data) do not provide any analytical guarantees for their performance. The exception is the GEE (Guaranteed-Error Estimator) estimator developed by Charikar et al. [3].

THEOREM 3.1. ([3]) *Using a random sample of size r from a table of size n, the expected ratio error for GEE is $O(\sqrt{n/r})$.*

We note that the result above is the best possible due to a matching lower bound described by the authors. Observe that the bound is quite weak — even if we allow a sample of 10%, the expected ratio error bound can be as high as $\sqrt{10} \approx 3.2$. Further, the GEE estimator does not scale well — to maintain the same error, the sample size needs to increase linearly with the size of the database.

Once we assume that the input distribution follows a Zipfian distribution with unknown parameter D, we can develop estimators which greatly improve upon the GEE estimator presented above. We focus first on determining exactly the number of distinct values in the database, and then we relax this requirement to devise estimators which may return a $(1+\epsilon)$-error approximation to the number of distinct values.

4 Exact Algorithm

We first seek to devise an algorithm which will return $\hat{D} = D$, but is allowed to fail with some small probability δ. Note that without the knowledge of the data distribution, the situation is grim — in the worst case we would need to sample a large fraction of the database to obtain the value of D with bounded error probability.

We begin by defining a notion of *c-regular* families of distributions.

DEFINITION 4.1. *A distribution family $\mathcal{F} = \{P_1, P_2, \ldots\}$ is called* c-regular *if the following two conditions hold:*

- **Monotonicity:** For any $i, 1 \leq j \leq i$: $Pr_i(P_j) \geq Pr_i(P_{j+1})$.
- **c-Regularity:** For any i: $Pr_i(P_i) \leq c \cdot Pr_{i+1}(P_{i+1})$.

The *monotonicty* condition ensures that the probability of an individual item i in the support, decreases as the overall support of the distribution increases. The *c-regularity* condition bounds the decrease in mass of the least weighted element.

Many common distribution families are c-regular for small values of c. For example, the family of uniform distributions is 2-regular. The family of Zipfian distributions of parameter θ is 5-regular for $\theta \leq 2$

To simplify notation, for a multiset S, let $Distinct(S)$ be the number of distinct values appearing in S.

The *Exact Count* algorithm presented below will continue to sample until a particular stopping condition is met, at which point the sample contains all of the distinct values with high probability.

Algorithm 1 Exact Count Algorithm

Let $Stop(t) = \frac{6\ln(t+1)}{Pr_{t+1}\mathcal{P}_{t+1}}$
Let $S \subseteq R$ denote the current sample
Draw a sample of size $Stop(3)$
while $Stop(Distinct(S)) > |S|$ **do**
  Increase S until $|S|$ grows by a factor of $c \log_3 4$
end while
Output $\hat{D} = Distinct(S)$

4.1 Analysis We will show that the above algorithm returns $\hat{D} = D$ with probability at least 1/2.

LEMMA 4.1. *Let S be a sample of size at lest $Stop(3)$ drawn uniformly at random from R. Let t be such that $Stop(t+1) \geq |S| > Stop(t)$. If $t < D$ then $Pr[Distinct(S) < t+1] \leq (t+1)^{-2}$.*

Before proving the lemma, let us interpret its meaning. Observe that the algorithm halts when $Distinct(S) \leq t$. This lemma bounds the probability of early halting (halting when $Distinct(S) \neq D$).

Proof. Consider the elements $1, 2, ..., D$. For the sake of the proof suppose we can split them up into $t+2$ groups with the following properties:

1. each element appears in exactly one group.
2. For groups $j \in \{1, ..., t+1\}$:

$$Pr_j(\mathcal{P}_{t+1}) \geq \sum_{i \in group\ j} Pr_i(\mathcal{P}_D) \geq \frac{Pr_j(\mathcal{P}_{t+1})}{2}$$

.

Let $\mathcal{E}$ be the event that $Distinct(S) < t+1$. For $\mathcal{E}$ to occur, all of the elements from at least one of the first $t+1$ groups can not present in S. Let $\mathcal{E}_j$ be the event that no element from group j appears in the sample.

$$\begin{aligned} Pr[\mathcal{E}_j] &\leq (1 - Pr_j(\mathcal{P}_{t+1})/2)^{|S|} \\ &\leq \exp(-\frac{Pr_j(\mathcal{P}_{t+1})}{2Pr_{t+1}(\mathcal{P}_{t+1})} \cdot (6\ln(t+1))) \\ &\leq \frac{1}{(t+1)^3} \end{aligned}$$

Where the last inequality follows since $Pr_j(\mathcal{P}_{t+1}) \geq Pr_{t+1}(\mathcal{P}_{t+1})$. For event $\mathcal{E}$ to occur one of the events $\mathcal{E}_1, \ldots, \mathcal{E}_{t+1}$ must occur. We can use the union bound to limit $Pr[\mathcal{E}]$. In particular $Pr[\mathcal{E}] \leq \sum_{i=1}^{t+1}(t+1)^{-3} = (t+1)^{-2}$.

It remains to show how the elements are broken up into the aforementioned groups.

We can achieve this division using the following simple algorithm. Consider the elements in order of decreasing probability: $Pr_1(\mathcal{P}_D), Pr_2(\mathcal{P}_D), \ldots, Pr_D(\mathcal{P}_D)$. Put the first element into the first group. The monotonicity property ensures that it will fit. Continue with elements of weight $Pr_2(\mathcal{P}_D), Pr_3(\mathcal{P}_D)$, etc. Once an element no longer fits into the first group, begin filling the next group. Again, by the monotonicity property, the first element will fit in the group. After filling the first $t+1$ groups, put the remaining elements into group $t+2$.

It is clear that each element will be present in exactly one group. Further, each group $\in \{1, \ldots, t+1\}$ is at least half full, since it contains at least one element, and the elements are considered in order of decreasing weight.

To conclude the analysis we need to bound the total number of times that the event $\mathcal{E}$ can occur. We do this by showing that the value of t increases every time we evaluate the stopping condition.

LEMMA 4.2. *For $|S| \geq Stop(2)$, $Stop^{-1}(c\log_3 4 \cdot |S|) - Stop^{-1}(|S|) \geq 1$, where $Stop^{-1}(U) = \min_u\{u : Stop(u) > U\}$.*

Proof. We will prove this by bounding the ratio $\frac{Stop(t+1)}{Stop(t)}$ for $t \geq 2$.

$$\begin{aligned} \frac{Stop(t+1)}{Stop(t)} &= \frac{\ln(t+2)}{\ln(t+1)} \cdot \frac{P_{t+1}(t+1)}{P_{t+2}(t+2)} \\ &\leq \log_3 4 \cdot c \end{aligned}$$

Where the second inequality follows from the c-regularity condition.

THEOREM 4.1. *The Exact-Count algorithm terminates successfully with probability at least 1/2. Moreover, the number of samples necessary is $O(\log D / Pr_D(\mathcal{P}_D))$.*

Proof. The algorithm can potentially fail only when the stopping condition is evaluated. Let E_i be the event that the algorithm halts on the i^{th} evaluation, but the sample does not yet contain all of the distinct values. Lemma 4.1 implies that at the point of i^{th} evaluation, $|S| \geq Stop(i+1)$, and so $Pr[E_i] \leq (i+1)^{-2}$. By the union bound, the event that algorithm fails is bounded by $\sum_{i=1}^{\infty} \frac{1}{(i+1)^2} < 1/2$.

COROLLARY 4.1. *For Zipfian distributions with parameter θ the estimator above requires $O(D^{\theta} N_{D,\theta} \log D)$ samples.*

The bound we present above is tight up to very small factors. Consider two distributions: $\mathcal{P}$ which is a uniform distribution on m values and $\mathcal{Q}$, a uniform distribution on $m+1$ values. Let S be a sample from one of the distributions, such that $|S| = o(m\frac{\log m}{\log\log m})$ then by standard coupon collector arguments S contains less than m distinct values with high probability. It is easy to see that the relative frequency of the elements in S is the same whether the elements were drawn from $\mathcal{P}$ or from $\mathcal{Q}$, and so $\mathcal{P}$ and $\mathcal{Q}$ are indistinguishable.

THEOREM 4.2. *There exist c-regular families of distributions $\mathcal{F}$ on which any algorithm requires $\Omega(D\frac{\log D}{\log\log D})$ samples.*

5 Approximate Algorithm

Although the above algorithm provides us with analytical guarantees about the number of samples in the distribution, the required number of samples can be high. In some applications there exists a hard limit on the running time of the estimator, thus we look for a procedure

which returns the best possible estimate on the number of distinct values given a sample from the database.

In this section we present an estimator which returns $\widehat{D}$ such that with probability at least $(1-\delta)$ the ratio error is bounded by $(1+\epsilon)$. We provide the analysis below for the case of Zipfian distributions parametrized by their skew, θ.

5.1 Zipfian Distributions The algorithm we consider is similar to the maximum likelihood estimator. Recall that $f_{in}(r)$ is the number of distinct elements in a random sample of size r. Let $f^*_{in}(r, D, \theta)$ be the expected number of distinct elements in a random sample of size r coming from a Zipfian distribution of parameter θ on D distinct values.

Observe that $f^*_{in}(r, D, \theta)$ can be computed when D is known. Our estimator returns

$$\widehat{D} \text{ such that } f^*_{in}(r, \widehat{D}, \theta) = f_{in}$$

In other words, our guess for the number of distinct elements would (in expectation) have us see as many distinct values as we did.

5.2 Analysis The analysis of the simple algorithm above proceeds in two parts. First, we show that with high probability the observed value of f_{in} does not deviate by more than a $(1+\epsilon)$ factor from its expected value, $f^*_{in}(r, D, \theta)$. We then show that if this is the case, then $\widehat{D}$ does not deviate from D by a factor of more than $(1+\epsilon)$ with constant probability, provided that our sample size is large enough.

LEMMA 5.1. *In a sample of size r,*

$$Pr[|f_{in} - f^*_{in}(r, D, \theta)| \geq \epsilon f^*_{in}(r, D, \theta)] \leq$$

$$2\exp(-\frac{\epsilon^2 f^*_{in}(r, D, \theta)^2}{2r})$$

Proof. Let X_i be the number of distinct elements that appear in the sample after i samples. Let $Y_i = E[X_r | X_1, \ldots, X_i]$ be the expected number of distinct elements in the sample after r samples given the results of the first i samples. Note that $Y_r = f_{in}$, the number of distinct values observed, and $Y_0 = f^*_{in}(D, \theta)$, the expected number of unique values observed. By definition the Y_is form a Doob martingale. Now consider their successive difference, $|Y_i - Y_{i-1}|$. Since the only information revealed is the location of the i^{th} sample, $|Y_i - Y_{i-1}| \leq 1$.

We can now invoke Azuma's inequality (see Section 4.4 of Motwani and Raghavan [16]) to bound the difference $|Y_r - Y_0|$:

$$Pr[|Y_r - Y_0| \geq \lambda] \leq 2\exp(-\lambda^2/2r)$$

Plugging in the appropriate values for Y_r, Y_0 and $\lambda = \epsilon Y_0$ gives us the desired result.

We can prove an identical lemma for the values of f_{out} and $f^*_{out}(r, D, \theta)$ defined analogously.

COROLLARY 5.1. *After r samples,*

$$Pr[|f_{out} - f^*_{out}(r, D, \theta)| \geq \epsilon f^*_{out}(r, D, \theta)] \leq$$

$$2\exp(-\frac{\epsilon^2 f^*_{out}(r, D, \theta)}{2r})$$

We have shown that the number of distinct elements seen after r samples is very close to its expectation. We now show that this implies that the maximum likelihood estimator will produce a low ratio error.

LEMMA 5.2. *Suppose that $|f_{out} - f^*_{out}(r, D, \theta)| \leq \epsilon f^*_{out}(r, D, \theta)$ and $r \geq \frac{1}{Pr_{\widehat{D}}(\mathcal{P}_{\widehat{D}})}$. Then the ratio error, $\max(\widehat{D}/D, D/\widehat{D}) \leq 1 + 2\epsilon$.*

Proof. For simplicity of notation let $p_i = Pr_i(Z_{D,\theta})$ and $\hat{p}_i = Pr_i(Z_{\widehat{D},\theta})$, and N and $\hat{N}$ be the normalizing values for the two distributions. Let us consider the value of $f^*_{out}(r, D, \theta)$.

$$f^*_{out}(D, \theta) = \sum_{i=1}^{D} (1-p_i)^r$$

The estimate $\widehat{D}$ returned by the algorithm is such that $f^*_{out}(r, \widehat{D}, \theta) = f_{out} \leq f^*_{out}(r, D, \theta)(1+\epsilon)$. Assume that $f_{out} \geq f^*_{out}$ (The proof is identical in the opposite case). Then $\widehat{D} \geq D$ and we seek to bound the ratio $\widehat{D}/D$ from above. By corollary 5.1, $f_{out} \leq (1+\epsilon) f^*_{out}(r, D, \theta)$ w.h.p.

Therefore, $1 + \epsilon \geq$

$$\frac{f^*_{out}(r, \widehat{D}, \theta)}{f^*_{out}(r, D, \theta)} = \frac{\sum_{i=1}^{\widehat{D}} (1-\hat{p}_i)^r}{\sum_{i=1}^{D} (1-p_i)^r} \geq \frac{\sum_{i=\widehat{D}-D+1}^{\widehat{D}} (1-\hat{p}_i)^r}{\sum_{i=1}^{D} (1-p_i)^r}$$

The second inequality follows since all of the terms in the numerator are positive. Now consider the ratio of the i^{th} terms of each sum. Since the ratio of the sum is bounded, there must exist at least one value i where the ratio of the individual terms is bounded by $(1+\epsilon)$. A simple analysis shows that the ratio decreases as i increases, and thus the lowest ratio is achieved by the last term.

$$1+\epsilon \geq \frac{(1-\hat{p}_{\hat{D}})^r}{(1-p_D)^r} = \left(\frac{1-\hat{p}_{\hat{D}}}{1-p_D}\right)^r$$
$$\geq \left(1+\frac{p_D-\hat{p}_{\hat{D}}}{1-p_D}\right)^r \geq \exp\left(r\frac{p_D-\hat{p}_{\hat{D}}}{1-p_D}\right)$$
$$2\epsilon \geq r\cdot\frac{p_D-\hat{p}_{\hat{D}}}{1-p_D} \geq r(p_D-\hat{p}_{\hat{D}})$$
$$\geq \frac{1}{\hat{p}_{\hat{D}}}(p_D-\hat{p}_{\hat{D}})$$
$$1+2\epsilon \geq \frac{p_D}{\hat{p}_{\hat{D}}} = \left(\frac{\hat{D}}{D}\right)^{\theta}\cdot\left(\frac{\hat{N}}{N}\right)$$

For $\theta \geq 1$ the result follows since $\hat{N} \geq N$. For $\theta < 1$ it can be shown that $\frac{\hat{N}}{N} \geq \left(\frac{\hat{D}}{D}\right)^{1-\theta}$. Combining the two inequalities we get $\frac{\hat{D}}{D} \leq 1+2\epsilon$.

The probability of success of this procedure is $2\exp(-\frac{\epsilon^2 f^*_{out}(r,D,\theta)}{2r})$. To establish the result with constant probability, we have to show that even after r as above samples, $f^*_{out}(D,\theta)$ is sufficiently large.

LEMMA 5.3. *After* $r = (Pr_{\hat{D}}(P_{\hat{D}}))^{-1} = \hat{N}\hat{D}^{\theta}$ *samples,* $f^*_{out}(r,D,\theta) = \Omega(D)$.

Proof. Consider the elements of rank $D/2$ through D, and let i be one of these elements. $Pr_i(\mathcal{P}_D) = \frac{1}{i^{\theta}N} \leq \frac{1}{(D/2)^{\theta}N}$. Therefore the probability that i is not present in the sample of size r is at most $(1-\frac{1}{(D/2)^{\theta}N})^r \leq \exp(-\frac{\hat{N}\hat{D}^{\theta}}{(D/2)^{\theta}N}) \leq \exp(-2^{\theta}(1+2\epsilon)^{2\theta}) = \Theta(1)$. The expected number of items not present is therefore $\Omega(D)$.

From the above two lemmas, the main theorem for Zipfian distributions follows:

THEOREM 5.1. *Given* $r = N_{D,\theta}(1+2\epsilon)((1+2\epsilon)D)^{\theta}$ *samples the algorithm will produce an estimate* $\widehat{D}$*, such that* $\max(\widehat{D}/D, D/\widehat{D}) < 1+\epsilon$ *with probability of error less than* $2\exp(-\Omega(D)\epsilon^2)$.

We have chosen here to analyze in detail the case of Zipfian distributions. Observe that the main algorithm works even for non-Zipfian distributions. As long as the value of $f^*_{out}(r,\mathcal{P})$ can be estimated the algorithm presented above is well defined. However, the exact value for r and the estimation error need to be recomputed for each family of distributions.

6 Experimental Results

In this section we validate our estimator by comparing it against GEE (the only other sampling based estimator with analytical guarantees), as well as AE (Adaptive Estimator), which was shown to outperform all of the other heuristic estimators in the experiments of Charikar et al [3]. We will refer to our estimator as ZE (for Zipfian Estimator). We first test the three estimators on synthetic data. We generate datasets according to a Zipfian distribution with skew parameter $\theta \in \{0, 0.5, 1\}$. We vary the number of distinct elements from $10k$ to $100k$, and vary the size of the overall database from $100k$ to $1000k$. We present here the results of all three estimators on a dataset of $500,000$ elements drawn under the corresponding Zipfian distribution on 50000 elements. The results for the other synthetic scenarios were almost identical to the ones shown.

Further, we tested the estimators on several real-world datasets that we assumed followed a Zipfian distribution. We present the results on the *Router* dataset was obtained from [18]. It is a packet trace from the Internet Traffic Archive. We are trying to predict the number of distinct IP addresses served by the router. Although this distribution is not a pure Zipfian, as the probabilities of the most frequent values and the least frequent values are a little bit skewed, the bulk of the data follows a Zipfian distribution with $\theta \approx 1.6$.

6.1 Estimating Zipfian Skew Parameter All of the analytical results above assumed that the parameter θ was known to us ahead of time. In practice, we can estimate the parameter from the data sampled for distinct value counts. Let f_i be the frequency of the i^{th} most common element. Then in expectation, $f_i = rp_i = \frac{r}{Z}i^{-\theta}$, and $\log f_i = \log\frac{r}{Z} - \theta\log i$. Since $\frac{r}{Z}$ is independent of i, we can estimate θ by doing linear regression on the log-log scale of the f_i vs i data. Many of the real world datasets (including *Router*) follow a Zipfian distribution for the bulk of the data, but not for the first or the last few elements, which can change the θ parameter of the sample. To counteract this problem we ignored the top 100 frequencies, as well as all of the elements which did not appear at least 10 times in the sample while estimating the value of the θ. Note that the value of the parameter was estimated for the synthetic datasets as well, even when we knew the exact value that generated the dataset.

6.2 Discussion In the synthetically generated datasets the ZE estimator was competitive with AE and often outperformed it. This is not surprising since ZE was designed particularly for Zipfian datasets. The GEE estimator performed poorly on most of the data, often having the results err by more than a factor of 5 even after a large sample.

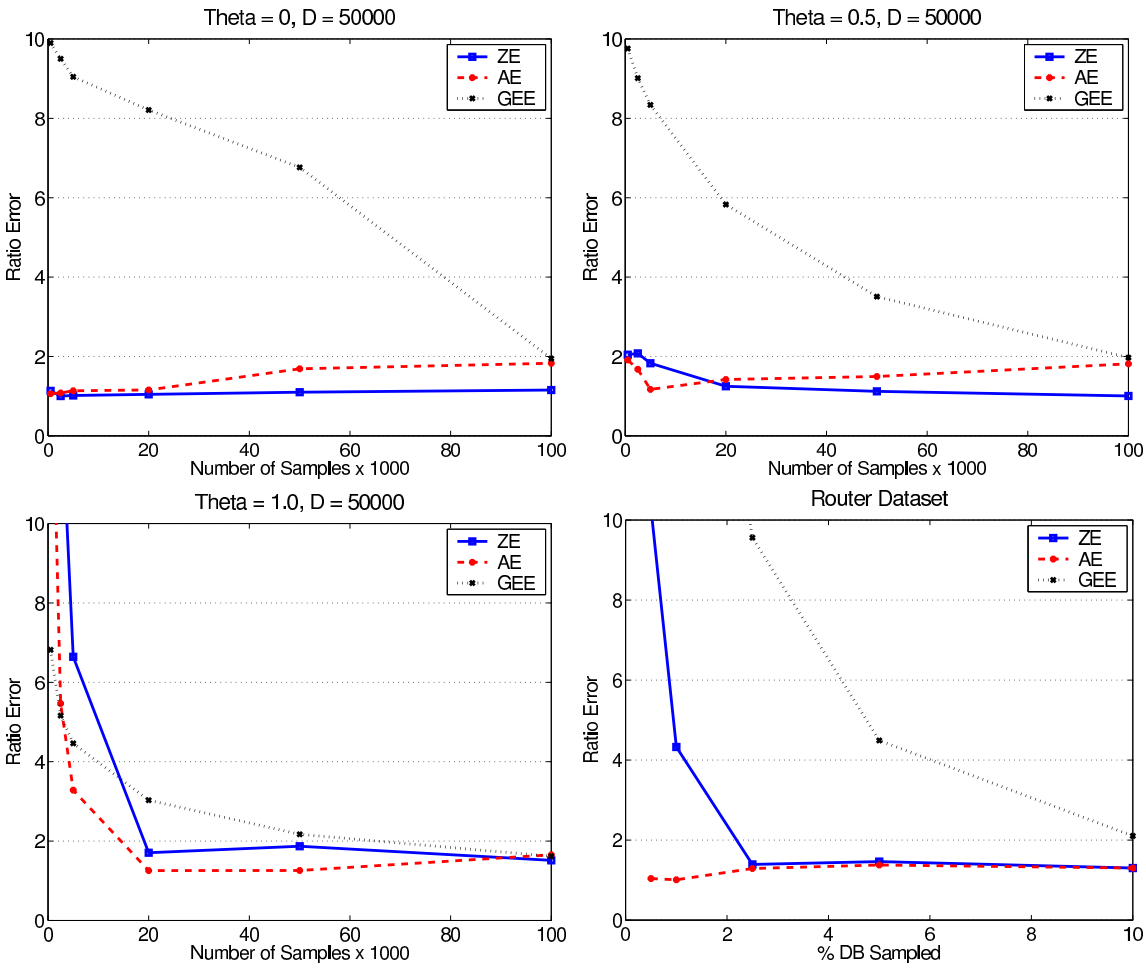

Figure 1: Empirical results on synthetic and real world data

On the real-world dataset AE performed very well, and ZE was competitive after about 2.5% of the database was sampled. One must keep in mind that the router dataset had very high skew ($\theta \approx 1.6$), and ZE was given fewer estimates than would be required by the theoretical guarantees, but performed well nonetheless. On real world data, the Zipfian Estimator, ZE grossly outperformed the other estimator with guaranteed error bounds. The results were comparable only after a 10% fraction of the database was sampled. It is important to note that because of the random access nature of the estimation algorithms, a 10% sample requires almost as much time to compute as a full linear scan of the database.

Although ZE and AE performed equally well, one must remember that the Zipfian Estimator presented here is guaranteed to perform well with high probability on all Zipfian inputs, while the AE estimator is only a heuristic and may perform poorly on some of the inputs. In particular, the error of AE often rises as more samples are taken from the database. When compared to the only other estimator which has guarantees on its results, GEE, the Zipfian estimator performed much better, often giving results more than 10 times more accurate on the same dataset.

References

[1] Alon, N., Matias, Y., and Szegedy, M. The space complexity of approximating the frequency moments. In *Proceedings of the 28th ACM Symposium on the Theory of Computing*, 1996, pp. 20–29.

[2] Bunge, J., and Fitzpatrick, M. Estimating the Number of Species: A review. *Journal of the American Statistical Association* 88(1993): 364–373.

[3] Charikar, M., Chaudhuri S., Motwani, R., and Narasayya, V. Towards Estimation Error Guarantees for Distinct Values. In *Proceedings of the Nineteenth*

ACM Symposium on Principles of Database System, 2000, pp. 268–279.

[4] Chaudhuri, S., Das, G., and Srivastava, U. Effective Use of Block-Level Sampling in Statistics Estimation. In *Proceedings of ACM-SIGMOD*, 2004.

[5] Durand, M., and Flajolet, P. Loglog Counting of Large Cardinalities. In *Proceedings of 11th Annual European Symposium on Algorithms (ESA)*, 2003, pp. 605–617.

[6] Flajolet P., and Martin, G.N. Probabilistic counting. In *Proceedings of the IEEE Symposium on the Foundations of Computer Science*, 1983, pp 76–82.

[7] Gibbons, P.B. Distinct Sampling for Highly-Accurate Answers to Distinct Values Queries and Event Reports. In *Proceedings of the 27th International Conference on Very Large Databases*, 2001.

[8] Goodman, L. On the estimation of the number of classes in a population. *Annals of Math. Stat.* 1949, pp. 72-579.

[9] Haas, P.J., Naughton, J., F., Seshadri, S., and Stokes, L. Sampling-based Estimation of the Number of Distinct Values of an Attribute. In *Proceedings of the 21st International Conference on Very Large Databases*, 1995.

[10] Haas, P.J., and Stokes, L. Estimating the number of classes in a finite population. In *Journal of the American Statistical Association* 1998, pp. 1475–1487.

[11] Heising, W.P. *IBM Systems J.* 2 (1963).

[12] Hou, W., Ozsoyoglu, G., and Taneja, B. Statistical estimators for relational algebra expressions. In *Proceedings of the 7th ACM Symposium on Principles of Database Systems*, 1988.

[13] Hou, W., Ozsoyoglu, G., and Taneja, B. Processing aggregate relational queries with hard time constraints. In *Proceedings of the ACM-SIGMOD International Conference on Management of Data*, 1989.

[14] Kleinberg, J., Kumar, R., Raghavan, P., Rajagopalan, S., and Tomkins, A. The Web as a graph: measurements, models and methods. In *Proceedings of the International Conference on Combinatorics and Computing*, 1999.

[15] Knuth, D.E. **Sorting and Searching**. Volume 3 of *The Art of Computer Programming*. Addison-Wesley, Reading, MA, 1971.

[16] Motwani, R., and Raghavan, P. **Randomized Algorithms.**Cambridge University Press, 1995.

[17] Mitzenmacher, M. A Brief History of Generative Models for Power Law and Lognormal Distributions. In *Proceedings of the 39th Annual Allerton Conference on Communication, Control, and Computing*, 2001, pp. 182-191.

[18] A packet trace from the internet traffic archive. `http://ita.ee.lbl.gov/html/contrib/DEC-PKT.html`.

[19] Shlosser, A. On estimation of the size of the dictionary of a long text on the basis of a sample. *Engrg Cybernetics*, 1981, pp. 97–102.

[20] Zipf, G. **The Psycho-Biology of Language.** Houghton Mifflin, Boston, MA, 1935.